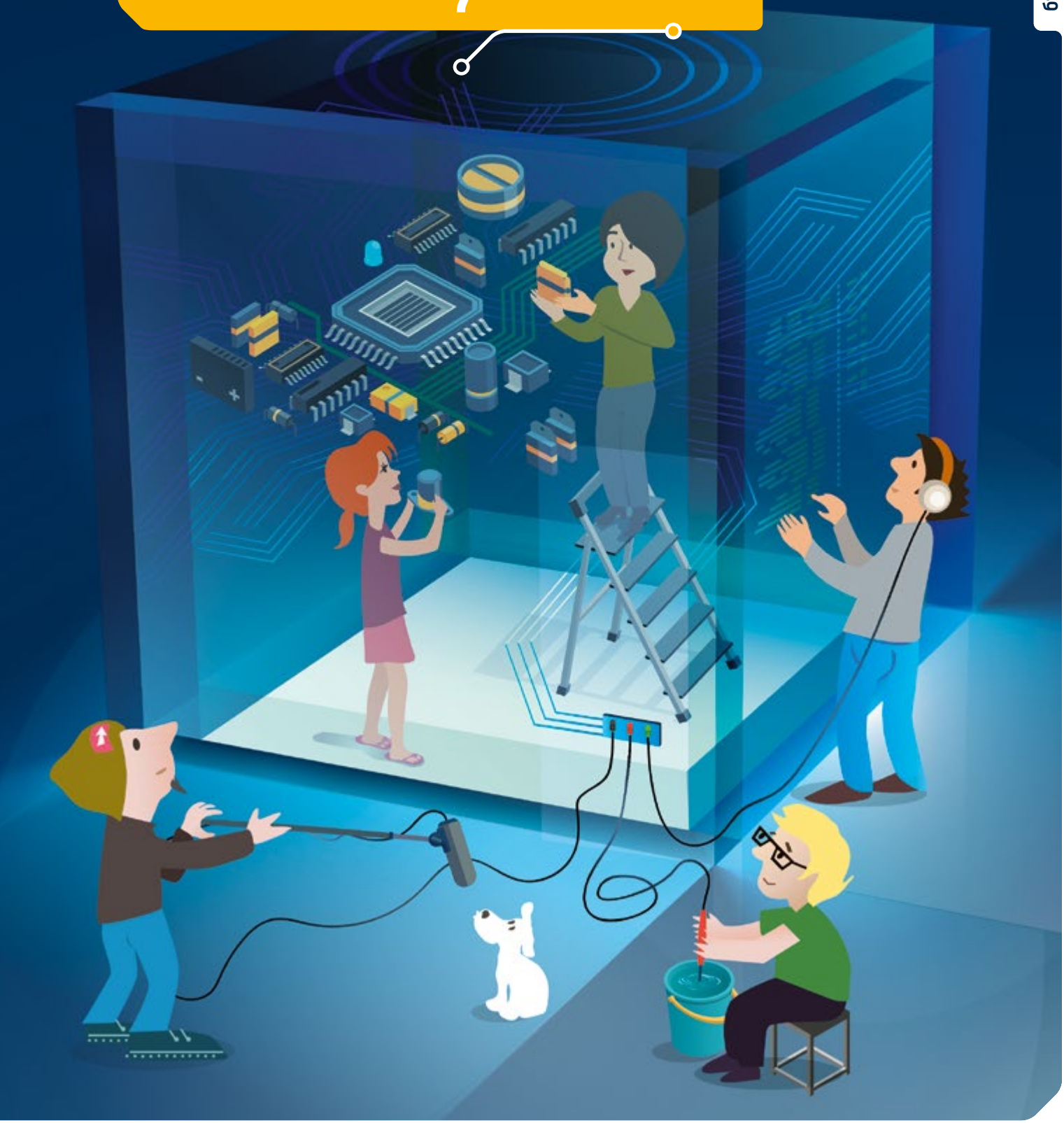
SCIENCE ON STAGE GERMANY

THE EUROPEAN NETWORK FOR SCIENCE TEACHERS

Програмирането в STEM обучението



<Печат>

<Публикувано от>
Science on Stage Deutschland e. V.
Ам Борсигтурм 15
13507 Берлин, Германия

<Главен координатор>
↳ **Д-р Йорг Гуцанк**, Гимназия Лайбниц | Международно училище в Дортмунд, Дортмунд, Германия
Председател на Science on Stage Deutschland e. V.

<Координатори>
↳ **Себастиан Фънк**, Вила Виверсбуш, Велберт – Лангенберг, Германия, Член на съвета на Science on Stage Deutschland e. V.
↳ **Жан – Люк Рихтер**, Гимназия Jean-Baptiste Schwilgué, Селестат, Франция, Заместник-председател на Наука на сцената Франция
↳ **Бернар Шриек (ret.)**, Гимназия Marien, Верл, Германия

<Цялостна координация и редактиране>
↳ **Даниела Нюман**, Проектен мениджър на Science on Stage Deutschland e. V.
↳ **Стефани Шлунк**, Изпълнителен мениджър на Science on Stage Deutschland e. V.
↳ **Йохана Шулце**, Заместник изпълнителен мениджър на Science on Stage Deutschland e. V.

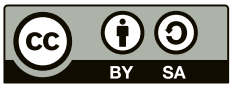
<Корекция и превод>
Мария Петрова, Петър Андреев, Десислава Цокова и Моника Ковачка-Димитрова

<Дизайн>
WEBERSUPIRAN.berlin

<Илюстрации>
Рупърт Таке, TricomKommunikation und Verlag GmbH

<С подкрепата на>
SAP SE

<За поръчка>
www.science-on-stage.bg
office@frgi.bg



Първо издание 2020

<Съдържание>

<Приветствие>

За мен е удоволствие да подкрепя тази брошура, касаеща обучението в областта на Науката, Технологиите, Инженерството и Математиката – важните STEM (Science, Technology, Engineering and Mathematics) дисциплини – и в по-общ план – проектът *„Програмирането в STEM обучението“*.

Дисциплините STEM са от решаващо значение за изграждането на конкурентна и устойчива Европа на бъдещето. Получаването на знания и достигането на високи постижения е ключова част от амбицията ни да създадем истинско Европейско образователно пространство до 2025 г. И все пак, ние сме изправени пред пропуски в уменията. Трябва да направим повече за популяризирането на STEM дисциплините в Европа. По-конкретно, преподаването на STEM предметите трябва да бъде привлекателен избор на кариера и да има повече модели за подражание, особено жени.

Това не се отнася само за икономическия растеж и развитие. Обучението в STEM дисциплините трябва да бъде приобщаващо, да дава възможност на ученици с различни способности и опит да се ангажират и да се възползват максимално от своите таланти. За да изградят бъдещето, младите хора се нуждаят от правилните умения и нагласи – и владеенето на STEM дисциплините трябва да бъде в основата им.

Чрез науката можем да разберем толкова много за нашето минало и настояще, което ни позволява да работим за по-добро бъдеще, за бъдеще на знанието. За да могат идните поколения да живеят в по-осъзнато общество, с по-богато познание за света, в който живеем.

Искам да благодаря на Science on Stage Германия, на Европейската мрежа на учителите по природни науки, която е в основата на този проект, и на SAP SE за финансовата подкрепа на проекти за програмиране.

Учители от седем европейски държави помогнаха за разработването на конкретни примери и практически съвети за придобиване на умения за програмиране. Това показва какво можем да постигнем, когато се съберем, колко общо имаме и колко можем да научим един от друг.

Проекти като *„Програмирането в STEM обучението“* имат ключова роля в популяризирането на STEM дисциплините в училищата, помагайки на младежта в Европа да придобие жизненоважни умения, необходими за успеха в живота. Поздравявам всички участващи и се надявам вашият пример да вдъхнови останалите.

Тибор Наврачич

Европейски комисар по въпросите на образованието, културата, младежта и спорта

<Приветствие>

Според прогноза на Световния икономически форум, около 65 на сто от децата, които днес започват основно училище, ще имат професии, които все още не съществуват. Въпреки това е ясно, че тези деца ще се справят по-лесно, ако имат определени технически умения; дигитализацията на нашата икономика просто не може да бъде спряна. Следователно, освен четене, писане и математика, работата с новите технологии се превърна в ключова тема за образованието.

SAP от години участва в общоевропейски инициативи за образование и обучение на деца и младежи, включително в областите на роботиката (*„Първа лего лига“*) и технологиите за програмиране (*„Запознайте се и програмирайте“*). Нашата цел е да запознаем младите хора с новите технологии по забавен начин и да улесним старта им в бъдещите им кариери.

В наши дни много учители също искат да дадат на своите ученици основни технически и дигитални знания, които да могат да ползват впоследствие. Това не означава непременно, че тези учители трябва да бъдат експерти. На тях са им необходими практически, изпитани работни материали за усвояване на технически умения по различните предмети и за различните нива на обучение.

В този смисъл, ние улесняваме придобиването на дигитални умения в ежедневния училищен живот и предоставяме учебни материали, които да се използват в обучението в STEM дисциплините. Вече инициирахме многобройни проекти в подкрепа на училищата заедно със Science on Stage Германия, включително настоящите учебни материали от проекта *„Програмирането в STEM обучението“*.

Радваме се на успешното ни сътрудничество със Science on Stage Германия в друг подобен проект и сме убедени, че и този ще има пълен успех. Искаме също да благодарим на учителите, оказали пълната си подкрепа за него.

<Предговор>

Тази книжка е специална в много отношения. Позволете ми да обясня какво имам предвид.

Писането на код е основно умение в днешния свят и е особено важно в науката, технологиите, инженерството и математиката (STEM). Уменията за програмиране се превръщат във все по-търсени и необходими във всички области на живота ни и вече не могат да бъдат оставяни само на ИТ специалистите. Следователно програмирането трябва да се преподава не само в часовете по компютърни науки, но и във всеки друг предмет. Европейските учебни програми по STEM дисциплините обаче не отговарят изцяло на тази нужда. В проекта *„Програмирането в STEM обучението“* на Science on Stage, Европейската мрежа за учителите по природни науки разработиха концепции за преподаване в опит да се преодолеят пропуските в уменията. Общата цел на мрежата е да предостави на европейските преподаватели по STEM платформа за обмен на идеи и най-добри практики. Science on Stage достигна 100 000 преподаватели в над 30 държави членки.

Някои от най-добрите учители в Европа допринесоха с идеите си за тази брошура. 23 учители от седем държави се срещнаха лично, за да обменят своите идеи за програмирането в образованието по природни науки по време на този 18-месечен проект. Всички учители, участващи в него, инвестираха много време и усилия в този процес.

Учител по природни науки и учител по компютърни науки от всяка една от седемте страни обмениха концепции за преподаване с екип от поне една друга държава. Те обсъдиха и изпробваха тези концепции в собствените си часове в двете страни, за да гарантират, че материалът, който държите в ръцете си, е полезен в реалните уроци и е изпитан от нашите експерти, самите учители.

Акцентът е върху програмирането на малки електронни устройства като Arduino, Calliope mini или Raspberry Pi компютри. Те са евтини устройства, които могат да се ползват за задълбоченото решаване на задачи, което ги прави идеални за училищата.

Участниците разработиха 11 учебни модула в областите „Фундаменталната наука в 1-ци и 0-ли", „Микроконтрол на света“ и „Околна среда 4.0“. Те са отлични примери за това, което бихте могли да направите в различните учебни програми по биология, химия и физика.

Проектът *„Програмирането в STEM обучението“* стана възможен благодарение на усилията на нашите ентусиазирани участници, които вършеха цялата тази работа в свободното си време, извън редовната си преподавателска работа. Благодаря на всички тях за тази вдъхновяваща публикация. Много благодаря и на другите координатори Жан-Люк Рихтер, Бернд Шрик и Себастиан Фънк, които свършиха чудесна работа, като обобщиха разнородните мисли и идеи на учители от различни културни и научни среди в едно хомогенно произведение. Искам също да благодаря на SAP SE и по-специално на Габриеле Хартман и Йенс Мениг. Тази публикация не би била възможна и без тяхната постоянна подкрепа.

„Програмирането в STEM обучението“ е специална книжка, която е разработена и тествана от учители за техните колеги в Европа и извън нея. Надяваме се да ви вдъхнови и съвсем скоро да започнете свои проекти в класната си стая!

Д-р Йорг Гутчанк
Председател на Science on Stage Deutschland e.V.

<Автори>

<Фамилия>	<Име>	<Държава>	<Глава>
Абад Небот	Имакулада	Испания	Микроконтрол на света
Ботельо	Лусио	Португалия	Околна среда 4.0
Компт Джове	Пери	Испания	Микроконтрол на света
Фернандес	Лиляна	Португалия	Околна среда 4.0
Фънк	Себастиан	Германия	Координатор
Георгулакис	Георгиос	Гърция	Фундаменталната наука в 1-ци и 0-ли
Гюргея	Михаела Ирина	Румъния	Фундаменталната наука в 1-ци и 0-ли
Гутчанк	Йорг	Германия	Координатор Фундаменталната наука в 1-ци и 0-ли
Ханчл	Мирек	Германия	Микроконтрол на света
Ивара	Люк	Белгия	Фундаменталната наука в 1-ци и 0-ли
Карагьоргу	Елефтерия	Гърция	Микроконтрол на света
Лукаш	Аннамария	Румъния	Околна среда 4.0
Майер	Андреас	Германия	Околна среда 4.0
Мествиришвили	Илия	Грузия	Фундаменталната наука в 1-ци и 0-ли
Николини	Марко	Белгия / Италия	Фундаменталната наука в 1-ци и 0-ли
Падин	Беатрис	Испания	Околна среда 4.0
Poncela	Elena	Испания	Околна среда 4.0
Рациу	Камелия Йоана	Румъния	Фундаменталната наука в 1-ци и 0-ли
Рейс	Хорхе	Португалия	Околна среда 4.0
Рихтер	Жан Люк	Франция	Координатор Околна среда 4.0
Шриек	Бернар	Германия	Координатор Микроконтрол на света
Шапакидзе	Дейвид	Грузия	Фундаменталната наука в 1-ци и 0-ли
Тома	Корина Лавиния	Румъния	Фундаменталната наука в 1-ци и 0-ли
Цилики	Севасти	Гърция	Микроконтрол на света
Цуцудакис	Астринос	Гърция	Фундаменталната наука в 1-ци и 0-ли
ван дер Бил	Соня	Германия	Околна среда 4.0
Уинклер	Джулия	Германия	Микроконтрол на света

Специални благодарности на Габриеле Хартман и Йенс Мениг от SAP SE за оказаната подкрепа!
Бихме искали да благодарим и на Холгер Бах и Пол Нугент за ценните им съвети по време на редактирането!

* Специални благодарности и на преводачите и редакторите на българското издание: Мария Петрова, Петър Андреев и Десислава Цокова.

Кодиране на H_2O

<Автор> Беатриз Падин
<Автор> Елена Пончела

<Инфо>

<Ключови думи> вода, сензори, ефективност, събиране на данни, околна среда, изпарение, кондензация, разтвори, смеси, дизайн, топлина и температура, топлопроводници и изолатори, слънчева енергия, инфрачервено излъчване, отразяване

<Дисциплини> физика, природни науки, химия, компютърни науки, математика

<Възраст на учениците> 13–15

<Хардуер> Arduino[1], Calliope mini[2], Raspberry Pi[3]

<Езици> Arduino[4], Python[5], блоково програмиране

<Ниво на програмиране> средно

<Резюме>

Учениците ще проектират, изградят и тестват соларен дестилатор за пречистване на вода. Те ще програмират сензори, за да измерят ефективността му.

<Въведение>

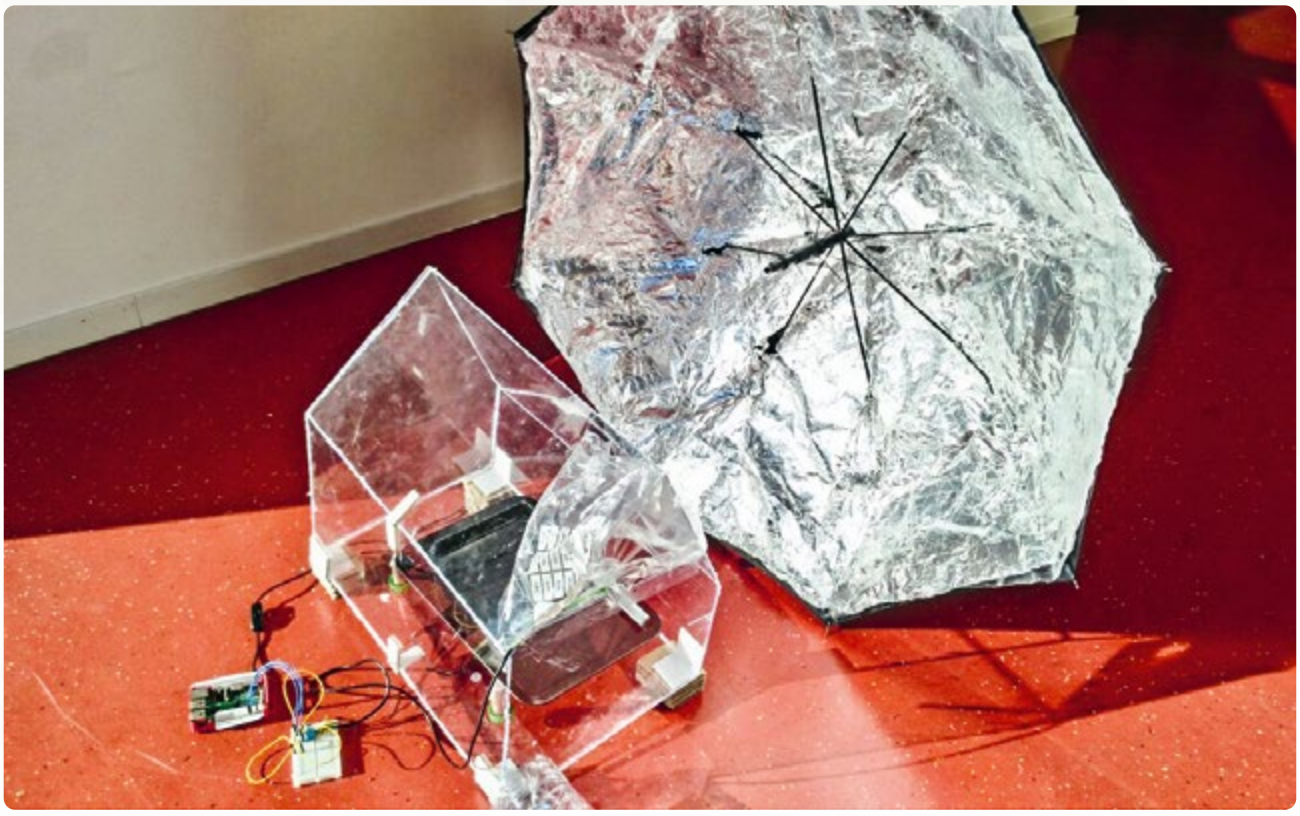
Ще обхванем следните понятия по физика:

- Промени в състоянието (по-конкретно – изпарение и кондензация) и техните основни характеристики;
- Фактори, които влияят върху процеса на изпарение (температура, повърхност и т.н.);
- Ефекти от загряването: промени в състоянието;
- Разлика между възобновяеми (слънчева енергия) и невъзобновяеми енергийни източници;
- Инфрачервеното излъчване и неговата роля за транспортиране на топлина от слънцето;
- Отражение на излъчването от различни повърхности;
- Методи за разделяне на смеси;
- Решения: „Какви са те?“; концентрация (грам/литър и обемни проценти и т.н.).

В зависимост от нивото на знания на учениците, те ще открият някои от тези понятия сами. Други ще трябва да бъдат обяснени от учителя и след това учениците да експериментират с тях.

Целта е да се проектира и изгради най-ефективният соларен дестилатор за пречистване на вода. Може да се използва само слънчева енергия. Първоначално ефективността на соларния дестилатор ще бъде определена чрез изчисляване на обемния процент на пречистената вода, която се получава. След това учениците ще програмират различни сензори, за да анализират ефективността на своите проекти.

Основната задача на тази дейност е да се програмират сензори. Поради това ще е необходимо учителите да



Соларен дестилатор

имат умения за програмиране и някои основни хардуерни познания.

Те ще трябва да знаят как да изградят верига, необходима за свързване на сензорите към платката на микроконтролера. В зависимост от уменията си, те могат да използват блоково програмиране (Calliope mini[2], Snap4Arduino[6], и т.н.) или текстово програмиране (Arduino[4], Python[5], и т.н.) за програмиране на сензорите.

<Какво правят учениците/учителите>

Този модул се състои от три части: проектиране и изграждане на соларния дестилатор, кодиране на сензорите и тестване на соларния дестилатор.

<Първа част: Проектиране и изграждане на соларен дестилатор>

Проектът ще бъде представен на учениците. Те ще тестват модел на слънчев дестилатор за пречистване на мръсна вода, като чрез измерване на количеството събрана пречистена вода, те ще изчислят ефективността на този дизайн. Докато правят това ще използват понятия от физиката, като промени в състоянието, слънчева енергия и решаване на проблеми.

След това учениците ще бъдат насърчени да подобрят този първоначален модел. За целта те ще трябва да работят в групи (2–3 ученици в група). Тази част може да бъде разделена на няколко задачи:

1. Учениците ще потърсят информация за соларните дестилатори – как работят, различни модели, които вече се използват и т.н. По време на този процес те ще трябва да обърнат внимание на следните въпроси:
   a. Процес на изпарение: Кои са основните фактори, влияещи върху този процес? Помислете за формата на съда, където ще поставите мръсната вода. Кое е по-добре – да имате голяма или малка повърхност, по-дълбок или по-плитък съд? Има ли значение цветът на контейнера?
   b. Процес на кондензация: Какво е необходимо, за да се получи конденз? Каква повърхност за кондензация на водата трябва да проектирате – голяма или малка? Как ще придвижите чистата вода до мястото, където искате да я съберете?
   c. Процес на излъчване: Как можете да увеличите максимално инфрачервеното излъчване, което достига до соларният ви дестилатор? Как можете да достигнете максималната възможна температура във вашия соларен дестилатор? Помислете за възможността да използвате повърхност, покрита с алуминиево фолио, за да отразите слънчевата светлина към соларния дестилатор.
2. Групите трябва да направят свои проекти и да ги представят пред учителя.
3. След като учителят ги одобри, учениците трябва да потърсят подходящи материали (в училище, вкъщи, да поръчат онлайн и т.н.) и да построят соларния си дестилатор.
4. Учениците ще определят ефективността на соларните си дестилатори без сензори. Използвайки мерителен цилиндър, те ще измерват обема на мръсната вода в началото на експеримента и обема на събраната чиста вода в края, като ще използват следното математическо уравнение:

$$Ефективност = \frac{обем\ събрана\ чиста\ вода}{обем\ на\ мръсната\ вода\ в\ началото}$$

За втората и третата част на тази дейност е предоставено ръководство "стъпка по стъпка", включващо задачи и въпроси за учениците.

Основна цел е да се даде на учениците различен избор по отношение на сензорите, езиците за програмиране и хардуера, като това зависи от всяко училище/клас по отделно (налични материали, познания по езици за програмиране и т.н.).

<Втора част: Кодиране на сензорите>

Само най-добрите модели ще се тестват със сензорите. В тази част ще имате нужда да:

1. Изберете платката на микроконтролера, езика за програмиране и сензорите, с които ще работите. Отговорете на следните въпроси, които ще ви помогнат да вземете най-доброто решение:
   a. Блоково или текстово програмиране ще използвате? Ако изберете блоково програмиране, помислете дали да използвате Calliope mini[2], или като алтернатива, да програмирате с Raspberry Pi[3] със Scratch[8]. В случай, че се спрете на текстово програмиране, бихте могли да използвате Arduino[1] или Python[5] за Raspberry Pi
   b. Цифрови или аналогови сензори ще използвате? Ако решите последното, Arduino е най-добрият избор.

2. Изберете сензорите, които искате да използвате. За да улесните нещата, изберете не повече от два сензора. Те могат да са за температура, влажност, дъжд и инфрачервено излъчване. Преди да решите, помислете за спецификациите на всеки сензор. Изходният сигнал аналогов ли е, или е цифров, само

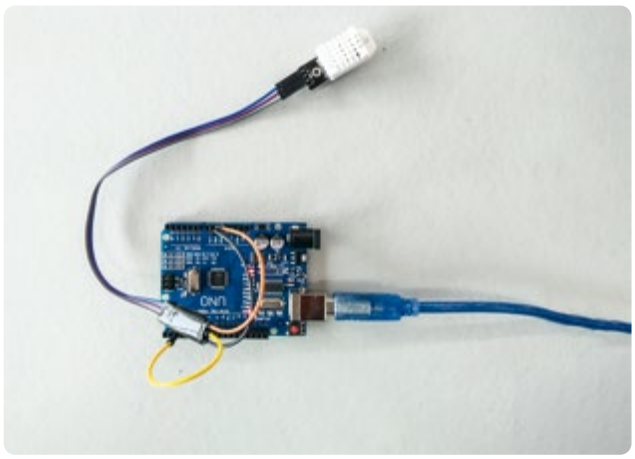
Сензор за пламък

с две възможни стойности (вярно / невярно)? Как изходният сигнал се свързва със стойността на параметъра, който измервате? Правопропорционални ли са? Увеличава ли се изходният сигнал с намаляване на параметъра?

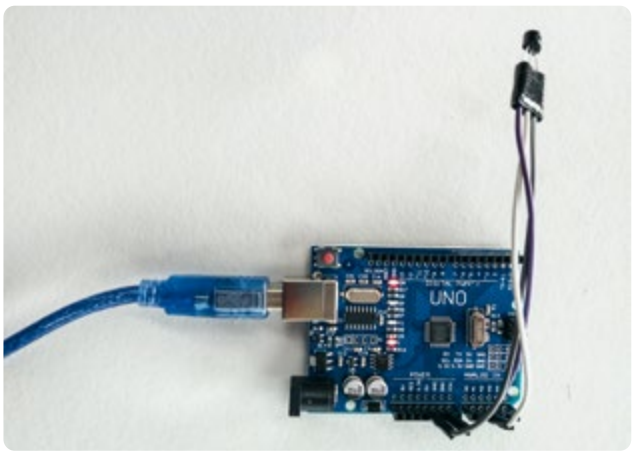
3. Изградете схемата, за да свържете сензора си към платката на микроконтролера. Използвайте предоставените допълнителни ресурси[7] или потърсете примери в Интернет.

4. Програмирайте сензорите. Трябва да следвате следните стъпки:
   a. Напишете какво искате да прави вашата програма, като вземете предвид характеристиките на сензорите, които сте избрали. Искате ли вашата програма да показва само стойността на измерения параметър? Искате ли да покажете максималните и минималните стойности? Дали вашият сензор показва действителната стойност на параметъра или трябва да направите някакви допълнителни изчисления?
   b. Напишете програмата си. Не забравяйте да напишете коментари за вашия код. Можете да използвате допълнителните ресурси, предоставени от вашия учител като ръководство.[7]

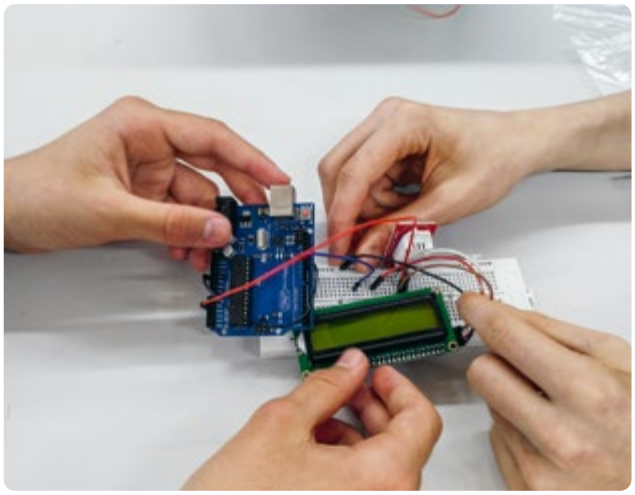
5. Тествайте програмата си. Работи ли според очакванията ви?
   Примери:
   ↳ Можете да измервате количеството инфрачервено излъчване, което достига до соларния ви дестилатор чрез сензор за пламък и Arduino UNO[1]. Ако използвате повърхност, покрита с алумини-



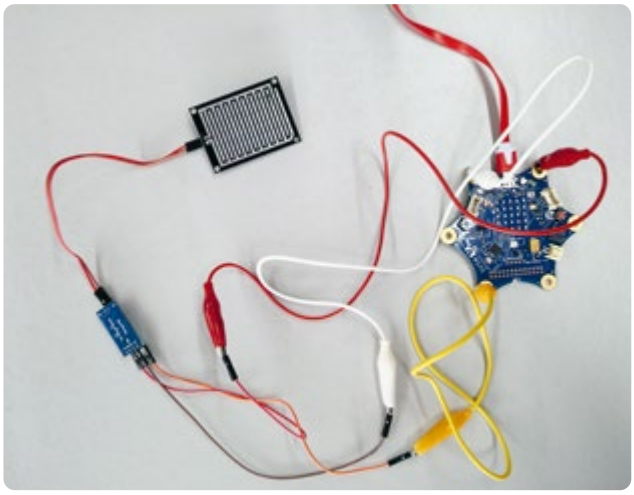
Arduino със сензор за влажност



Arduino със сензор за температура



Arduino with LCD екран и сензор за влажност



Сензор за дъжд с Calliope mini

ево фолио, използвайте този сензор, за да проверите дали лъчите се отразяват в соларния дестилатор.

- Можете да запишете максималната температура и относителната влажност, достигната вътре в соларния дестилатор със сензор за влажност и температура DHT11 или DHT22 и вашия Arduino UNO.
- Можете да напишете програма с Python 3, за да определите времето, което отнема на първите капки вода да се кондензират, като използвате датчика за дъжд FR-04 и Raspberry Pi[3]. YМожете също да използвате този сензор с Calliope mini и да го програмирате със Snap![10](блоков език за програмиране).

<Трета част: Тестване на соларния дестилатор със сензори>

Екипът ви трябва да използва програмираните сензори, за да тества и сравни дизайна (проекта) си на соларен дестилатор с друг подобен. За целта соларните дестилатори трябва да работят при едни и същи условия и да използват един и същ тип сензори.
Ще анализирате причините, поради които ефективността е различна, отбелязвайки ключови факти като факторите, които влияят върху изпарението и т.н.

Как да подобрим ефективността на соларния дестилатор:

- Тествайте го по обяд в слънчев ден;
- Изчакайте достатъчно дълго, за да може оборудването ви да достигне най-високата възможна температура;
- Уверете се, че соларният дестилатор е херметично затворен, за да избегнете загуба на водна пара;
- Изваждайте чистата вода непрекъснато от съда, в който я събирате, за да не се изпари;
- Оцветете мръсната вода, за да гарантирате, че соларният дестилатор работи правилно;
- Изберете широк, черен контейнер за мръсната вода;
- Използвайте чадър, покрит с алуминиево фолио, за отразяване на слънчевата светлина към соларния дестилатор.

<Резултати>

Първата част от дейността (изграждането на соларен дестилатор) доведе до създаването на проекти, които значително се различават по своята ефективност. Когато са тествани в слънчеви дни през май / юни в Испания, най-добрият модел дестилатор пречиства 95% от мръсната вода за 24 часа. 54% ефективност се получава за 4 часа. Въпреки това някои дестилатори въобще не събират чиста вода, поради различни недостатъци в дизайна им.

Чрез използването на сензори, учениците получиха следните резултати:

- Максималната температура, достигната в соларния дестилатор след един час работа в слънчев ден беше 65 °C.
- Анализира се влиянието на цвета на контейнера за мръсна вода. При сравнение на бял и черен контейнер, оставен на слънце за няколко минути, се установи, че температурата на водата в черния контейнер е с почти 5 °C по-висока от тази в белия.
- Връзката между температурата на водата и скоростта на изпарение е изследвана с помощта на сензора за влажност. Относителната влажност в соларен дестилатор, съдържащ вода със стайна температура, беше 55%. При нагряване на водата до

45 °C, относителната влажност се увеличава до 98% само за няколко секунди.
- Количеството инфрачервено излъчване, отразено от метална повърхност към соларния дестилатор, беше измерено с помощта на сензора за пламък. Стар чадър, покрит с алуминиево фолио, беше много ефективен при отразяване на инфрачервените лъчи.

<Заключение>

Учениците ще използват своята креативност и умения за комплексно мислене, за да проектират най-ефективния слънчев дестилатор. Този проект ще им даде възможност да развият уменията си за критично мислене и решаване на проблеми, което е много полезно за учениците и ще могат да ги използват в ежедневието си. Те ще научат някои основни понятия по физика (самообучение), като наблюдават, експериментират, тестват и анализират резултатите си, вместо просто да четат за тях в учебниците по физика.

Те ще развият и уменията си за изчисляване, докато програмират сензорите си. Ще правят изчисления по физика (т.е. техните кодове ще си взаимодействат със света на физиката). През цялото време ще следват различните стъпки на научния метод, описан по-горе, за да разработят най-добрия соларен дестилатор и след това да го построят с подходящи материали.

Важно е всеки ученик да участва в проектните задачи. Някои задачи могат да се извършват самостоятелно (търсене на информация, събиране на първоначалните идеи за дизайна ...), за да се гарантира тяхната ангажираност.

Основните проблеми, които могат да възникнат са: някои ученици да нямат подходящи умения за програмиране (поради което сме включили алтернативи за блоково програмиране), достатъчно знания за изграждане на схемите (допълнителните ресурси[7] са много полезни в това отношение) или да не разбират как работят сензорите. Освен това материалите може да не са налични във всички училища, така че може да се наложи да бъдат закупени.

Допълнителни дейности:
- Данните, събрани от сензорите, могат да се съхраняват на SD карта за допълнителен анализ;
- Може да се използва LCD екран за визуализиране на измерванията;
- Internet of Things (IoT): събраните данни могат да бъдат изпращани чрез интернет в реално време, така че да са обществено достъпни;
- Могат да се използват допълнителни сензори, като датчик за CO или други парникови газове, сензор за проводимост, за да се провери дали чистата вода е все още солена, pH сензор, който измерва pH на мръсната и чистата вода и т.н;
- Солеността може да се измери с проби от морска вода;
- Може да се добави метод за дезинфекция на събраната вода.

Учениците могат да използват соларните дестилатори и за изследване на парниковия ефект, фотосинтезата, дишането на клетките, идеалните газове и др. Много сензори могат да бъдат използвани в други проекти за измерване на физични или химични параметри; например за наблюдение на замърсяването на въздуха или качеството на водата.

<Сътрудничество>

По-ефективни ли са соларните дестилатори в страни със слънчево време? Учениците от различни европейски училища могат да споделят своите резултати, използвайки онлайн карта с определяне на местоположението. Солеността на различните места може да бъде сравнена чрез вземане на проби от морска вода.

<Препратки>

[1] www.arduino.cc/
[2] https://calliope.cc/en
[3] www.raspberrypi.org
[4] www.arduino.cc/reference/en/
[5] www.python.org/
[6] http://snap4arduino.rocks/
[7] Всички необходими материали са налични на www.science-on-stage.de/coding-materials.
[8] https://scratch.mit.edu/
[9] www.raspberrypi.org/documentation/usage/python/
[10] https://snap.berkeley.edu/

Как работи Водата

<Автор> Луцио Ботельо
<Автор> Лиляна Фернандес
<Автор> Хорхе Рейс

<Инфо>

<Ключови думи> Вода, обработка на изображения, събиране на данни, микроклимат, роботи

<Дисциплини> Математика, биология, социални науки, роботика, изкуства

<Възраст на учениците> 6–10, 11–15 и 16–18

<Хардуер> <начално ниво> Calliope mini[1], LEGO We Do 2.0[2], малки обучителни роботи[3], WeeeMake[4]

<средно ниво> LEGO EV3[2] с LEGO ултразвуков и сензор за цвят, или Anprino[5] с Arduino[6] и подходящи ултразвукови сензори[7] и сензори за цвят[8]

<ниво за напреднали> компютър с интернет достъп

<Езици> Snap![9], Scratch[10], WeeeCode[4], Open Roberta[11], LEGO Blocks[2]

<Ниво на програмиране> начално, средно, за напреднали

<Резюме>

Този урок е разработен като трансдисциплинарен, т.е. да насърчава съвместната работа между ученици от различни класове от начално до средно училище. Като алтернативен вариант всяка отделна част може да се преподава на съответното ниво поотделно. Започвайки с подход за изчислително мислене, с помощта на програмиране в Scratch[10], през програмиране на роботи и екологичен дом, в урока „Как работи водата“ учениците ще открият всичко свързано с темата за водата.

<Въведение>

Този проект е свързан с водата, нейната роля в нашия живот и нашата роля за опазването ѝ. Разделен на три нива (лесен за ученици в начално училище, среден за ученици в средно училище и за напреднали, за ученици от гимназиален етап), този проект може да бъде адаптиран за съвместна работа на различни училищни нива, а също и в междукултурни дейности.

<Какво правят учениците/учителите>

<Начално ниво: Откъде идва водата?>

Учениците ще бъдат насърчени да проучат откъде идва водата. Учителят ще задава въпроси, за да стимулира интереса на учениците и тогава ... приключението ще започне! Учениците ще проучат, научат и след това ще споделят своите открития със съучениците си. В същото време ще започнат да развиват и умения за изчислително мислене с помощта на лесни предизвикателства, които ще ги научат как да програмират прости роботи.

След като приключат с проучванията си, учениците ще започнат да работят в малки групи. Те ще построят някои прости проекти за начинаещи, използвайки демо режимите на приложението WeDo 2.0[12], като ще се акцентира върху задачите, свързани с водата.

После учениците трябва да изградят едно екологично решение, използвайки конструктор по избор, с което да представят иновативен подход за пестене на вода.

В примера по-долу учениците построиха екологична къща[13] и я комбинираха с някои допълнителни елементи и комплекта WeDo 2.0. След това добавиха резервоар за дъждовна вода, свързан с филтър (програмиран с приложението LEGO), който насочва водата към фермата, така че животните да могат да пият прясна вода (📷 1).

📷 1: Екологична къща

В същото време учениците, все още работейки в малки групи, ще започнат да планират и проектират свои постелки (изрисувани полета за роботите виж. 📷2), свързани с водата, за малки обучителни роботи, които биха могли да бъдат програмирани без компютър. Представяйки работата си пред другите ученици, те ще ги мотивират да програмират и едновременно с това да учат за свойствата на водата. Учениците могат да използват различни евтини обучителни роботи за изпълнението на задачата.[3]

Пълните инструкции за отпечатване на постелките са достъпни онлайн.[14]

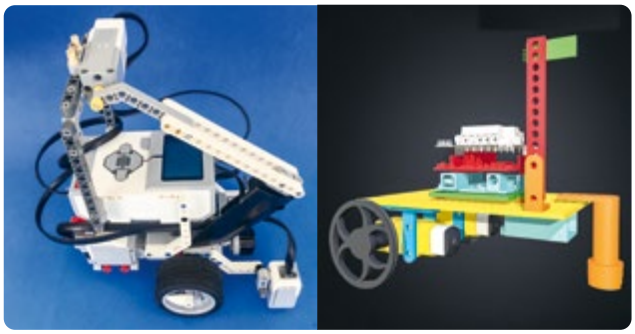
📷 2: Учениците проектират постелка

В допълнителния онлайн материал ще намерите и връзка към план на урока за изпълнение "стъпка по стъпка".[14]

<Средно ниво: Изграждане на робот за пречистване на водата в язовирите>

Проектът за почистване на язовири е свързан с робот, който се движи във язовира и открива твърди отпадъци.

Този проект има две версии, които използват два различни робота. Версията LEGO (📷3) използва обучителен комплект EV3 LEGO. Версията Anprino robot[5], това е робот, който се отпечатва с помощта на 3D принтер и след това се сглобява, работи на базата на Arduino (📷4). Arduino microcontroller[6] и неговата гама от аксесоари могат да се намерят в Anprino.



📷 3: Вариантите с LEGO 📷4: Вариант с 3D Anprino

Започнете с изграждането на модел на резервоара за вода, като използвате хартия или картон. Резервоара трябва да е около 2 м. х 1 м. и да бъде оцветен в синьо, за да симулира вода. Изградете бреговете, като използвате здрав картон, за да ограничите пространството, в което роботите да се движат, направете отпадъците в язовира с парчета черен картон.

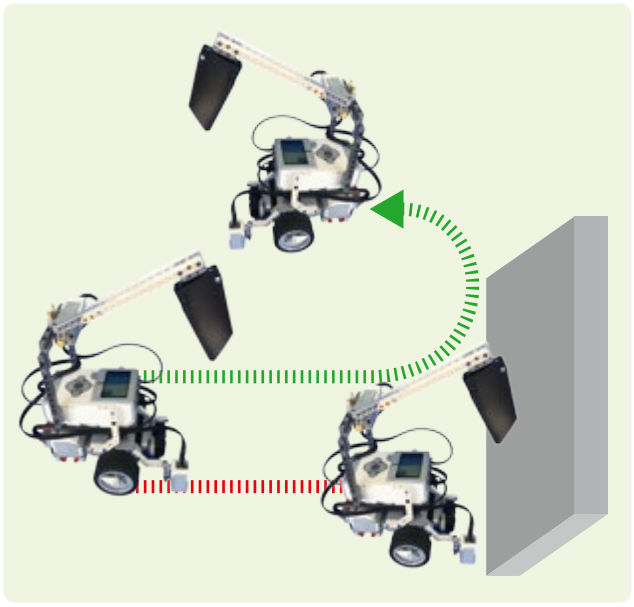
📷 5: Модел на резервоар на язовир с вода

Ултразвуков сензор

Ултразвуковият сензор[7] излъчва звукови вълни, с помощта на които открива обекти и измерва разстоянието до тях. Той може също да изпраща звукови вълни и да функционира като сонар или да получава звукова вълна, която да стартира програма на робота.

Използвайки ултразвуковия сензор, роботът може да открива препятствия и да реагира по различни начини, в зависимост от кода. Роботът може да бъде програмиран да спре или да промени посоката си на движение. При модела на язовира препятствията са картонените брегови линии.

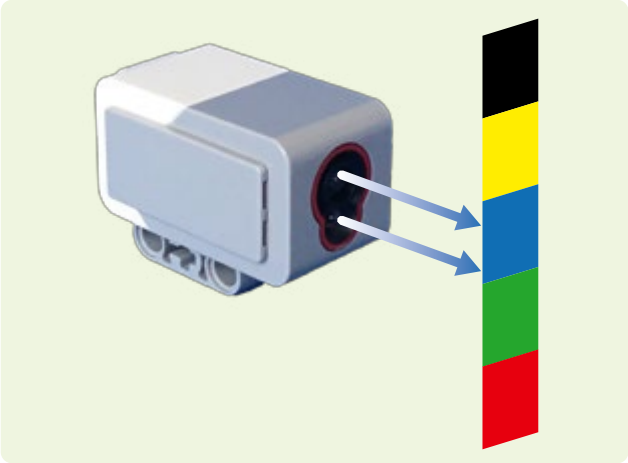
📷 6: LEGO ултразвуков сензор



📷 7: Роботът спира / Роботът променя посоката си

Сензор за разпознаване на цвят

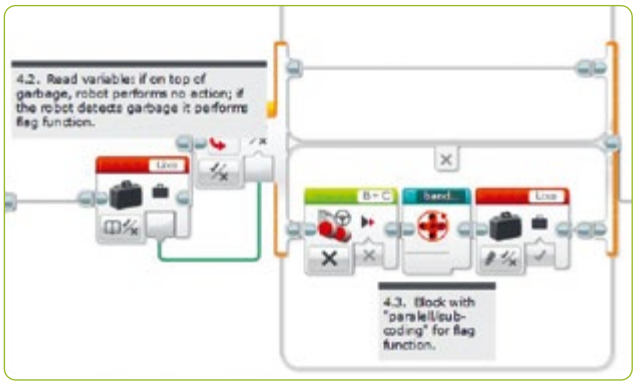
Сензорът за цвят[8] може да открие различни цветове, както и липсата на светлина. Той работи и като сензор за светлина, като разпознава различен интензитет на светлината. Учениците могат да построят различни цветни линии, които роботът да следва.



📷 8: Модел на сензор за разпознаване на цвят: Той различава цветовете, като чете техния RGB код.

Да програмираме, използвайки LEGO програмни блокове

Учениците трябва да изработят модели на различни видове отпадъци, попадащи във водните басейни – битови, промишлени, свързани с туризма, органични и др. Основната цел е те да се запознаят със замърсяването на реките и язовирите. Учениците трябва да симулират детектор за отпадъци, прикачен към лодка, а след това да планират и построят лодката за събиране на отпадъци.



📷 9: Извадка от програмирането на LEGO; пълната схема е достъпна онлайн[14]

Роботът трябва да възпроизвежда определен звук за всеки вид отпадъци, който открие. За да се постигне това, трябва да използваме сензор за цвят и специфични цветни петна, с които да се маркира всеки вид отпадък.

Учениците могат да търсят информация в публикациите на екологични/ правителствени организации и да направят различните видове замърсявания според статистиката.

Учениците трябва да отидат в близост до реки и язовири, за да проучат качеството на водата и видовете замърсители. След това да приложат тези наблюдения в моделите, които създават. С помощта на робота трябва да сканират и отбелязват получените резултати в таблица (📷10).

Когато учениците съберат достатъчно резултати, те трябва да представят своите изследвания пред класа. Целта е те да развият своите умения за критично мислене, изследване и програмиране. Когато учениците погледнат какво се случва във водните ни басейни, те ще видят последиците от вековната слепота на човечеството към екологичните проблеми. За да могат да разберат това, те трябва да са придобили необходимите екологични практически умения, за да могат да изразят позиция в своята общност. Това може да бъде например чрез предупреждение към хората за необходимостта от промяна в поведението, с което се уврежда околната среда и в частност водата. Освен това те трябва да могат да планират и взимат решения, когато открият проблем. Общата цел е да се увеличи гражданското им участие и чувството за екологична отговорност в тяхната общност.

Моля, обърнете внимание: нашите ученици трябва вече да са изградили и тествали LEGO версията, но все още са в процес на подобряване на версията на Arduino. Пълният код, използван за програмиране на Arduino, е достъпен онлайн.[14]

📷 10: Таблица за идентифициране на отпадъци, данни, събрани от две различни екскурзии и отпадъци, събрани от всеки един от екипите за почистване от училищния клуб по околна среда.

	Вид отпадък					
Дата	Битов	Индустриален	Недифе-ренциран	Органичен	Друг вид	Почистена зона
Екскурзия Април 2018	3,450 кг			32 кг	8 кг	100 м²
Екскурзия Май 2018	0,730 кг			6 кг		100 м²

<Ниво за напреднали: Програмиране на образователни игри, свързани с околната среда>

Основната цел е учениците да се запознаят със замърсяването на водата. Те ще използват Scratch[10] за програмиране на игри, които мотивират другите да помагат за опазването на водата и по този начин насърчават хората да не изхвърлят отпадъци във водните басейни.

Първата ни игра симулира малка риба в океана. Тя трябва да се храни, като в същото време избягва други морски същества (акули и раци) и падащи отпадъци (чаши, консерви и др.). Колкото повече яде, толкова по-голяма става и толкова повече точки печели играчът.

Рибата не трябва да се сблъсква с отпадъци и други риби, в противен случай се наранява и получава превръзка. Когато превръзките станат три, играта свършва. Тази игра е забавна и повишава вниманието не само на децата, но и на възрастните към нарастващото количество отпадъци в нашите водни басейни (океани, реки и т.н.).

Втората игра е базирана на добре позната видео игра, в която жаба трябва да пресече улица. Но в нашия случай героят трябва да премине река (използвайки трупите, тъй като водата тече бързо) и да избягва боклука, както и други животни (прилепи и змии). Освен това може да яде мухи, за да спечели допълнителни точки.

В тази игра има четири различни сценария, като произволно в началото се избира един. Жабата (Спрайт) има три живота, след което играта приключва.

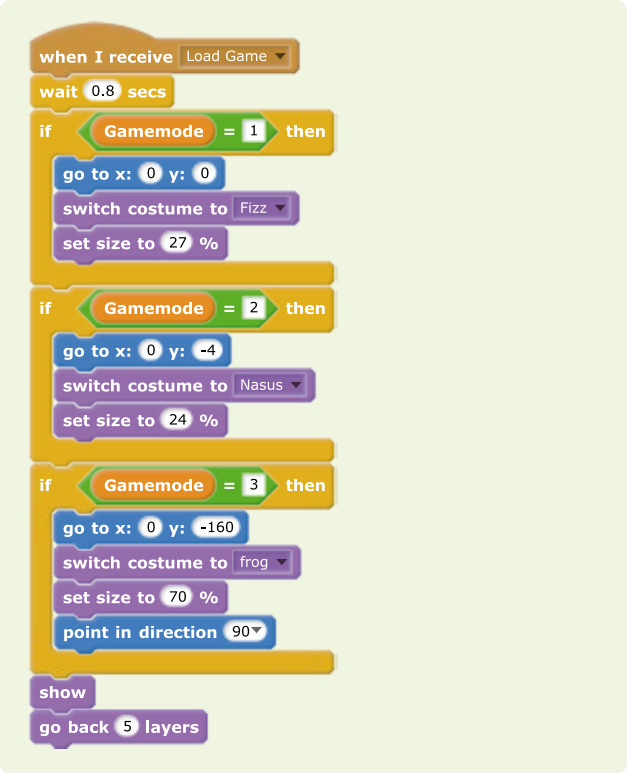
Следващият раздел съдържа подробности за програмата.

📷11 показва частта от програмата, която контролира движението на някои от враговете в различните режими на игра. (*Забележка – 📷11 присъства в оригиналния текст на аглийски, но тук не се побра. Пълният код може да вземете от [14]) В показания пример врагът изчезва, когато докосне някои от краищата. Докато не докосва край, той повтаря същото движение, като увеличава скоростта с корекция от 0,04 след увеличаването на резултата на играча. Това е много умен начин да направите играта малко по-предизвикателна, тъй като резултатът се увеличава с нарастващото ниво на трудност.

Начало на играта: Изберете един от трите режима на игра (📷12). Към момента са готови две игри и учениците разработват трета, наречена Game mode 2.

Например, Game mode 2 може да бъде в езерце, където патиците трябва да хванат някаква храна.

Патиците редовно ядат малки риби и рибни яйца, охлюви, червеи, мекотели и малки ракообразни като раци, трева, листа, плевели, водорасли, водни растения, корени, малки жаби, саламандри и други земноводни. Освен това патиците трябва да се опитват да избягват други патици или отпадъци в езерото (или в напреднали нива, случайни бракониери).



📷 12: Програма Scratch за начало на играта

Ако рибата докосне някой от враговете (1, 2 или 3), тя губи един живот и се чува звук.

Ако играчът загуби всичките си животи, играта свършва, т.е. всички скриптове/сценарии са спрени (📷13).

📷 13: Програма Scratch за врага 1–3

Играта е много добре програмирана и конструирана, тъй като за двете версии на играта се използва един и

същ код. Може да променяме вида на всеки герой, например от „Акула“ на „Прилеп“.

Всички ученици ще се научат да подобряват тази програма, като дават нови идеи или помагат с иновативни решения за програмиране, така че кодът да бъде още по-добър и по-гладък.

Това е възможно, тъй като играта има подобни цели:
- ↳ да избегнеш враговете;
- ↳ да хванеш храна/мухи;
- ↳ когато изгубиш 3 живота, играта свършва;
- ↳ печелиш точки (когато рибата яде храна за риби, когато жабата яде мухи и се достига ново ниво).

Те ще използват клонинги на враговете, така че да може един и същ герой да се появява от различни посоки и да има различно поведение в играта.

Пълната програма е достъпна за изтегляне.[14]

<Заключение>

В този урок учениците ще работят съвместно с връстниците си и ще учат и споделят своите знания за водата: кръговрат на водата, недостиг на вода, замърсяване и др. Също така ще разработят ресурси за наблюдение, спестяване и опазване на водата. В същото време учениците ще разработят инструменти за изследване и програмиране и развият умения в областта на роботиката. По-големите ученици ще бъдат ментори и ще подкрепят по-малките, всички те ще се мотивират и предизвикват взаимно да напредват в работата си. Това ще допринесе изключително много за успеха на проектите.

В края на тази учебна година забелязахме, че учениците не само са подобрили уменията си по програмиране, но и са били по-запознати с проблемите на водата и опасностите за животните и растенията, които зависят от чистата вода, за да живеят безопасно.

Не е лесно да програмирате няколко игри в една. Игрите трябва да са сходни, за да може кодът от една игра да бъде адаптиран и приложен към всички версии. Това е умен начин да спестите от програмен код.

Избрахме да работим заедно, макар и в различни училища, които са далеч едно от друго, защото това ни позволи да споделяме идеи и да подобрим съвместната работа между ученици от различен (социален и икономически) произход и възраст. Не беше лесно да се срещнем лице в лице или да съберем учениците заедно, както възнамеряваме, но се оказа добър вариант, тъй като позволяваше на учениците да споделят своите идеи и методики и да общуват с непознати връстници, което подобри и техните комуникативни умения. Те имаха възможност да участват в различни състезания, да обсъдят резултатите си и да предложат съвети за подобряване на работата на другите ученици. Алтернатива на личните срещи може да бъде общуването онлайн, чрез видеоконферентни връзки. Накрая, учениците имаха възможност да споделят работата си с общността и да играят роля в промяната на отношението на общността към опазването на водата.

С този урок можете да стимулирате учениците да разработят други идеи и концепции по темата за опазване на водата и да допринесат за подобряване на екологичното си поведение и по този начин да повлияят на общността за намаляване на въглеродния отпечатък.

<Сътрудничество>

Наука на сцената е платформа за обмяна на ресурси между учителите!

В резултат на този проект се създаде и развиви общност от учители и бяха споделени ресурси и идеи. Това допринесе за по-доброто обучение на ученици в цяла Европа. Споделянето и сътрудничеството е най-добрият начин за всички нас да подобрим и доразвием тези проекти.

<Препратки>

[1] https://calliope.cc/en
[2] https://education.lego.com
[3] Прости роботи за използване в начално ниво: Bee Bots от tts, DOC от Clementoni, Jack от Imaginarium
[4] www.weeemake.com/
[5] Anprino е робот, разработен от Португалската национална асоциация на учителите по информационни технологии (ANPRI); информация и файлове за 3D принтиране http://www.anpri.pt/anprino/index.php/anprino-luis/ (29/11/2018)
[6] www.arduino.cc/
[7] Ние използвахме HC-SR04 ултразвуков сензор.
[8] Ние използвахме BE15000624 сензор за цвят.
[9] https://snap.berkeley.edu/
[10] https://scratch.mit.edu/
[11] https://lab.open-roberta.org/
[12] https://education.lego.com/en-us/downloads/wedo-2/software (29/11/2018)
[13] LEGO SET 31068
[14] Всички допълнителни материали са достъпни на www.science-on-stage.de/coding-materials.

Спасителен план за растенията през ваканцията (СПРВ)

<Автор> Андреас Майер
<Автор> Соня ван дер Бил

<Инфо>

<Ключови думи> автоматизирано поливане на растения с помощта на микроконтролер за управление на водна помпа

<Дисциплини> компютърни науки, естествени науки, технологии

<Възраст на учениците> 10–14

<Хардуер> компютър (по един за всеки ученик, ако е възможно), Calliope mini[1] със сензор за влажност и температура (един за групата)

<Езици> Scratch[2] (online или offline), редактор за Calliope[3] (online)

<Ниво на програмиране> начално

<Резюме>

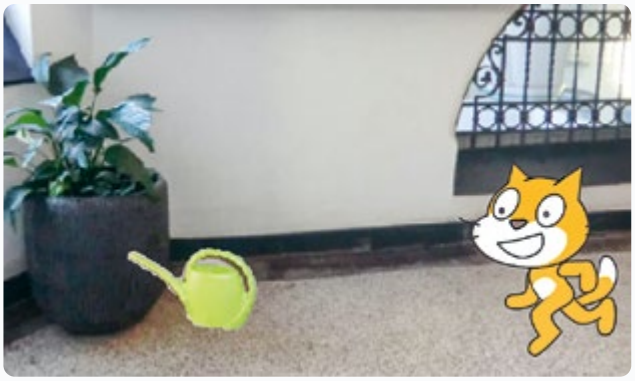
Саксийните растения в училищните сгради често умират по време на лятната ваканция, защото никой не се грижи за тях. За да решим този проблем, имаме нужда от спасителен план за растенията през ваканцията. В тази част ще разработим виртуален и реален проект за спасение на училищните растения.

<Въведение>

Проектът е подходящ за всички STEM дисциплини, тъй като изискваното ниво за програмиране е начално. В първата част, поливането на растенията през равни интервали, се реализира във виртуални условия. Това ще обхване различни контролни структури (условни конструкции) в компютърното моделиране, като обектно ориентиране, цикли и условия, както и използването на променливи. Учениците ще направят своята първа лесна програма, като ще използват разработен онлайн урок в ‚Първи стъпки със Scratch'[4]" (продължителност: 45 минути).

Обектно-ориентираното програмиране (ООП) е вид програмиране, което дава на обектите свойства (атрибути) и възможности (методи). В нашия случай тези обекти са котка, наречена „Спрайт", лейка и сцена. Всеки обект принадлежи към съответен клас. Всички обекти от един и същи клас имат еднакви свойства и възможности. Можете да тълкувате класа като проект, а обекта като представител, т.е. конкретният изпълнител на проекта. В Scratch фигурите са представители на клас „Спрайт" или накратко „спрайтове". Котката, наречена „Спрайт", е екземпляр от класа „Спрайт".

Един от атрибутите на спрайт е неговият костюм; в този случай това е образът на котка. Друг екземпляр от класа „Спрайт" може да бъде изобразен като човек и да бъде наречен Гюнтер. Котката „Спрайт" и човекът „Гюнтер" са и обекти (примери) от класа „Спрайт" или накратко: и двата са спрайтове. В Scratch има само два класа: сцена и спрайтовете. В този проект имаме два спрайта (котката и лейката) и сцената (📷1).



📷 1: Виртуален план за спасяване на живота на растенията през ваканцията

Към втората част на проекта може да се пристъпи след завършване на първата част, а може да се изпълни и самостоятелно, ако учениците познават споменатите по-горе структури за управление и вече са придобили начален опит в програмирането с Calliope mini[1]. Вместо променливи величини ще се използват сензори за микроконтролер, които ще управляват клапана на водната помпа.

<Какво правят учениците/учителите>

Всички необходими материали и работни листове са на разположение за изтегляне.[5]

<Част 1: Виртуално редовно поливане на растенията>

Стъпка 1: Програмиране на програма само с една променлива, време; запознаване цикли с едно условие (продължителност: 180 минути.)

След анализиране на проблема (Как може виртуално да се спаси животът на растенията през ваканцията?), учениците ще бъдат помолени да помислят за основната структура на такава програма. Те трябва да си водят бележки под формата на рецепта (алгоритъм) и да тестват идеите си взаимно. След това те ще трябва да постигнат съгласие за общата структура на програмата (виж Работен лист 1[5]).

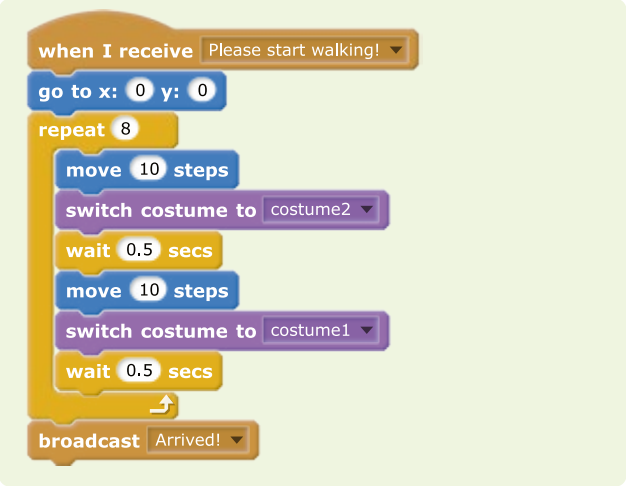
Този етап ще помогне на учениците да мислят за основната структура на програмата, която искат да програ-

мират. Ключовите думи „списък с инструкции (команди)“, „цикъл“ и „условие“ са извлечени от контекста.

Сега учениците ще реализират програмата в Scratch[2], като сглобят отделните компоненти[5], обединени в работна програма. Те ще научат повече за Scratch в процеса на работа и по-специално следното:

- ориентация на обекта (всяка фигура има собствен скрипт, дори и сцената);
- структура (Как изглежда структурата блоковете за условие или цикъл в Scratch?);
- сценарий / костюм / и звуци могат да бъдат зададени на всяка отделна фигура.

Освен това, учениците ще научат повече за основната структура на програмата за поливане на растения, докато обмислят как най-добре да подредят отделните части от кода, за да може програмата да работи. Особено важно е да решите кои условия трябва да бъдат вътре в цикъла за повторение (📷 2 и 3); това може да стане чрез опит и грешка.



📷 2

📷 3

Въз основа на програмата, дадена в предходната част, учениците вече ще могат да създадат своя собствена програма, в която котката „Спрайт“ се придвижва към растенията и ги полива, според променливата – време. Единственият файл, който те ще получат, е със стартовата сцена.[5]

Учениците ще бъдат насърчавани да изследват езика за програмиране самостоятелно, за да изпробват собствените си идеи и да бъдат креативни. Важно е да могат да използват необходимия език за програмиране с увереност (в съответствие с нивото на знания); по този начин ще им бъде по-приятно.

Стъпка 2: Напишете програма, която включва променливите „ниво на водата“ и „температура“ (необходимо време около 270 минути).

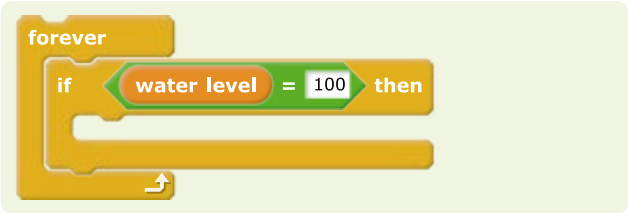
Като въведение към втората стъпка, учениците трябва да помислят за други фактори, които определят колко често трябва да се полива едно стайно растение. „Стайната температура“ и „нивото на водата“ в контейнера за растения със сигурност ще играят роля тук, като променливи величини.(виж Работен лист 2[5]).

Учениците ще получат работна програма, в която програмирането на котката „Спрайт“ и напояването могат да бъдат почти същите, както преди.

Въпреки това ще има нов код за етапа, който контролира променливата „ниво на водата“ на мястото на предишния таймер. Котката ще полива растението само веднъж. Учениците ще бъдат насърчавани да мислят за това, как променливата „ниво на водата“ може да бъде определена и как тя да контролира дейността на котката. От друга страна, те ще имат задачата да разрешат проблема, така че растението да се полива само веднъж. Има наличен помощен файл, ако е необходимо.[5]

Използвайки само една променлива, програмата ще бъде ясно структурирана. Ще е твърде предизвикателно за начинаещ програмист да координира няколко променливи наведнъж.

Използваната структура на цикъла е по-сложна от преди, тъй като е вързана към състояние (ниво на водата) (📷4). Учениците ще трябва да помислят внимателно какво трябва да се повтаря, колко често и при какви условия.

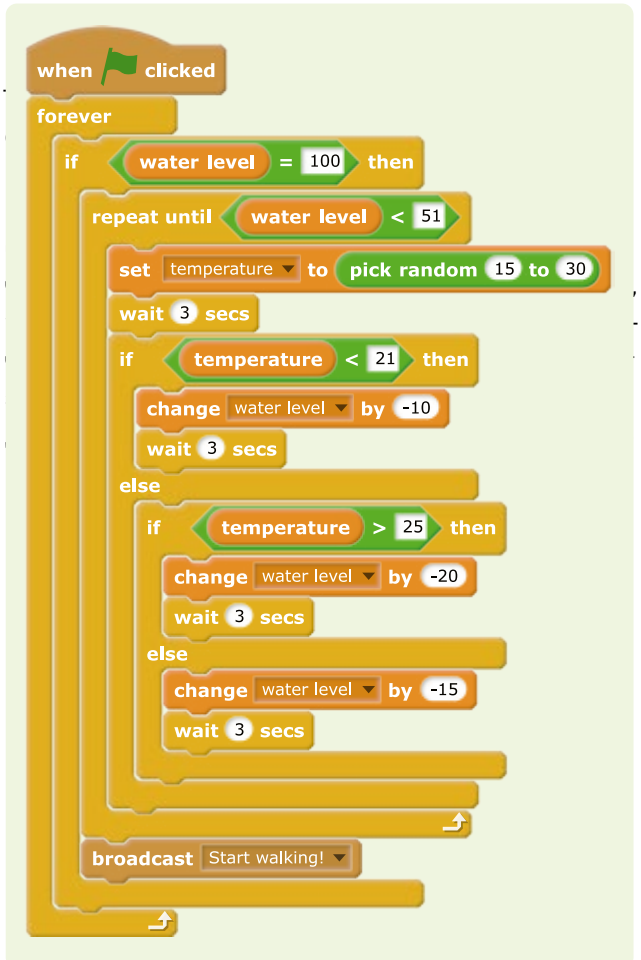


4

Способността за правилно подреждане на програмата (структуриране) е от съществено значение при изучаването на всеки програмен език, но решавайки дадената задача ще се учи по много по-забавен начин; учениците ще могат да изпробват всичко без негативни последици.

По-бързо възприемащите ще имат възможност и да променят програмата и да изпробват собствените си идеи в края на този работен етап.

В следващата стъпка променливата „температура“ ще играе роля в програмата. Стойността ѝ се определя от генератор на произволни числа, който подава число между 15 °C и 30 °C. Нивото на водата в контейнера за растенията се променя в зависимост от тази стойност



5

Както и в предишната стъпка, на учениците ще бъде предоставена възможност да персонализират получената програма, като работят според техните идеи и възможности за програмиране.

В края на първата част на проекта, където програмата за напояване на растенията през ваканцията е реализирана във виртуални условия, ще се представят много и разнообразни резултати, за да се проверят идеите на учениците и да се докаже тяхната ефективност.

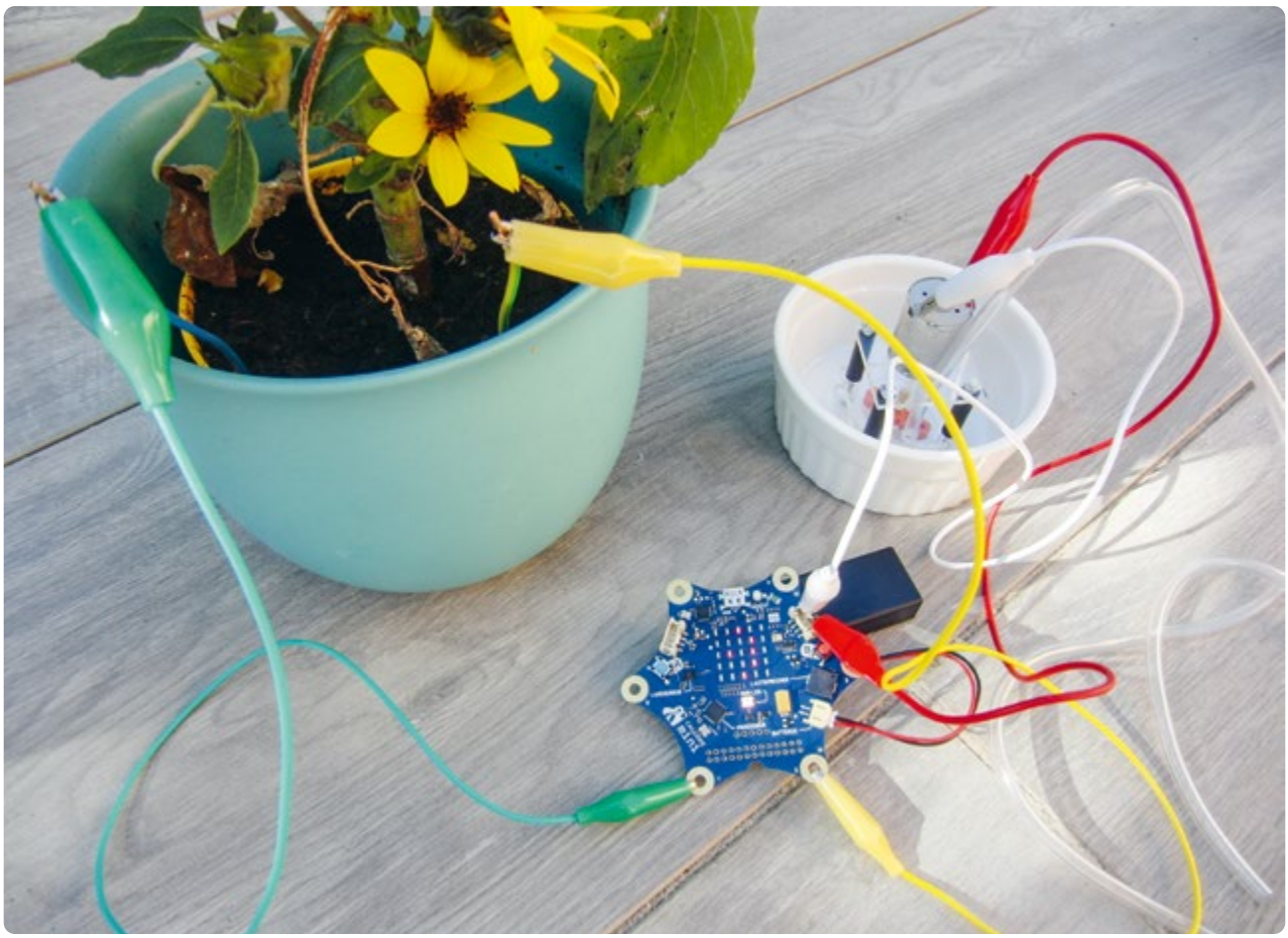
<Част 2: Редовно поливане на растение, контролирано от микроконтролер>

По време на проекта бързо стана ясно, че някои ученици не са доволни от виртуалното решение на проблема с поливането. Те се интересуваха от начините за поливане на истински растения с помощта на тяхната програма. Програмата за виртуално поливане е относително лесно да се прехвърли в реални условия с помощта на микроконтролер, особено след като много от тези мини-компютри също могат да бъдат програмирани с Scratch[2] или подобно приложение. В нашия проект използваме Calliope mini[1], микроконтролер подобен на BBC micro: bit[6], който съдържа и удобни за потребителя „plug-and-play функции“ като сензори и връзки. Въпреки това СПРВ може да се контролира и с всички други микроконтролери, които често се използват в училищата, като LEGO EV3, LEGO NXT, Arduino, Raspberry Pi, Teensy и др.[5]. Програмирането на Calliope mini е лесно, тъй като трябва само да го свържете към компютъра чрез USB кабел. Open Roberta lab[7], е онлайн среда за програмиране, която поддържа различни микроконтролери (налични са алтернативни редактори[5]), с подходящ интерфейс за програмиране. Интерфейсът на програмиране е достъпен на различни езици и можете да го смените, като кликнете върху иконата на глобус, след като сте избрали вида микроконтролер, в нашия случай на Calliope mini.

Една проста версия на СПРВ може да работи както следва:

1. Влажността на почвата се измерва постоянно.
2. Ако почвата е твърде суха, се изпомпва определено количество вода, докато почвата се овлажни достатъчно.

Следователно микроконтролерът трябва да може да измерва влагата на почвата и да контролира двигателя на водната помпа.



6: СПРВ контролиран от Calliope mini

Calliope mini има четири сензора за докосване. Те измерват електрическата проводимост между точките на свързване. Тъй като водата провежда електричество, влажната почва има по-висока проводимост от сухата почва. Можете да използвате два медни проводника, като сензори, които поставяте в саксията за цветя на известно разстояние. След това те се свързват с щипки тип крокодил към мини Calliope с контактите в ъглите (-, P1). Изходната стойност на сензора P1 (аналогов пин) ще бъде между 0 и 1023. Ако проводимостта намалее под определена стойност, помпата, която полива растението, ще се активира (вижте 6).

Основен компонент на помпата е малък електродвигател, който се свързва към Calliope mini директно или с помощта на допълнителен драйвер на двигателя, в зависимост от необходимото ниво на мощност. В менюто за действие / движение са налични два порта за двигатели (A, B). Количеството на водата в помпата може да се регулира чрез промяна на стойността „скорост в %“.

Разбира се нужно е определено ниво на умения за конструиране на помпата и за изграждането на двигателя.[5] Помпата струва само няколко евро.

Докато работят с истинския спасителен план за растения през ваканцията, учениците ще предложат множество идеи как да го оптимизират, което ще задълбочи знанията им в процеса на работа. Например:

- Нашите растения нуждаят ли се от много вода или само от малко? Колко голям трябва да е резервоарът?
- Кое е най-доброто време за поливане на растенията? По-добре ли е да поливате растенията много веднъж на ден или пък по-малко, но няколко пъти на ден?
- На каква дълбочина трябва да се поставят сензорите за влажност в почвата, за да се осигурят оптимални измервания? Какво е идеалното разстояние между сензорите?
- Колко дълго трае захранването на СПРВ? Може ли да се повиши енергийната ефективност на помпата, така че напояването да е възможно за цялата ваканция в училище?

Както можете да видите, проектът СПРВ предлага на учениците различни подходи от областта на биологията и физиката за бъдещи експерименти или работа по

проекти. Друг интересен вариант би бил да свържете СПРВ към Интернет и да наблюдавате онлайн (Internet of Things). Въпреки, че това би надхвърлило нашето въведение в програмирането с СПРВ, то показва какви резултати могат да се постигнат, като задаваме обикновени въпроси.

<Трансфериране към други програмни езици>

Проектът може лесно да бъде прехвърлен на Snap![8], което е по-нататъшно развитие на Scratch[2]. Примерите за програмиране са предоставени онлайн[5]. Тези два езика за програмиране са особено подходящи за проект, който е насочен към начинаещи програмисти, тъй като те са лесни за разбиране и насърчават учениците да изпробват идеи, като използват „плъзгане и пускане“.

<Заключение>

Ученици, които са нови в програмирането, ще направят първите си стъпки в тази област и ще научат важни основи на език за програмиране при изпълнение на този проект, който представя реална ситуация. Фокусът не е върху изучаването на синтаксиса на програмния език, а върху изпробването на ефекта на различни контролни структури: „Какво работи и защо?“.

Грешките са добре дошли, защото обикновено са лесни за намиране и обяснение и по този начин помагат на учениците да разберат как работи езикът за програмиране.

От една страна, дадената рамка дава на учениците сигурност (който не е сигурен, ще реши само пъзела за сглобяване на предоставените части на програмата), а от друга страна, има достатъчно място за творчество за онези ученици, които изпълняват „задължителните задачи" бързо.

Някои идеи се появиха в резултат от проекта, напр. да се изгради „истински“ робот за поливане или да се разработи игра за поливане. Във всеки случай учениците придобиха положителен начален опит в програмирането, който се надяваме да има траен и устойчив ефект върху тях.

Времевата рамка, определена в нашето училище, понякога беше трудна. Имахме само 45 минути на урок да работим по този проект. Самата организационна част на уроците (влизане в компютъра, отваряне на файлове, запазване на файлове, излизане от компютъра) отнемаше 15 минути, така че нямаше достатъчно време за реално програмиране и работа. Можете да поискате достъп в Scratch, като учител[2], което ще ви позволи да създадете клас и да споделяте материали.

<Сътрудничество>

Тъй като СПРВ е въведение в програмирането, има много малко възможности за сътрудничество.

Веднага след като проектът се приложи в реалния живот с помощта на микроконтролерите, сътрудничеството между учениците става много полезно, тъй като степента на трудност на проблема се увеличава при използването на допълнителни компоненти (сензори, помпа). Например, по-големите ученици могат да помогнат за изграждането на помпата. Училища в различни страни могат да работят заедно по проекта СПРВ и да сравняват своите резултати и решения.

Може да се създаде обща база данни за различните растения, за да се адаптира СПРВ към индивидуалните характеристики на различни видове растения. Ако СПРВ е свързан с интернет, училищата могат да приемат растения от друго училище и да поемат отговорността за поливането му.

<Препратки>

[1] www.calliope.cc/en
[2] www.scratch.mit.edu/
[3] www.calliope.cc/en/los-geht-s/editor
[4] https://scratch.mit.edu/projects/editor/?tip_bar=getStarted (29/11/2018)
[5] всички допълнителни материали са достъпни на www.science-on-stage.de/coding-materials.
[6] www.microbit.co.uk/home
[7] https://lab.open-roberta.org
[8] https://snap.berkeley.edu/

Магическа ръкавица

<Автор> Аннамария Лукаш
<Автор> Камелия Йоана Рациу

<Инфо>

<Ключови думи> експеримент, околна среда, температура, влажност, светлина, магнитно поле

<Дисциплини> физика, химия, биология, екология, компютърни науки

<Възраст на учениците> 10–18
<Ниво 1> начално училище (възраст: 10–11) и основно училище (възраст: 12–15)
<Ниво 2> гимназия (възраст: 15–18)

<Хардуер> Arduino UNO[1], сензори, съвместими с Arduino (напр. сензор за светлина, температурен датчик, сензор за магнитно поле, датчик за влажност, сензор за газ), LCD екранен с бутони, проводници – джъмпери, външна батерия

<Езици> C[2], Arduino 1.8.5[3], Snap![4]

<Програмно ниво> средно

<Резюме>

Младите хора се вълнуват от технологиите, така че урок, който съчетава природните науки с компютърните науки, ще бъде успешен. Учениците ще направят и използват ръкавица с различен сензор на всеки пръст. Това ще им позволи да извършват различни експерименти, като свързват само необходимия сензор.

<Въведение>

Предимството на използване на устройството (ръкавица), оборудвано с няколко сензора за различни измервания, е двойно. От една страна, учениците в основното и средното училище могат да използват „вълшебната“ ръкавица за измерване на температура, яркост, влажност, наличие на магнитно поле, интензитет на звука и т.н. Всичко, което трябва да направят, е да изберат желания сензор и са готови да започнат да търсят приложение на ръкавицата, за да правят изследвания в различни области и училищни предмети.

Ръкавицата може да бъде използвана и навън, тъй като се захранва от батерия. Това осигурява възможност на учениците да провеждат изследвания извън лабораторията. От друга страна, гимназистите могат сами да си направят ръкавица, за да извършват определени измервания. Те имат теоретичните знания за понятията от различните науки (физика, химия, биология, екология) и наистина са въодушевени от възможността да ги изследват в практически експерименти.

Учителите ще трябва да представят основните понятия и правила за създаване на програма в C[2] или на друг език за програмиране, поддържан от Arduino, включително Snap![4], за да могат учениците да програмират Arduino[1]. Ако изберат C, като език за програмиране на Arduino, те могат да гледат видео уроци в Интернет и да се научат да програмират. Това ще им помогне да разберат по-добре как трябва да се напише програмата и също така ще повиши увереността им, тъй като повечето от тях ще бъдат изненадани от това колко лесно нещо е кодът.

<Какво правят учениците/учителите>

<Ниво 1>

Имате готова ръкавица с LCD дисплей и бутони. Учениците слагат ръкавицата и избират желания сензор с бутоните **UP / DOWN**; след което натискат бутона **SEL** и започват измерването. На екрана се показва стойността. Учениците могат да повторят измерванията, когато пожелаят. За да се върнат към началния екран, натискат бутона **BACK**. Например на 1а-1в, можете да видите как се определят полюсите на магнита.

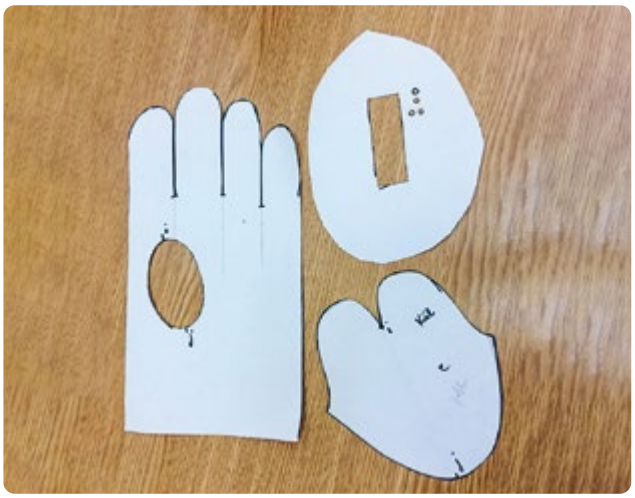
1 а-в Определяне на полюсите на магнита

<Ниво 2>

Учениците в клас могат да бъдат разделени в четири групи. Едната група крои и шие ръкавицата, втората група свързва веригата, третата група програмира, а последната група калибрира сензорите.

<Да направим ръкавица>

Учениците изработват шаблон (📷2), след като разглеждат документацията от няколко уебсайта[5]. Те сгъват материала (в нашия случай кожа, но могат да се използват и други материали) на три и го изрязват с помощта на шаблона. За да получат подходящата ръкавица, учениците зашиват двете части, като ги поставят лице с лице. После изрязват отвора за LCD дисплея и бутоните и зашиват последното парче от плата/кожата, чак след като монтират Arduino с LCD и сензорите на горната страна на ръкавицата.



📷 2: Шаблони за ръкавица

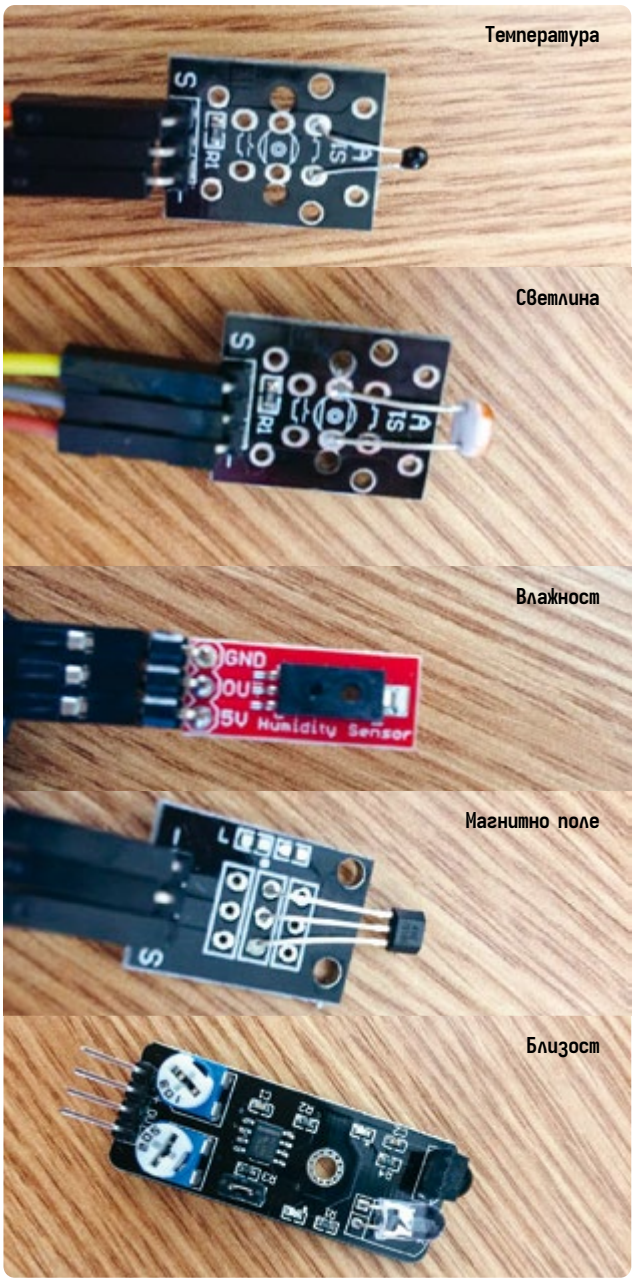
<Изграждане на веригата>

Учениците свързват веригата по схема, която предварително са изработили, обсъдили и анализирали с учителя. Веригата може да бъде фиксирана (застопорена) или не. Тя трябва да отчита правилното свързване на сензорите към платката Arduino, а именно **GND** от сензора към **GND** на платката Arduino, **VCC** от сензора до **5V** на Arduino и OUT на сензора към един от **ANALOG IN** (A0, A1, A2, A3, A4 или A5) на платката. Ако трябва да се свърже датчик към **DIGITAL IN**, необходимо е да се внимава да не се използва някой от входовете, използвани за LCD, защото ще възникнат операционни грешки. В нашия пример свързахме следните сензори: температура (A1), светлина (A2), влажност (A3), магнетизъм (A4) и близост (A5) (виж 📷3а–3е).[6]

<Писане на кода за веригата>

Учениците в гимназията, които изучават езика за програмиране C[2], могат лесно да програмират Arduino[1]. Има много уроци, достъпни онлайн на много езици. Например нашите ученици използваха уебсайт на румънски език[7]. Ръководства на английски език се предлагат на уебсайта на Arduino или на уебсайтовете на дистрибуторите[8], наред с ръководства, преведени и на други езици. Има и много сайтове, където учениците могат да намерят уроци.

Учителят може да ги напътства как да напишат програмата за Arduino. Може да намерите пълния код, който използвахме – онлайн.[9]



📷 3а–е: Сензори

<Калибриране на сензорите>

Има налични калибрирани сензори, но има и некалибрирани. Понякога е по-приятно учениците да намерят начин да ги калибрират. Те намериха формули за калибриране на някои от сензорите в Интернет. Например има формула за калибриране на сензора за влажност[6], тъй като показваните стойности варират и функцията не е линейна.

По отношение на калибрирането на температурния сензор, учениците проследиха стойностите, показани от сензора. Използваха калибриран термометър в лабораторията и сравняваха показаната стойност със стойността на термометъра. Те откриха, че стойностите на този сензор се променят линейно и намериха формулата за калибриране. В допълнително предоставения материал[9] има примери с формули за калибриране на датчиците за влажност и температура.

След като учениците калибрират сензорите, завършат програмата и проверят свързването на дисплея, за да се уверят, че сензорите в пръстите на ръкавицата съответстват на данните, появяващи се на екрана, дисплеят се монтира окончателно на ръкавицата. Последната стъпка е да зашиете външния слой на материала. Учениците използваха няколко закрепващи пръстена за всеки сензор на всеки пръст (📷4), за да фиксират по-добре сензорите.

📷 4: Фиксиране на сензорите

<Алгоритъм за използване на други езици>

Ако искате да използвате друг програмен език, схема, с всички необходими елементи на основната програма, е достъпна онлайн.[9]

<Заключение>

Учениците се радват да откриват нови неща и са много изобретателни. Те обичат да правят експерименти, а ръкавицата изглежда така, сякаш идва от научнофантастичен филм. Учениците от гимназията ще се забавляват с програмирането на елементите и ще видят резултатите веднага в нещо практично, което действително работи. Това преживяване имаше невероятен успех и учителите и учениците научиха много неща заедно.

Те се сблъскаха с една трудност. Не е лесно да намерят правилните сензори[6] и да ги калибрират, но има решения. Ако не може да се намери формула за калибриране, решението се крие в закупуването на калибрирани сензори, въпреки че те са по-скъпи. Тази ръкавица може да се програмира и с Calliope mini, което ще я направи по-лека и по-малка. Нашите ученици имаха желание да направят такава ръкавица, дори и с различен език за програмиране.

<Сътрудничество>

Ученици от различни училища и страни могат да направят такива ръкавици, използвайки различни микроконтролери и подходящи сензори, и след това да обсъдят и сравнят резултатите. Учителят по изобразително изкуство може да бъде помолен да допринесе за дизайна на ръкавицата. Освен това може да се организира конкурс между училищата, в който учениците сами да предлагат различни модели. Ръкавицата е лесна за изпращане по пощата, така че учениците в различни училища могат да експериментират с ръкавиците, направени от техни връстници в други училища и други страни.

<Препратки>

[1] www.arduino.cc

[2] https://en.wikipedia.org/wiki/C_(programming_language)

[3] www.arduino.cc/en/Main/Software

[4] https://snap.berkeley.edu/

[5] Информация и уроци как се шият ръкавици: http://ofdreamsandseams.blogspot.ro/2012/04/1950s-hand-sewn-leather-gloves.html, www.instructables.com/id/How-to-Make-Gloves/

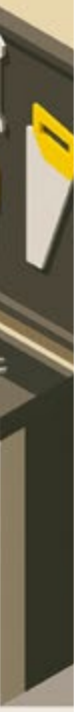
[6] Използвахме сензорите в kit 37 in 1' for Arduino. Сензорът за влажност е Humidity Sensor HIH-4030, brand: Sparkfun, code: SEN-VRM-09.

[9] Всички допълнителни материали са достъпни на www.science-on-stage.de/coding-materials.

* Пълният списък с линкове може да видите в английската версия.

Наука за Триенето

<Автор> Илия Мествиришвили
<Автор> Дейвид Шапакидзе



<Инфо>

<Ключови думи> сила на триене, спирачен път, анти-блокиращ механизъм на спирачките (ABS), програмиране на приложение, обработка на данни

<Дисциплини> физика, компютърни науки, математика

<Възраст на учениците> 14+

<Хардуер> Arduino[1], серво, двигател, Bluetooth модул, разширителен модул за мотор, фотоврата за прецизно измерване на скорост и ускорение

<Език> Arduino програмна среда[2], AppInventor[3], Snap4Arduino[4], Blockly[5]

<Ниво на програмиране> лесно, средно

<Резюме>

Изследването на факторите, влияещи върху силата на триенето, може да се превърне в интересен и забавен експеримент чрез изграждане на нискотарифен автомобил с проста спирачна система, управляван посредством Bluetooth. Това ще даде възможност на учениците да наблюдават и сравняват реални данни като скорост на автомобила (абсолютна стойност на скоростта) преди да се задействат спирачките, спирачен път и как масата на автомобила и вида на повърхността влияят на силата на триене. След това учениците ще проведат експерименти, за да проучат с висока точност как различните фактори, влияят на спирачния път, за да проверят собствените си хипотези и тези предложени им от учителя.

<Въведение>

Триенето е много важна сила в ежедневието и се преподава във физиката, както на ниво среден курс, така и в гимназията. Въпреки това, традиционните експерименти, свързани с темата за триене са ограничени и не много забавни. Този проект ще превърне проучването на триенето във вълнуващ групов проект, който включва:

1. Изграждане и тунинговане на кола.
2. Програмиране на Arduino[1] микроконтролер за измерване на моментната скорост и спирачния път.
3. Програмиране на мобилен телефон с помощта на AppInventor[3], който да изпраща, получава и показва на дисплей данни в реално време.

<Какво правят учениците/учителите>

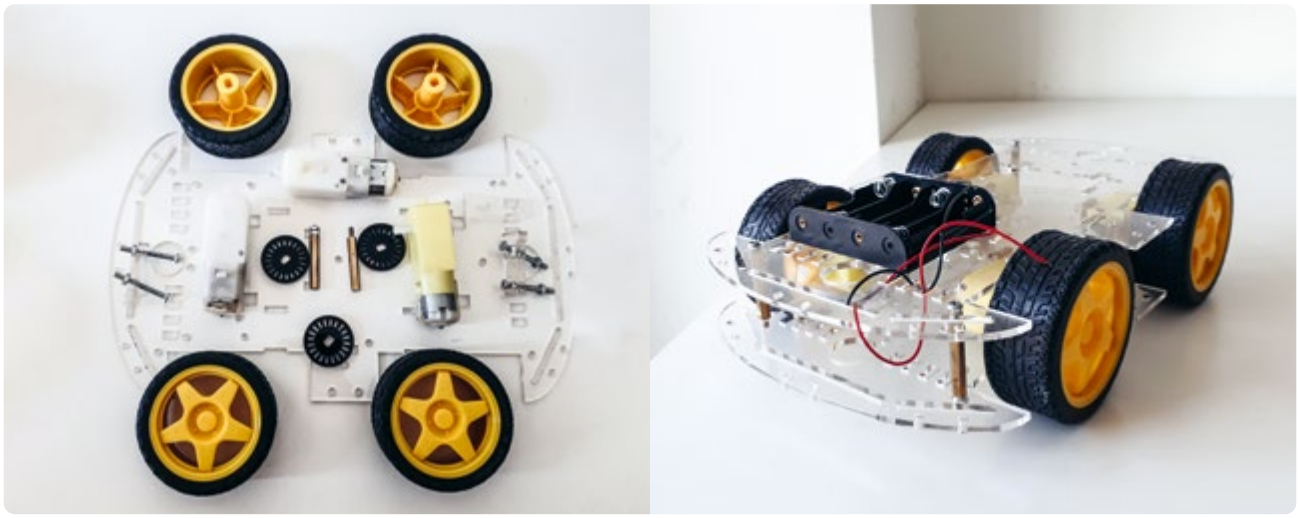
Тъй като тези три задачи могат да бъдат изпълнявани едновременно, препоръчваме на учителя да раздели своя клас на групи от по двама до трима ученици, които в началото ще работят по задачите самостоятелно, но накрая ще се съберат заедно, за да обсъдят и разгледат работата си.

Когато по учебен план се представя темата “Сили на триене”, учителите могат да поднесат теоретичната част от учебното съдържание заедно с този проект, което ще мотивира учениците и ще подобри разбирането на теоретичните понятия. Най-добрият начин да започнете проекта е да зададете следните въпроси към вашите ученици и да им дадете време да ги обмислят и да излязат със собствени идеи, предположения или хипотези:

- Каква е връзката между скоростта на автомобила и спирачния път? (Разбира се, отговорите могат да варират, т.е. "спирачният път е пропорционален на скоростта преди да се натисне спирачката" или някои могат да помнят от физиката, че спирачният път е в действителност пропорционален на квадрата на скоростта);
- Как увеличаването на масата на колата би се отразило на спирачния път, при условие че всичко останало не е променено? (Често срещан отговор е, че увеличаването на масата трябва да увеличи спирачния път.);
- Как трябва да приложим спирачките, за да спрем автомобила възможно най-бързо? (Възможни отговори: най-добрият начин е да спрем колелата напълно, но ако ги накараме да се въртят в обратна посока на движението, това ще спре колата по-бързо; и т.н.);
- Ако предната и задната спирачка са идентични, ще спрат ли колата едновременно? (Учениците могат да се замислят за собствения си опит с велосипедите);
- Всички други въпроси, които могат да възникнат от учителя или учениците.

След като учениците са написали първоначалните си идеи, следващата стъпка ще бъде да помислят как да построят проста кола и как от нея да получават данни, които да проверяват и после да доразвият първоначалните си идеи и предположения. Учителят може да улесни този процес и да предложи на учениците да построят кола, която да събира и изпраща съответните данни на телефон, който от своя страна може да контролира колата, като в същото време получава и показва данните.

В зависимост от техните интереси, умения и предпочитания, на този етап на проекта учениците може да се разделят на групи. Все пак, възможно е и всички ученици да останат в една голяма група и да се справят със задачите поетапно. Следващите стъпки от проекта ще са еднакви за всички независимо кой сценарий изберете.



1: Пълният комплект на шасито

<Изграждане на Шасито и поставяне на електрическите компоненти>

Този подход включва изграждане на автомобил от евтиния и лесно достъпен комплект за шаси Arduino[1], (Robot chassis kit for all ARDUINO systems), изобразен на 1.

Учителите и учениците се насърчават да изпробват различни подходи за проектиране и изпълнение, например различни начини за събиране и изпращане на данни, управление на автомобила от разстояние, както и различни софтуерни и езикови средства, за да се напише необходимият код.

След като сглобят колата и решат къде да монтират Arduino контролера и двигателя, учениците ще трябва да помислят за различни начини за измерване на скоростта на колата. Препоръчителният подход е да се даде възможност на учениците да мислят и предложат някои свои собствени идеи. С подходяща помощ от учителя, най-добрият и лесен начин да направите това е да използвате фотоврата и да преброите колко са оборотите на свободно задно колело. По този начин учениците ще могат да измерят както моментната скорост, така и изминатия път. Това може да се направи с помощта на материалите в комплекта на шасито – Arduino 4 wheel chassis kit или ако учителят предпочита, да се разработи отделно.

Ще е необходимо да използваме някои математически операции, за да се преобразува броят на спиращите събития, преброени от фотовратата в скорост или път. Комплектът включва диск с 22 отвора в него и колело с диаметър 5,1±0,1 см. Не е трудно да се изчисли, че 1 импулс от фотопортала ще означава, че колелото е завъртяно 1/22 от пълен оборот, което съответства на разстояние от d = 0,72 см. В същото време фотовратата измерва и изпраща информация през интервал от време t, отчетено в милисекунди между последователните импулси. Моментната скорост може да се изчисли чрез разделяне на 0,72 см на този интервал от време.

Може да се предприемат следните стъпки, независимо дали учениците са разделени на различни групи, или работят в една група. Ако са в една група, то тя ще преминe през последователните стъпки една след друга, докато различните групи ще си поделят трите задачи.

<Програмиране с Arduino>

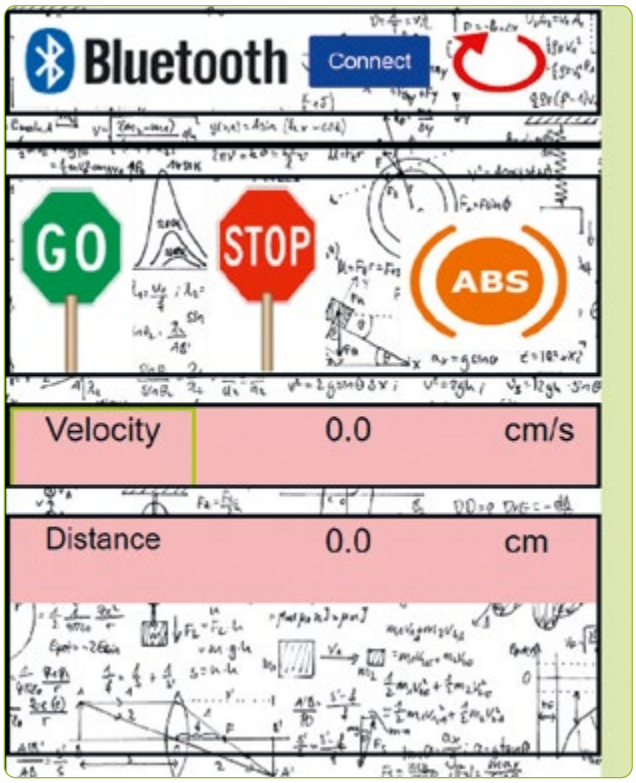
Групата, която ще работи по програмирането с Arduino ще разработи кода, като използва следния подход:

1. Дефинира действията и съответните методи или функции за събиране и изпращане на данни посредством Bluetooth.
2. Написва и тества код за всеки метод по отделно.
3. Сглобява програмата.

Начинаещите може да започнат с TinkerCad[6], която позволява онлайн дизайн, свързване на Arduino и симулации, като по този начин се предотвратяват къси съединения и прегаряния по веригата на етап прототип.

Следващите секции дефинират всяка част по-подробно:

1. Необходимите действия са: стартиране и спиране на двигателя, прилагане и освобождаване на спирачките, измерване на разстоянието, измерване на скоростта, изпращане и получаване на данните чрез Bluetooth.
2. Решаващата част тук е да се напише код за измерване на скоростта и разстоянието, изминато по време на същия експеримент. И двете от тях използват



📷 2: Потребителският интерфейс на приложението

импулси от фотовратата и се активират, когато '2' се получи от приложението за телефон чрез Bluetooth:

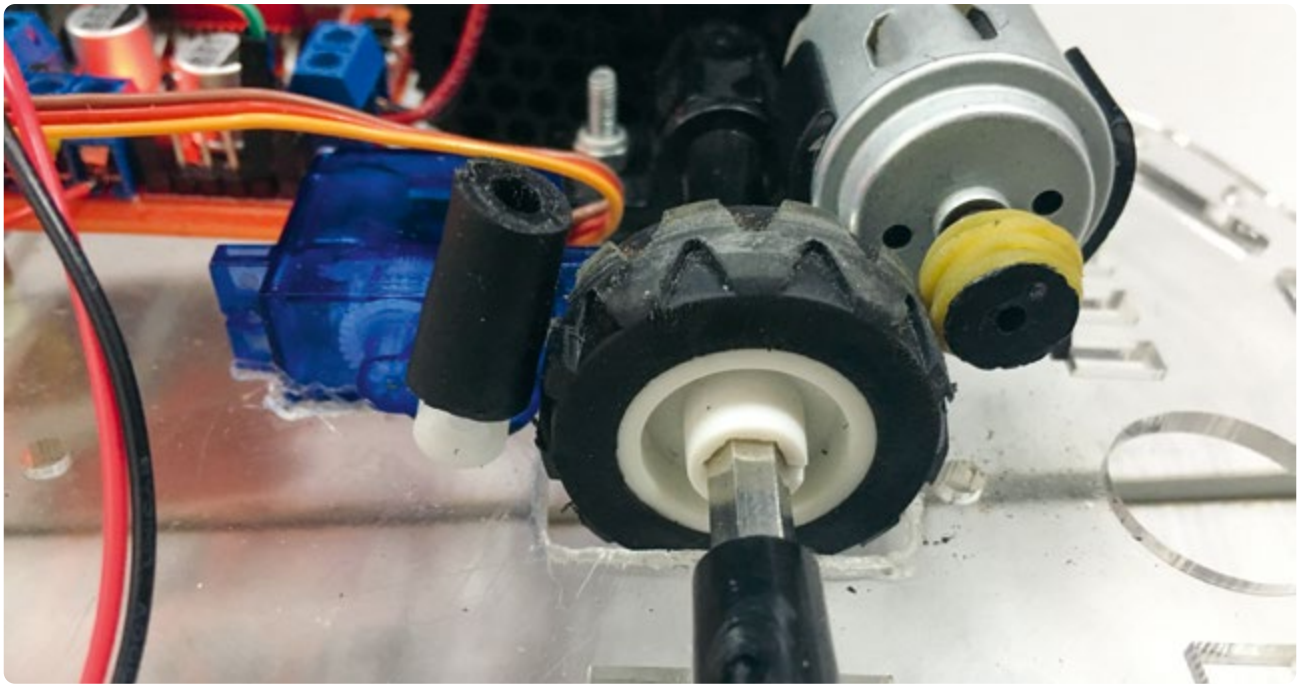
- ↳ За да се измери разстоянието използваме брояч, който измерва импулсите, изпратени до Arduino от свободно въртящото се задно колело, след прилагане на спирачките към предните колела.
- ↳ Времевите интервали между импулсите се използват за измерване на моментната скорост. Задното колело изминава 0.72 см за един импулс време. Ето защо това трябва да се раздели на времевия интервал между импулсите.
- ↳ Функционирането и ефективността на системата против блокиране на спирачките (ABS) може да се отчете след като спирачките се задействат и отпуснат за различни интервали от време от 50 до 200 милисекунди (оптималният интервал се намира с експерименти), което в повечето случаи води до по-кратък спирачен път.

3. Когато програмирате код за Arduino, трябва да се гарантира, че всички дейности и функции се случват в един голям цикъл. Следователно, ако програмата бъде прекъсната на някоя конкретна стъпка, това ще засегне всички следващи стъпки, тоест програмата няма да сработи.

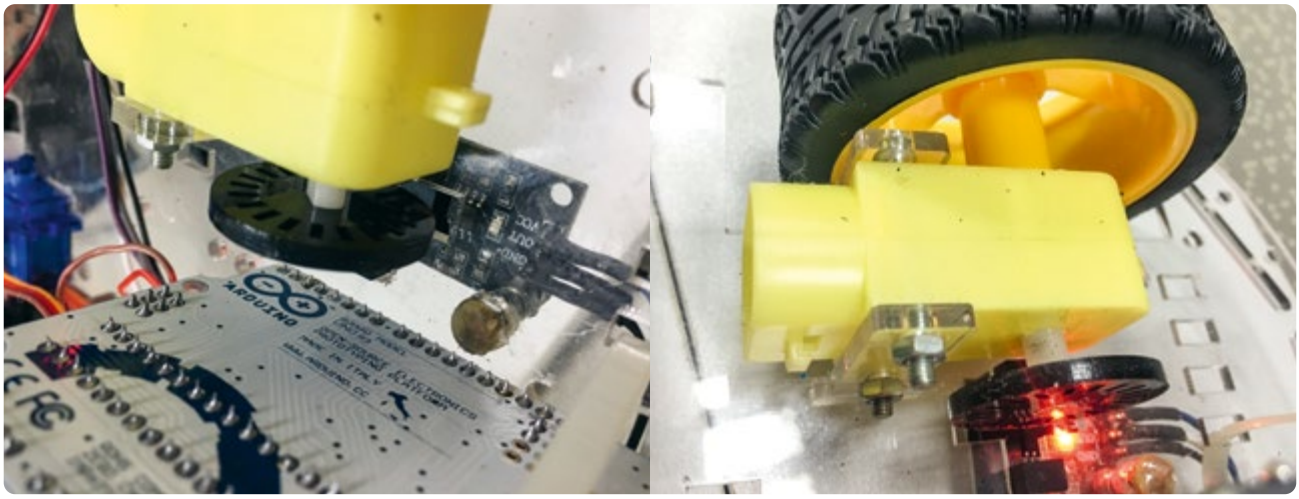
Примерният код и препратките към други източници за всяка от тези функции са достъпни онлайн[7], но с малко помощ от учителите си, учениците следва да бъдат насърчавани да опитат да напишат свой собствен код.

<Програмиране с Android>

Групата, която програмира с Android, може да проучи възможностите на AppInventor[3] и да помисли за начини, по които да показва данните на екрана (потребителски интерфейс, UI). Учениците ще решат къде и как да поставят бутоните за управление на колата, както и панелите и индикаторите на екрана (📷2). Кодът на програмата



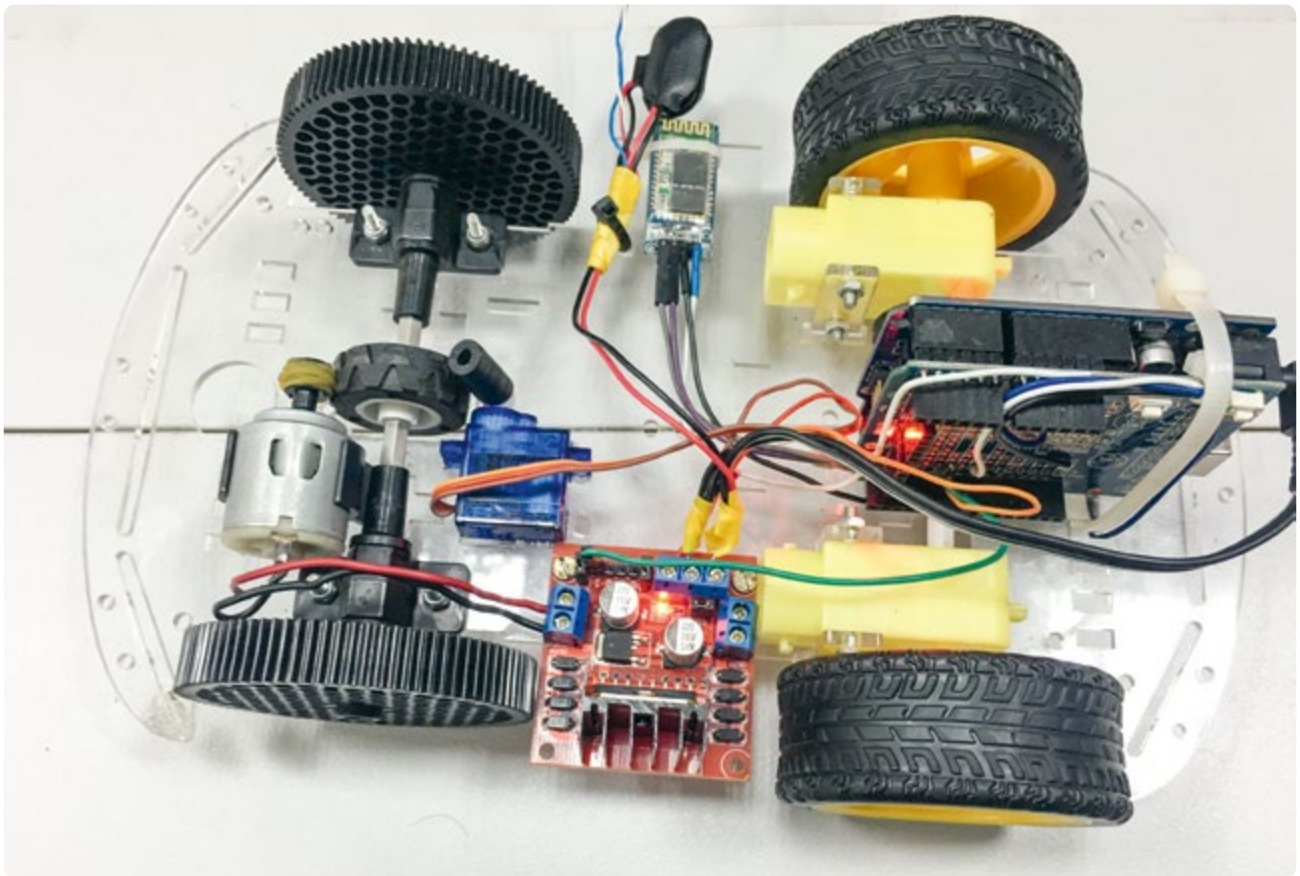
📷 3: Спирачната система в близък план



4: Фотоврата

AppInventor е предоставен онлайн[7] и има следните функционалности:

1. Pushing the **START** button sends '1' to the Arduino[1] via Bluetooth and starts the motor in the car.
2. При натискане на бутон **STOP** се изпраща сигнал '2' до Arduino чрез Bluetooth, което спира мотора и задейства спирачките.
3. При натискане на бутон **ABS** се изпраща сигнал '3' до Arduino чрез Bluetooth, което спира мотора и задейства спирачките на постоянни интервали (симулира се ABS система).
4. След натискане на бутона **STOP** или **ABS** бутона, данните за моментната скорост преди спиране и спирачния път на колата, след натискане на спирачките, ще бъдат показани на два панела, обозначени съответно със 'скорост' и 'разстояние'.
5. При натискане на бутон **RESET** се изпраща сигнал'0' до Arduino чрез Bluetooth, което занулява и изчист-



5: Сглобена кола

ва данните за скоростта и разстоянието, след което нулира Arduino модула.

Предлагаме да използвате предоставените кодове, само като помощен материал за учители и да дадете възможност на учениците сами да изследват AppInventor[3] и да напишат собствен код, базиран на посочените по-горе функционалности.

<Сглобяване на колата>

От групата за изграждане на колата ще се очаква да мисли за подходящи места и начини за закрепване на компонентите: мотор, фотоврата, серво, батерии, Bluetooth модул, модула за защита на мотора и накрая самата Ардуино платка. Когато сервото се завърти, е много важно да притисне плътно оста с гуменото уплътнение към въртящия се диск, монтиран на оста на предните колела (📷3).

Това ще приложи достатъчно сила, за да спре колелата незабавно. Позиционирането на фотовратата е изключително важно. Поставете я така, че тя коректно да отброява всички импулси – препоръчва се фото сензор Arduino, с вграден LED, който примигва, когато импулс попадне върху него. Така ще можете да се уверите, че фотовратата отброява правилно импулсите, когато задните колела се завъртят (📷4).

Също така се уверете, че задните колела се въртят възможно най-свободно. Не забравяйте, че изчисляваме скоростта и разстоянието, като използваме свободното въртене на задните колела. Накрая, когато колата е готова за експеримента, тя трябва да изглежда подобно на тази, изобразена на 📷5.

Препоръчваме ви да дадете подробни насоки за отправна точка на учениците, след което да им предоставите възможност да мислят и приложат собствените си решения за решаване на задачата.

<Заключение>

Този проект е забавен начин да се научат основни понятия от физиката, като сила на триене в покой и движение, с помощта на проучването на сравнително модерни технологии като ABS, където физиката, електрониката, програмирането и дизайнът се събират, за да изследват фактори, влияещи на спирачния път на колите. Винаги е предизвикателство да се интерпретират и анализират реални експериментални данни. Така се въвеждат ключови понятия като несигурност, валидност, възпроизводимост и визуализация. Проектът позволява на учениците да се запознаят и да разберат силата на триенето в практически условия по начин, който провокира тяхното мислене и креативност.

<Сътрудничество>

Този проект предлага голям потенциал за сътрудничество, тъй като трите му независими елемента – дизайнът на автомобила, програмирането на Arduino[1] и програмирането с AppInventor[3] – могат да бъдат допълнително развити и подобрени. Всички партньори, които сътрудничат, ще се възползват взаимно от приноса си към всеки от тези компоненти.

Друга възможност за сътрудничество би могла да бъде конкуренцията между училищните екипи, основана на това кой може да направи автомобил със същите компоненти и да го спре по-бързо, при условие че „пътната“ повърхност и скоростта преди спирането са едни и същи.

Много благодарности на нашите колеги от Гърция: Астринос Цуцудакис, който ни предостави важни идеи за физичните аспекти на проекта и Георгиос – за неговите изключително полезни предложения за оптимизиране на кода. Искаме да благодарим и на Йорг Гутчанк за изчерпателната обратна връзка и подкрепа, които направиха този проект още по-интересен и забавен.

<Препратки>

[1] www.arduino.cc
[2] www.arduino.cc/en/Guide/Environment
[3] http://appinventor.mit.edu
[4] http://snap4arduino.rocks/
[5] https://developers.google.com/blockly/
[6] www.tinkercad.com/circuits
[7] Всички допълнителни материали са налични на: www.science-on-stage.de/coding-materials.

Търкалящи се звуци

<Автор> Георгиос Георгулакис
<Автор> Астринос Цуцудакис

<Инфо>

<Ключови думи> фундаментална наука, събиране на данни, кръгово движение, звукови вълни, геометрия, тригонометрия

<Дисциплини> физика, математика, компютърни науки

<Възраст на учениците> 14–17

<Хардуер> Arduino микроконтролер[1] или подобен (с инсталирани необходими драйвъри), микрофон, мощен зумер (източник на звук), мощна настолна бормашина, материали за дървен диск

<Програмен Език> Snap4Arduino[2]

<Ниво на програмиране> средно

<Резюме>

Този интердисциплинарен образователен сценарий съчетава физиката с компютърните науки. Може да се използва в часовете по компютърни науки или в часовете по физика и включва, наред с изчисляването на други физични величини, изчисляване на равномерно кръгово движение и изчисляване на линейната скорост на въртене по два различни начина.

Първият от тези методи открива колко често сигналът на инфрачервен сензор за разстояние е блокиран от малка метална пластина, която е прикрепена към една точка на въртящ се диск, като по този начин се измерва периодът. Вторият метод използва доплеровото изместване на източника, излъчващ звук – зумер, поставен върху въртящият се диск.

<Въведение>

Експериментално проучване на следните физични величини (период T, честота f, линейна скорост v и ъглова скорост ω) на равномерно кръгово движение на базата на знанията, които в Гърция се преподават в прогимназията (ученици на възраст: 14–17 години) и са включени в учебните програми на повечето Европейски страни. Теоретичната подготовка включва и изучаването на доплеровия ефект. Честотата, ъгловата и линейна скорост се изчисляват по добре известните формули:

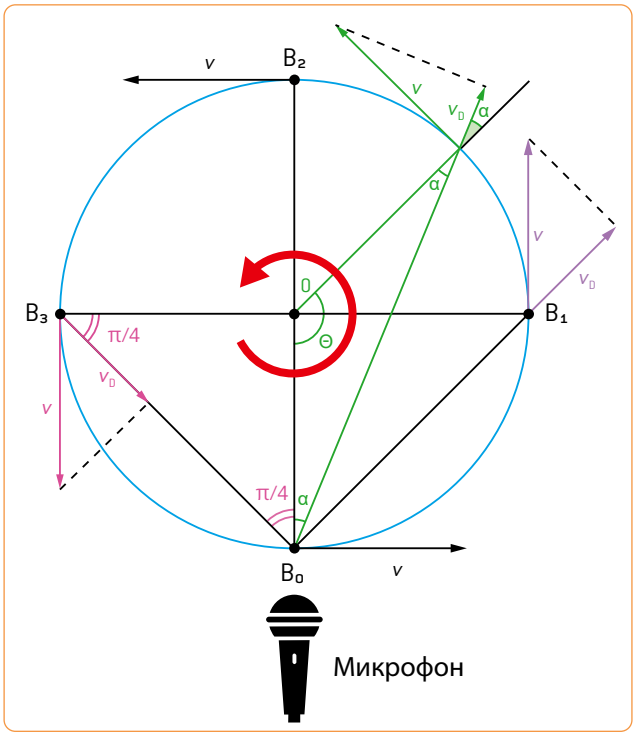
$$f = \frac{1}{T},\ \omega = \frac{2\pi}{T} \text{ и } v = \frac{2\pi r}{T}$$

Периодът T в случая на експеримента се взима от вградения часовник на микропроцесора и е времето между отделните интервали, а радиусът r съответства на разстоянието между металната пластина или зумера и центъра на диска.

<Експеримент базиран на Доплеровия ефект>

Доплеровият ефект е промяната на честотата или дължината на вълната, ако източникът се движи спрямо стационарен наблюдател. Пример за това явление е промяната в тона на сирената на движеща се линейка. Когато линейката се приближава – звукът се чува по-силно, а когато се отдалечава – звукът отслабва. Възприеманият звук е постоянен само в момента, в който линейката се изравни с наблюдателя.

Използваме зумер на въртящ се диск, като източник на звуковата вълна и статичен микрофон като наблюдател в нашия опит (виж 1).



1: Схема на експеримента

Докато дискът се върти обратно на часовниковата стрелка (1), скоростната компонента на линията на хордата, свързваща точка B_0 с точката, където се намира зумерът B всеки път, се увеличава от нула до максимум в точка B_1 и впоследствие намалява до нула в точка B_2. Тази скоростна компонента е действителната скорост на отклонение на Доплеровия ефект. От точка B_2 до B_3 компонентът на скоростта, който сега е скоростта на приближаване, се увеличава от нула до максимум в точка B_3 и след това отново намалява до нула в точката B_0.

Изчислете линейната скорост, като приложите формулата за доплеровото отклонение за движещ се източник в точка B_3 и наблюдател в покой. Линейната скорост остава постоянно перпендикулярна на радиуса на

кръга и ъгълът $\frac{\pi}{4}$ се определя по познатите ни от геометрията свойства на правоъгълния триъгълник B_3OB_0.

$$f = f_0 \cdot \left(\frac{v_s}{v_s - v_D}\right) \Rightarrow f = \frac{f_0 \cdot v_s}{v_s - v_D} \Rightarrow f \cdot v_s - f \cdot v_D = f_0 \cdot v_s$$

$$\Rightarrow f \cdot v_D = (f - f_0) \cdot v_s \Rightarrow v \cdot \cos\left(\frac{\pi}{4}\right) = \left(\frac{f - f_0}{f}\right) v_s$$

$$\Rightarrow v = \left(\frac{f - f_0}{f}\right) \frac{v_s}{\cos\left(\frac{\pi}{4}\right)}$$

f: измерена честота
f_0: излъчена честота
v: скорост
v_s: скорост на звука
v_D: скорост на отдалечаване/приближаване

Тъй като прилагането на бързата трансформация на Фурие за извличане на честотното съдържание на произведения звук е много над възможностите за програмиране на учениците, безплатен софтуер за редактиране на аудио като Audacity[3] ще осигури адекватен текстови файл с всички необходими данни.

<Допълнителни материали>

Друг метод, който използва тръбата на Пито и сензор за диференциално налягане (движещия се въздушен поток в тръба, създава налягане, което е пропорционално на скоростта на движение на тялото), умишлено е оставен извън обхвата на това упражнение, за да се опрости моделът, но цялата необходима информация е достъпна онлайн[4]. Онлайн материалите предлагат подробно описание на експеримента, алтернативни идеи, заедно с теоретична документация и анализ на използваните похвати стъпка по стъпка.

<Какво правят учениците/учителите>

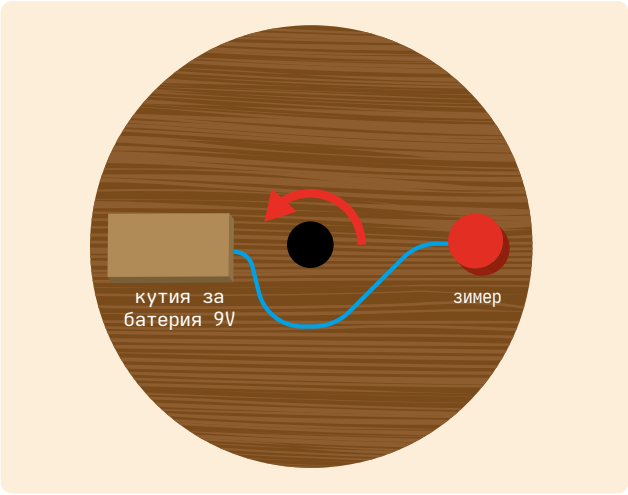
В частта, свързана с физиката в тази учебна единица, учениците ще измерват физичните параметри на кръгово движение с различни радиуси и ще изследват доплеровия ефект. Но първо, ще трябва да проектират и сглобят основна експериментална платформа.

<Дървена дискова платформа>

Учениците ще изградят платформа състояща се от дървен диск, който се върти от бормашина, а върху него е прикрепен зумер, свързан с 9V батерия. В близост е разположен независим инфрачервен сензор за разстояние, който предава сигнали на микроконтролера за всеки пълен кръг, а микрофон записва звуците, които той възпроизвежда. При идеални условия Доплеровия ефект трябва да бъде доловим за слуха за определена скорост на въртене, която не бива да бъде много висока от съображения за сигурност. (📷2 и 3)

Разработено е приложение с атрактивен и удобен за учениците интерфейс, както е показано на 📷4 , за въвеждане на всички необходими параметри и извличане на изчислените стойности.

📷 2: Интерфейс на експеримента



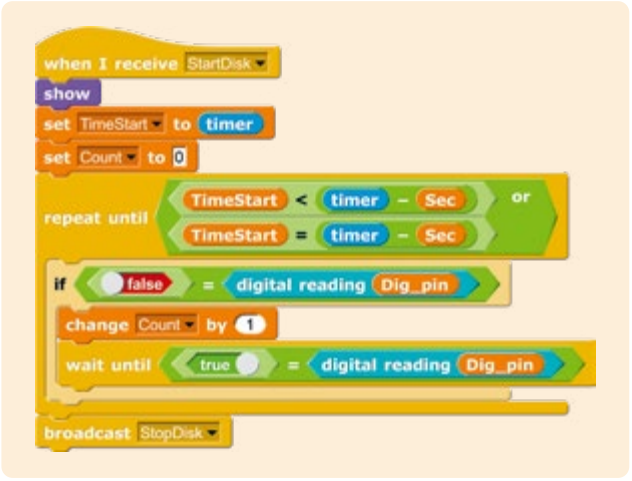
3: Дискът, погледнат отгоре, с кутия за 9V батерия за захранване на зумера

4: Интерфейс на експеримента

Учениците се нуждаят от основни умения за програмиране и опит с използването на блокови езици за програмиране (като Scratch или Snap!). Ние даваме на учениците основен шаблон, за да разработят техен код. Така се гарантира, че те ще се фокусират върху целите на урока, а не върху потребителския интерфейс и външния вид на програмата.

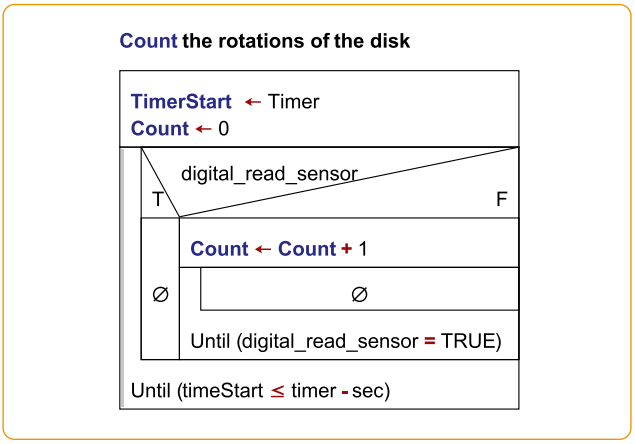
Ето защо, ние предлагаме основния шаблон на програмата в Snap![5] файла project.xml и работен лист, който дава на учениците основни инструкции за шаблона на програмата и обяснение какво очакваме от тях. Както шаблона, така и задачите са достъпни онлайн.[4]

Учениците ще проверяват и валидират получените данни, ще се свързват и комуникират с външни устройства, ще получават и обработват данни от сензорите и ще напишат прост алгоритъм за серийно търсене.



5: Изчисляване на ротационния период

Завършената програма е достъпна за изтегляне от учителя като връзка[4]. 5 съдържа примерна екранна снимка на средата за програмиране Snap4Arduino[2]. 6 съдържа съответната диаграма на Наси – Шнайдерман.



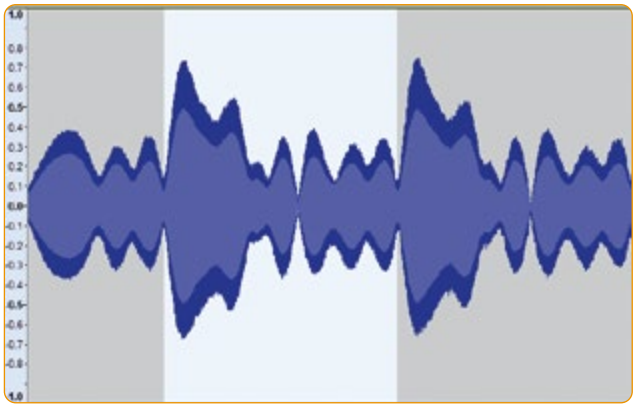
6: Диаграма на Наси – Шнайдерман за периодичност (цикли)

Както вече споменахме, препоръчваме да използвате безплатен софтуер за редактиране на звук, като Audacity[3], за да извлечете честотното съдържание на възпроизведения звук. Софтуерът осигурява адекватен текстови файл с всички необходими данни за учениците, за да се научат как да обработват звуков сигнал чрез специализиран софтуер.

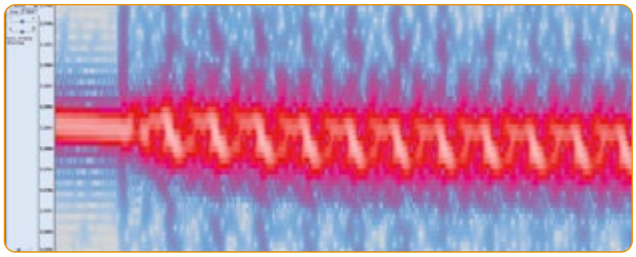
Импортирането на сигнала и основната обработка са илюстрирани на 7–9, докато 10 показва част от изходните данни. За да се избегне задълбочен анализ и да се даде възможност за по-добро разбиране и усвояване на материала от учениците, в този момент трябва да се направи предположение. Жълто маркираната честота, която притежава максималното ниво, е честотата на зумера в покой, измерена в точки B_0 и B_2, докато изобразе-

ната в синьо и зелено – показват нашите честотни изместванията в най-близките пикове (📷10).

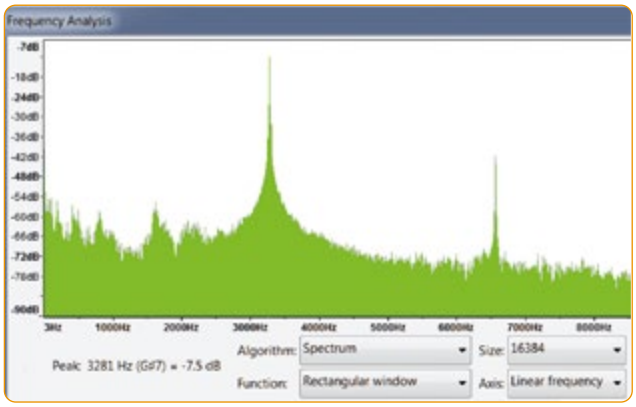
Все пак, предложената програма търси само честотата откроена в зелено, но за по-голяма точност може лесно да се промени, за да се намерят и двете /синята и зелената/. За да ускорим нещата, разделът с данни може също така да бъде ограничен само от 50 до 100 стойности над и под честотата на максималното ниво.



📷 7: Запис на звукова вълна

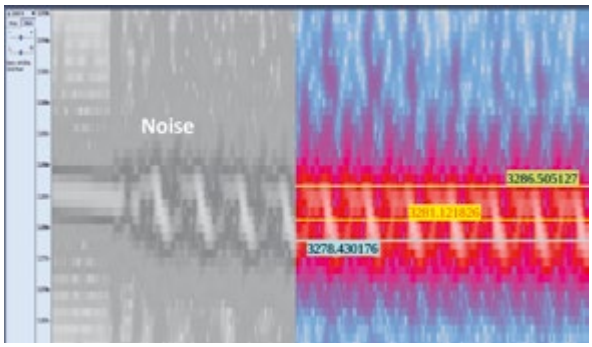


📷 8: Спектрограма показваща Доплеровото отместване



📷 9: Анализ на честотата от бързата трансформация на Фурие

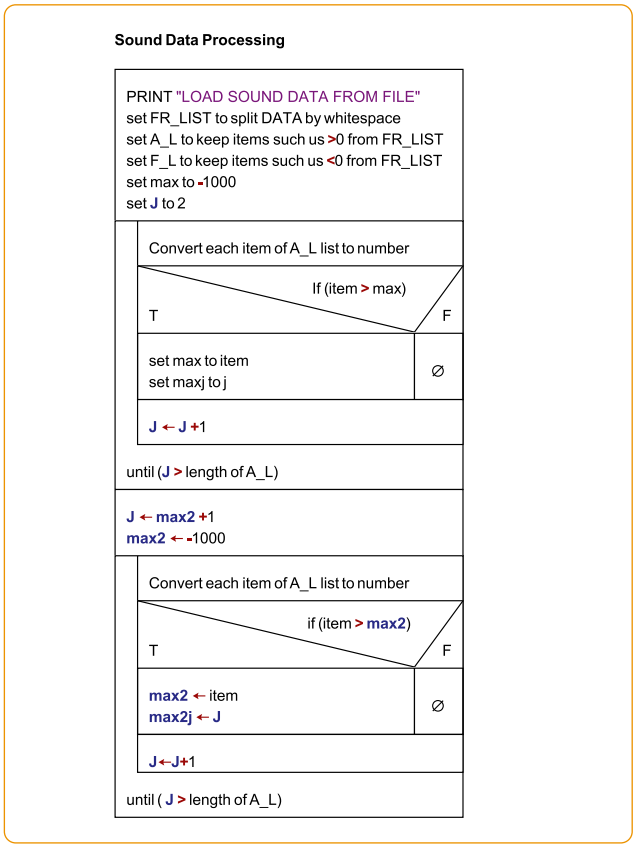
Frequency(Hz)	Level (dB)
3273.046875	-27.597595
3275.738525	-22.331339
3278.430176	-12.437067
3281.121826	-7.5547090
3283.813477	-10.041918
3286.505127	-9.7750780
3289.196777	-16.848948
3291.888428	-26.916197



📷 10: Изходни данни

Обработката на данните от звуковия спектър е показана на 📷11 и 12. Подробна информация за използваните променливи е налична онлайн.[4]

📷 11: Доплерово отместване – част от експеримента[4]



12: Диаграмата на Наси – Шнайдерман за обработка на звукови данни

<Алгоритъм за други програмни езици>

Основният шаблон дава възможност за лесно пренасяне на кода към всеки друг език за програмиране, стига да има основна библиотека за комуникация с микроконтролер. Следователно изборът на микроконтролер няма да окаже значително влияние върху проекта.

<Заключение>

Това е ниско-бюджетен проект, който е лесен за сглобяване и експлоатация; ще бъде интересен и стимулиращ за учениците.

<Сътрудничество>

Платформата Наука на сцената е създадена за сътрудничество и обмен на идеи за преподаване и прилагане на иновативни образователни подходи. Съвместното преподаване с Илия Мествиришвили и Дейвид Шапакидзе, е чудесен пример за учителски екип от Грузия, които преодоля в голяма степен предизвикателствата на голямото разстояние и проблемите с работния график, давайки възможност да разработим нови техники на работа заедно. Въпреки липсата на опит в обучението на ученици със специални образователни потребности, е добра идея този учебен материал да се модифицира, така че да стане наистина достъпен за всички ученици.

<Препратки>

[1] www.arduino.cc
[2] http://snap4arduino.rocks
[3] www.audacityteam.org
[4] Допълнителни материали са налични на www.science-on-stage.de/coding-materials.
[5] https://snap.berkeley.edu

↳ www.physicsclassroom.com/mmedia/circmot/ucm.cfm
↳ https://education.pasco.com/epub/PhysicsNGSS/BookInd-904.html
↳ http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/dopp.html
↳ http://newton.phys.uaic.ro/data/pdf/Doppler_experiment.pdf
↳ https://manual.audacityteam.org/man/tutorials.html (Декември 2018)

Физиката като двигател

<Автор> Михаела Ирина Гюргея
<Автор> Корина Лавиния Тома

<Инфо>

<Ключови думи> анимация, спрайт, блокове, контури, графики, гравитационни закони, сблъсъци, свободно падане, траектория на хвърляне, сила на триене, движение, импулс, оператори, променливи

<Дисциплини> компютърни науки, физика, математика, ICT

<Възраст на учениците> 14–16

<Хардуер> компютър

<Език> Scratch[1]

<Ниво на програмиране> начално, средно

<Резюме>

Какво бихте си помислили, ако ви кажем, че вашите ученици могат да учат по-лесно и едновременно два, на пръв поглед много различни предмета – физика и компютърни науки? В тази глава „двигателят“, т.е. средата за програмиране Scratch[1], е вълшебният инструмент, който ще помогне на учениците да създадат интересни приложения, свързани с ежедневните природни явления, за да разберат по-добре законите на физиката и в хода на всичко това да подобрят уменията си за програмиране.

<Въведение>

Защо използвахме Scratch[1]? Scratch е среда за визуално програмиране, която анимира спрайтове (герои), използвайки блокове на компютърния екран и помага на учениците да създават приложения по-лесно, отколкото с класически програмни среди (C ++, Java и др.). В допълнение, нашият „двигател“ ни помага да преподаваме два предмета, които учениците считат за трудни: физика и компютърни науки. Премахвайки основните препятствия, невъзможността да си представим как действително работи дадено явление и сложният синтаксис на програмирането, ние създадохме приятен нов начин на преподаване.

Учениците, които участваха в разработването на този учебен модул, имаха известен опит в програмирането, така че успяха да се справят със Scratch. Те го използваха, както в редовните, така и в избираеми часове по компютърни науки. Освен това, необходимите им знания по физика са част от стандартната учебна програма и винаги е полезно да се преговори и приложи това, което е научено.

<Какво правят учениците/учителите>

В тази част се редуват последователно обучения по теми, включващи програмиране и физика.

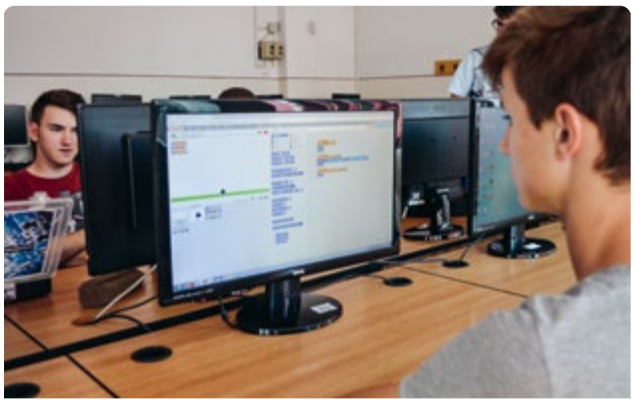
Първо учителят по компютърни науки представя основите за създаване на проект в Scratch[1]. Учениците се запознават с няколко ключови думи, свързани със средата за програмиране Scratch: сцена, спрайтове, костюми и движение. Можете да следвате инструкциите и обясненията на първото приложение за обучение в Scratch, което е без формули от физиката.[2]

Учениците трябва да разберат взаимодействията между спрайтовете и тяхната синхронизация, както и как работи координатната система. Можете да намерите пълен урок за Scratch онлайн.[3]

За да разберат по-добре основните алгоритми в Scratch, учениците разгледаха вълнуващи и интересни приложения на сайта на Scratch. Виждайки кода зад тях, бяха изненадани да открият, че сами могат да създават такива приложения.

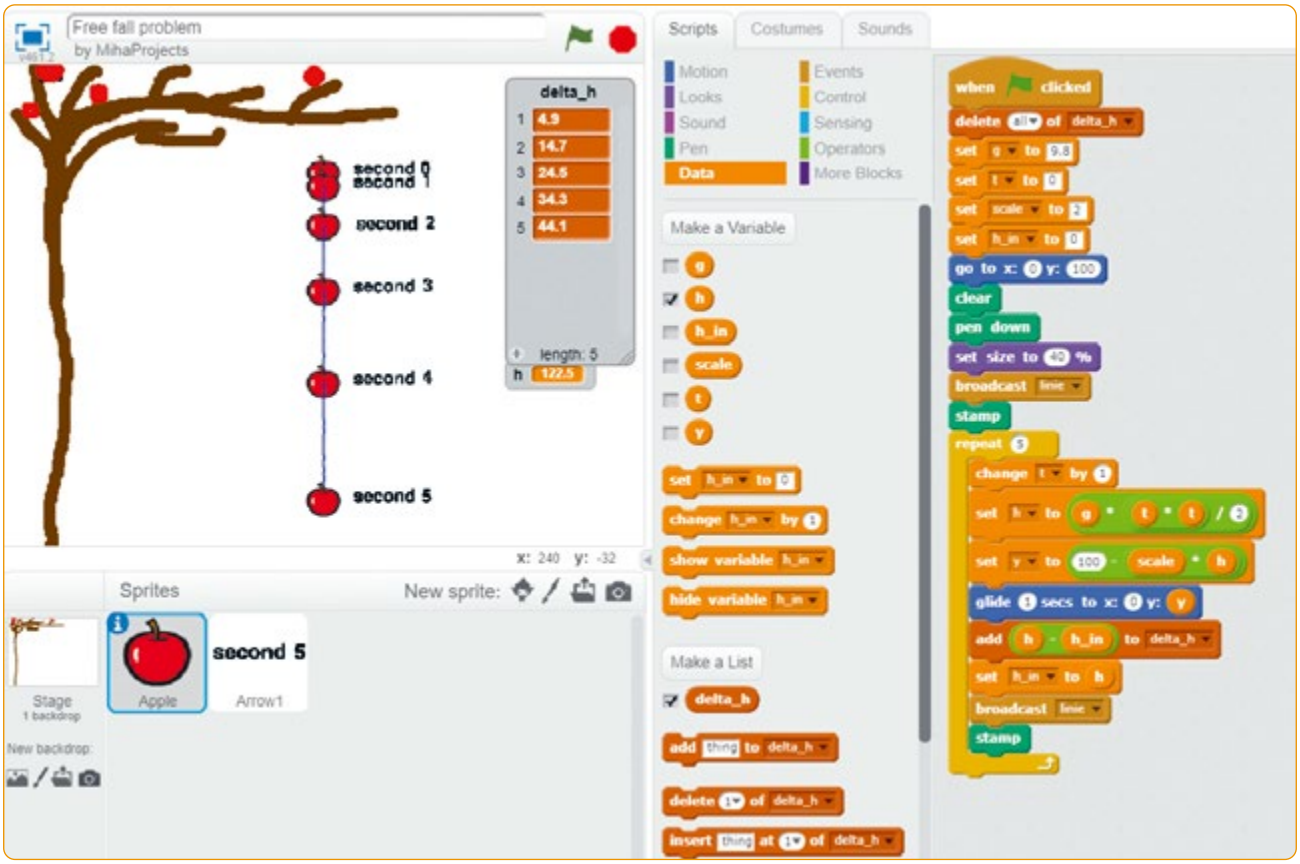
За частта по физика учениците прилагаха теоретичните принципи, стоящи зад явленията от света около тях. Учителят по физика предложи голямо разнообразие от теми[4], които учениците след това обсъдиха: необходими формули, възможна анимация, дизайн и т.н.

Учениците избраха някои от тези теми. Седмица по-късно получихме приложения за траекторията на хвърляне на топка, свободното падане на ябълка, по-сложното падане на капка вода или сблъсък между две топки, а също и за движението на планетите в Слънчевата система или дори малки игри.(📷1)



📷 1: Индивидуална работа

Накрая всички представиха проекта си пред класа и получиха обратна връзка от своите връстници. Това



2: Задача свободно падане

улесни учениците да разберат кои части от проектите им трябва да бъдат подобрени: програмирането или физиката.

В края на нашия проект по-големите ученици станаха учители на по-малките ученици (12–13 години), като представиха подходящи приложения и ги тестваха по време на уроците по физика. Те се радваха на внимание и много се гордееха с работата си. По-големите ученици също получиха предложения от по-малките ученици. Най-добрите от тези симулации за ученици е достъпно в платформата Scratch.[2]

Следващите раздели съдържат примери как подходихме към частта по физика и програмиране.

<Приложение 1: Проблем със свободно падане (ябълка на Нютон)>

Всеки ученик е чувал за свободното падане на този исторически обект – ябълката на Нютон.

Приложението на 2 е вдъхновено от класически проблем. Какво е разстоянието, изминато всяка секунда от ябълката на Нютон при свободно падане?

Физика теория

Ние намираме линейното движение с постоянно гравитационно ускорение $g = 9{,}8\ \frac{\text{m}}{\text{s}^2}$.

След време t изминатото разстояние $h(t)$ на ябълката е: $h(t) = \frac{gt^2}{2}$.

Началната точка $h(0)$ е фиксирана върху клона на дървото, от който ябълката се отделя.

След това изчисляваме изминатото разстояние за по-дълъг период, $t + \Delta t$: $h(t+\Delta t) = \frac{g(t+\Delta t)^2}{2}$.

Общата формула за разстоянието $\Delta h(t)$, разстоянието, изминато от ябълката за времето Δt, е:

$$\Delta h(t) = h(t+\Delta t) - h(t) = \frac{g(2t\Delta t + \Delta t^2)}{2}.$$

Ще използваме данните за конкретния проблем, за да персонализираме основната формула; в този случай 1сек за Δt. За първата секунда $t = t_{in} = 0$ резултатът е $\Delta h_1 = 4{,}9$ м, за следващата секунда, t=1сек резултатът $\Delta h_2 = 3 \cdot 4{,}9$ м = 14,7 м и т.н. Чрез математическата логика можем да изчислим изминатото разстояние през n-тата секунда, отчитайки $t = (n-1)$ сек:

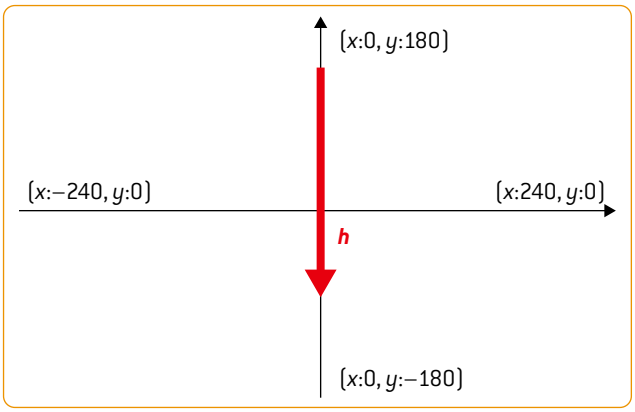
$$\Delta h_n = \frac{9{,}8(2n-1)}{2}\ \text{m}.$$

Тогава учениците могат да изчислят, а също и да видят в нашата анимация, че изминатото разстояние се увеличава със същата стойност всяка секунда – 9,8 м.

Как да програмираме това?

Използвани променливи:
g: гравитационно ускорение
t: брояч на секунди (със стойности: 0, 1, 2, 3, 4, 5)
h: изминатото разстояние след *t* секунди
h_in: първоначалната позиция на ябълката
delta_h: списък (масив) с всички изминати разстояния във всяка секунда
y: y-координата на ябълката
Забележка: В това приложение координатата x остава постоянна = 0, така че можете да преместите по-лесно траекторията наляво или надясно на сцената.

В началото ябълката е в точката с координатите (0,180). Началната точка и посоката за изминатото разстояние *h*(*t*) са отбелязани на 3.



3: Ориентация в координатна система

```
Free_fall

delta_h: array of real
g←9.8
t←0
h_in←0
scale←2
go to (0,100)
clear()
pen(down)
stamp()

while (t<5)
    t← t + 1
    h← g * t * t / 2
    y←100-scale*h
    glide (0, y)
    add (delta_h, h-h_in)
    h_in ←h
    broadcast(linie)
    stamp()
```

4: Задача свободно падане

Използвайки цикъла 5 пъти, преизчислихме разстоянието, което ябълката изминава след всяка секунда, изчислявайки новата y-координата и отчитайки характеристиките на екрана. Вижте 4 за алгоритъма на програмата.

Предизвикателство

Учениците модифицират Δ*t* и изминатото време *t* (ябълковото ни дърво може да е много високо – вероятно е по-добре да нарисуваме кула) или преместват задачата върху друга планета със собствено гравитационно ускорение. За да създадат сложен проект, те биха могли да добавят силата на триене от въздуха и да обмислят променливо гравитационно ускорение, като хвърлят ябълката от метеорологичен балон на по-голяма надморска височина.

<Приложение 2: Падаща капка вода>

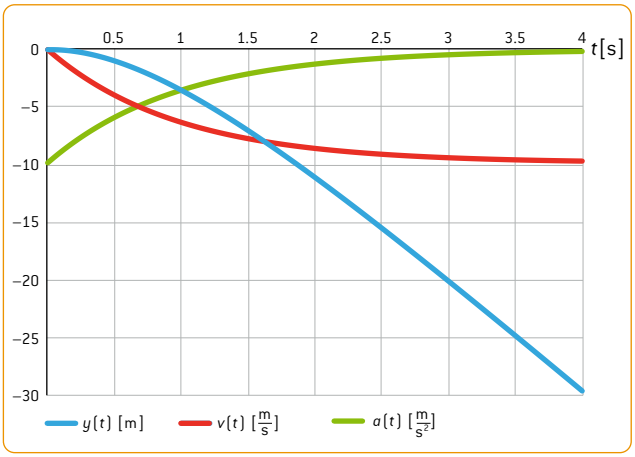
В дъждовен ден всеки може да наблюдава падането на водни капки. Учениците анализираха линейното движение на една капка с нашата симулация. В началото те видяха, че падането на водната капка се ускорява, но с намаляващо ускорение. След известно време скоростта на капката достигна своята граница, крайната скорост v_t, когато ускорението достига нула. След това капката вода продължи да се движи с тази постоянна скорост. Как можете да обясните това?

Физика теория

В ускоряващата част от движението, две сили в противоположни посоки действат на капката: гравитационната сила $G = mg$ (*m*: маса на капката, *g*: гравитационно ускорение) и силата на триене $F_f = kv$ (*k*: константа на пропорционалност, *v*: моментна скорост). Ускорението на капката става: $a = g - \frac{k}{m} v$.

В нашата симулация измерваме голяма капка, с диаметър около 5 мм с крайна скорост $v_t = 9{,}8\ \frac{m}{s}$.[5] В този случай константата е $\frac{k}{m} = \frac{1}{s}$. Ускорението намалява, когато скоростта се увеличава (виж 5). Първоначалните стойности са $a = 9{,}8\ \frac{m}{s^2}$, $v = 0$ и $y = 0$.

Използваме малка програма в C++[4], за да изчислим моментното ускорение и скорост, където отчитаме ускорението и константата на скоростта за много малки времеви интервали Δ*t* (като 0,05 сек). В този случай скоростта се увеличава $\Delta v = a\Delta t$, а изминатото разстояние с $\Delta y = v\Delta t$ за всеки избран Δ*t* (метод стъпка по стъпка).

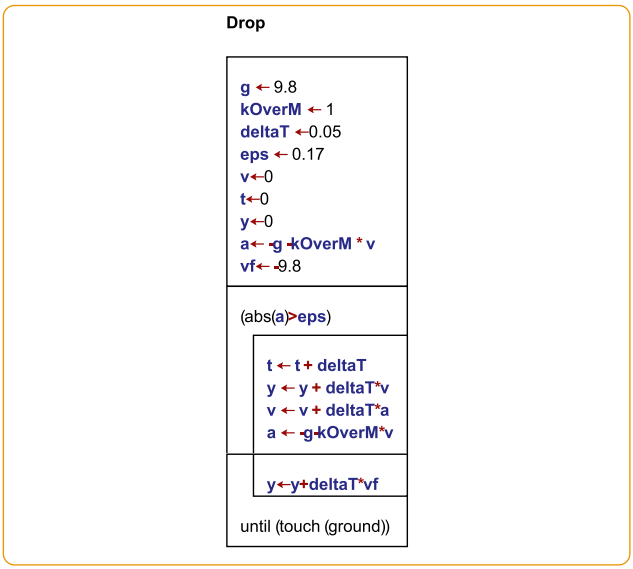


5: Отношение между изминатото разстояние, скоростта и ускорението

Как да програмираме всичко това?

1. Оцветете спрайта водна капка.
2. Оцветете една хоризонтална зелена линия в долната част на фона.
3. Направете спрайт със съобщението „ускорение = 0“, което се появява, когато ускорението има стойност около 0.
4. Напишете кода за спрайта – капка. Капката старира в точката $(0, y_{init})$. Използвайки цикъл, преизчисляваме $a(t)$, $v(t)$, $y(t)$. Използваме разстоянието, достигнато при падането след всеки Δt, като изчисляваме новата y-координата и отчитаме характеристиките на екрана. Ускорението намалява и когато е около 0, цикълът е завършен. В този момент на екрана се показва спрайта със съобщението. След това капката пада с постоянна скорост, докато не докосне зелената линия на фона.

6 дава ясно обяснение на кода.



6: Падаща капка

Предизвикателство

Учениците могат да подобрят това приложение, ако добавят променлива за масата на водната капка (диаметърът на водната капка обикновено може да приеме стойности от 1 мм до 5 мм)[6] и друг вид сила – на триене: $F_f = \frac{kv^2}{2}$.

Учениците могат също да добавят повече капки с различни маси и да сравнят как падат.

<Приложение 3: Еластичен сблъсък>

Има много примери за сблъскване на тела около нас. Тези сблъсъци са сложни, но ние считаме еластичните сблъсъци за приложими в реалния живот за билярдни или стоманени топки или на теория за сблъсъци на молекули, когато учениците изучават модела на идеалния газ.

Физика теория

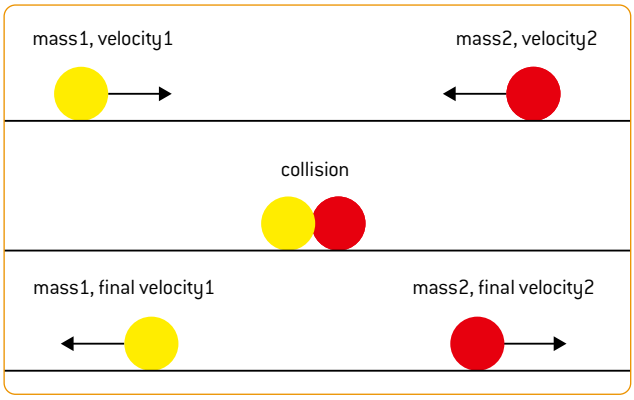
Импулса и кинетичната енергия за две топки с масата m_1 и m_2, начални скорости $(\vec{v_1})$ и $(\vec{v_2})$ и крайни скорости $(\vec{v_{1f}})$ и $(\vec{v_{2f}})$, движещи се една срещу друга, се запазват. [7]

$$m_1\vec{v_1} + m_2\vec{v_2} = m_1\vec{v_{1f}} + m_2\vec{v_{2f}} \text{ и}$$

$$\frac{1}{2}m_1v_1^2 + \frac{1}{2}m_2v_2^2 = \frac{1}{2}m_1v_{1f}^2 + \frac{1}{2}m_2v_{2f}^2$$

Ако движението се извършва по една и съща линия (движение по x-оста), можем да използваме знаци + или -, за да обозначим посоките. Векторното обозначение на скоростта не е необходимо за случая на сблъсък по права линия и крайните скорости могат да бъдат изчислени по следните уравнения:

$$v_{1f} = 2\frac{m_1v_1 + m_2v_2}{m_1 + m_2} - v_1 \text{ и } v_{2f} = 2\frac{m_1v_1 + m_2v_2}{m_1 + m_2} - v_2.$$



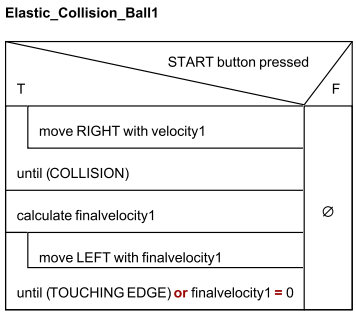
7: Еластичен сблъсък

Как се програмира това?

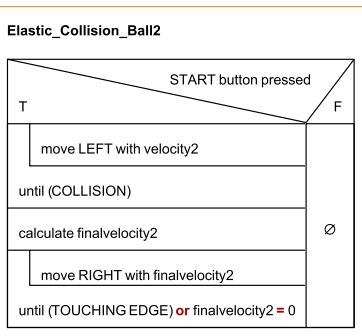
1. Изберете два спрайта за топките (Ball1 и Ball2) и един спрайт за бутона **START** (START спрайт).

2. Използвайте променливи: *маса1*, *маса2*, *скорост1*, *скорост2* (масата и началната скорост) за всеки обект. Направете променливи плъзгачи и задайте минималната и максималната стойност за тях.
3. Въведете масата и началните скорости за всеки обект.
4. Натиснете бутона **START**. В този момент спрайтът изпраща съобщение на спрайтовете на топката. Когато получат съобщението, всяка топка се придвижва към другата, използвайки добре познатата формула *разстояние = скорост × време*.
5. Изчислете крайните скорости на топките и ги използвайте за придвижване на топките в правилната посока, докато топката докосне ръба или напусне сцената, или остане на мястото си, защото новата ѝ скорост е 0.

8 и 9 дават ясна представа за начина, по който са анимирани двете топки в Scratch[1].



8: Еластичен сблъсък за monka 1



9: Еластичен сблъсък за monka 2

Два примера за използване на това приложение:

1. Изберете едната скорост да е равна на 0 и еднаква маса за топките; след този сблъсък ще наблюдавате, че движещата се топка спира, а другата се движи със същата скорост, която първата топка е имала преди удара.
2. Топките имат различна скорост и еднаква маса; след сблъсъка ще видите, че обектите приемат стойността за скорост на другата.

И в двата примера топките променят инерцията си.

Предизвикателство

Учениците могат да променят размера на топките правопропорционално на тяхната маса; те биха могли да направят приложение за двуизмерен еластичен сблъсък (симулация за ефекта на Комптон) или биха могли да програмират симулация за сблъсък на топка със стена. Можете да продължите изследването с друга физична теория, например. нееластичен сблъсък.[4]

<Заключение>

<За учениците>

Предимства

Учениците учат теорията по физика по по-приятен начин и успяват да разберат по-добре природните явления, използвайки симулации в Scratch. Те задълбочават едновременно знанията си по компютърни науки и физика. Въпреки че не всички техни проекти бяха перфектни, учениците определено подобриха уменията си за програмиране и алгоритмично мислене.

Недостатъци

Учениците работят сами и повече вкъщи. Получават обратна връзка в училище.

<За учителите>

Предимства

Наблюдавахме реален интерес на учениците към създаването на оригинално приложение и така научават повече, отколкото в класическите уроци.

Недостатъци

Беше ни трудно да координираме целия клас, поради голямото разнообразие от теми по физика и много грешки във всяко приложение. Смятаме, че би било по-добре да дадем на всички ученици една и съща тема и да ги насърчим да я подобрят максимално, в зависимост от нивото на подготовка и възможностите им.

<Дейност на сътрудничество>

Ученици от различни училища и страни биха могли да решат предизвикателствата в проектите и да създадат нови идеи свързани с първоначалната тема. Всички тези приложения могат да бъдат събрани в платформата Scratch и да се организира конкурс за определяне най-добрите от тях в категории според нивото на програмиране и физика.

<Препратки>

[1] https://scratch.mit.edu/
[2] www.science-on-stage.de/coding-materials.
* Останалите препратки може да видите в английската верисия.

Научна магическа кутия

<Автор> Люк Ивара
<Автор> Марко Николини

<Инфо>

<Ключови думи> микроконтролер, датчик, сензор, преобразувател, изходно устройство, сигнал, физична величина, цикъл, разклонение, последователност, процес, калибриране, вход, изход, четене, писане, аналогов, цифров, линейност, преобразуване, дъска, щифт, запояване, интерфейс човек-машина

<Дисциплини> физика, електроника, математика, ICT, логика, биология

<Възраст на учениците> 14–18

<Хардуер> Arduino UNO[1] с Arduino DUE[2] и/или TI-Nspire CX CAS с TI-Innovator Hub

<Език> C++ (използване Arduino IDE[3]) и/или TI-Basic

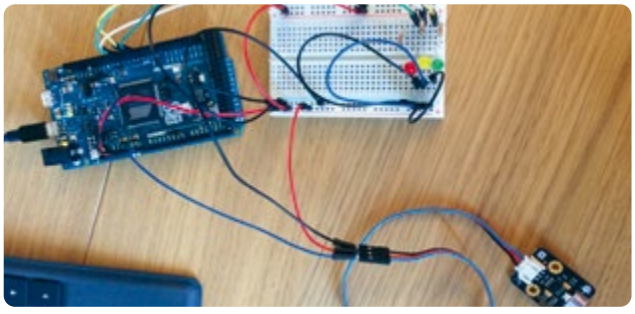
<Ниво на програмиране> средно, с аудио част за напредналите ученици

Списък на абревиатурите, специалните термини и съкращенията, използвани в този раздел, е достъпен онлайн.[4]

<Резюме>

Учениците ще научат как да програмират самостоятелно изградена хардуерно-софтуерна среда (базирана на Arduino) и готов джобен компютър (калкулатор TI-Nspire CX CAS с разширението му, TI-Innovator Hub). И двете се използват като устройства за събиране и преобразуване на данни от сензори за лесно управление, четене, конвертиране и изпълнение.

Използвайки платформата Arduino, учениците ще програмират сензори, които регистрират информация за физични параметри и я подават, като входни сигнали към Arduino, там те задействат преобразувателя, който реагира на получения входен сигнал и пораждат изходен сигнал. След като микроконтролерът е обработил входния сигнал, той го изпраща към правилния изход за изпълнение от изходни устройства (виж 1).



1: Платка Arduino

TI-Innovator Hub е кутия „готова за употреба“, която дава възможност на учениците да научат основите на програмирането. Трябва да бъде включена в калкулатор TI-Nspire CX CAS. Той има добър I/O интерфейс, който включва сензор за светлина, два светодиода и вграден зумер, който издава звук с определена честота (вижте 2).

2: TI-Nspire и TI-Innovator Hub

<Въведение>

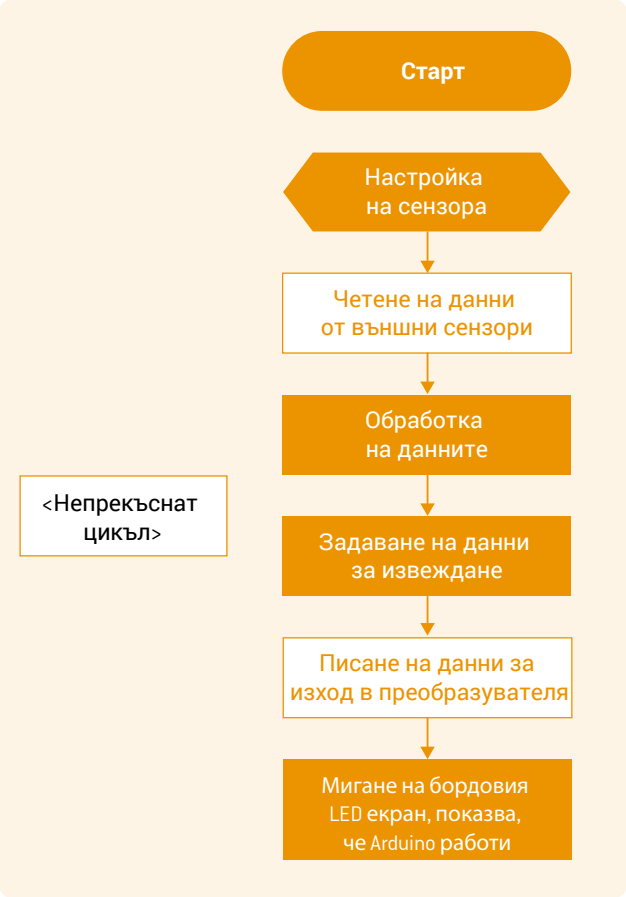
Този раздел запознава учениците със света на програмирането за решаване на физични проблеми, въз основа на откриване и измерване на физични величини, обработка на данните, реакция и вземане на решение за извършване на последващо действие от изпълнителните устройства.

Кодът обикновено се основава на безкраен цикъл (машината обикновено е "жива", докато е захранвана и трябва да работи през цялото време), където трите действия на измерване, обработка и действие се извършват в този ред.

Учениците ще научат, че могат да напишат всеки код с помощта на:

1. последователни инструкции
2. цикли (докато ... прави; повтаряй ... докато)
3. разклонения (ако...тогава...още)

както е посочено в теоремата на Бьом-Якопини (виж „Допълнителна информация“[4]).



3: Блок схема

Втората цел на тази част е да запознае учениците с микроконтролерите. Те ще се научат да настройват входни датчици и изходни изпълнителни устройства, използвайки цифрови или аналогови портове, да напишат проста програма, която чете вход, обработва данни и извежда изход. Те могат да избират звукови или светлинни сигнали, които да бъдат използвани като „аларми“, за да издават предупреждение въз основа на входния сигнал, прочетен от сензорите.

Учениците ще използват Arduino интегрираната развойна среда[1] (Arduino IDE[3]) за програмиране в C ++, с готови за употреба функционални библиотеки, които правят процеса на програмиране по-лесен и бърз.

Трябва да имате на разположение и прототипна платка – Breadboard (вижте „Допълнителна информация“[4]), за да могат учениците да изработват своите прототипи и по-лесно да свързват пиновете на сензорите към Arduino I/Os, 5V захранване и GND пин.

Структурата на всяка програма в блок схема е показана на 3.

Моля, обърнете внимание, че инструкцията *„докато (TRUE)“* е трик да кажете на всеки процесор да повтаря включените инструкции за неопределено време (стига микроконтролерът да се захранва).

Третата основна цел е да научите как да преобразувате регистрираните физични параметри в други такива (например интензитет на светлината в звук), готови за предаване във външната среда, като следвате тези стъпки:

1. Физичният сигнал (светлина, звук, сила, енергия ...) се получава от сензор и се превръща в електрически сигнал.
2. Електрическият сигнал се преобразува в число, достъпно за процесора.
3. Числото се обработва и превръща от процесора в друго число и след това се използва за извършване на действие с преобразувател (преобразувател – изпълняващо устройство, което превръщаща входния сигнал (електрически, оптичен, механичен, пневматичен и др.) в изходен сигнал (светлина, звук, сила, енергия, движение), въздействащ върху управлявания обект).
4. Преобразователят превръща числата в електрически сигнали, които са готови за извеждане.
5. Накрая електрическият сигнал се трансформира във физичен сигнал (например звук, светлина).

В стъпки 1 и 5 сигналите трябва да бъдат преобразувани от една форма в друга. В тези фази линейността на трансформацията или „близката до линейността“ зависимост е изключително важна (вижте „Допълнителна информация“[4]).

Вижте „Допълнителна информация“[4] за подробно обяснение на последния ред от блок схемата („Мигащ бордов светодиод“).

Входният сигнал обикновено се нарича „стимул“ и идва от средата, където сензорите са поставени за получаване на данни. Процесорът и кодът са проектирани да „реагират“ на стимула с математически/логически операции (изпълнявани от инструкциите на кода, обработени от микроконтролера) и да извеждат този „отговор“ в околната среда. В нашите дейности средата е пространството около Arduino, което е в състояние да „вижда“, „чува“ и „чувства сили“, благодарение на сензорите.

Работата с TI-Innovator Hub позволява на учениците да се съсредоточат повече върху програмната част на

своята работа, тъй като микроконтролерите и сензорите вече са настроени и готови за употреба.

<Какво правят учениците/учителите>

Препоръчваме ви да започнете с брейнсторминг, за да събирате прости идеи от всички ученици относно сензорите и автоматичното управление на устройствата. Събирането на тези идеи, натрупването на практически опит със сензорите и автоматичното управление на устройствата, сравняването на резултатите с техните предишни (може би неправилни) идеи е добър начин да помогнете на учениците да разберат истински процеса.

Можете да подготвите формуляр с въпроси като:

- ↳ Знаете ли как термостат контролира температурата в помещение?
- ↳ За какво е звуковата предупредителна система за паркиране? Как реагира водачът, когато системата излъчва звук?
- ↳ Имате ли индукционен/стъклокерамичен котлон в кухнята си? Какво означава светещ светодиод?

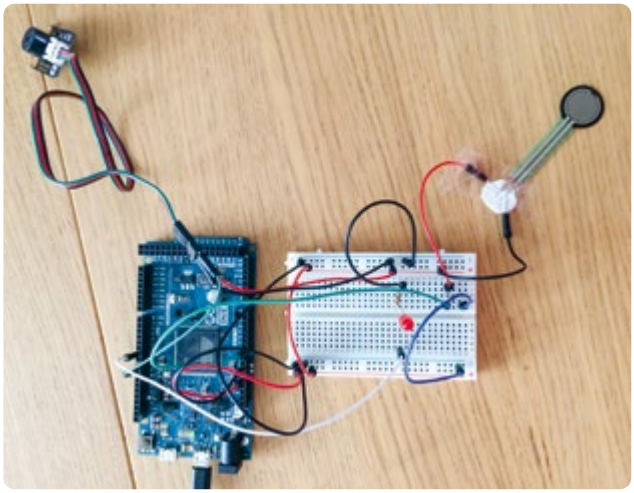
Вижте „ПОК и списък с въпроси“[4] за вариант на пълен списък с въпроси и препратки към методологията на преподаване на ПОК (Предупреждение, опит и корекция).

<Теоретична част Arduino[1]>

Преподавателят ще въведе C++ програмирането[3] със структурата и основните инструкции, така че учениците да могат да напишат обикновен цикъл, използвайки инструкциите *analogRead, digitalRead, analogWrite, digitalWrite, if…then…else, loop, while*.

Хардуерна част

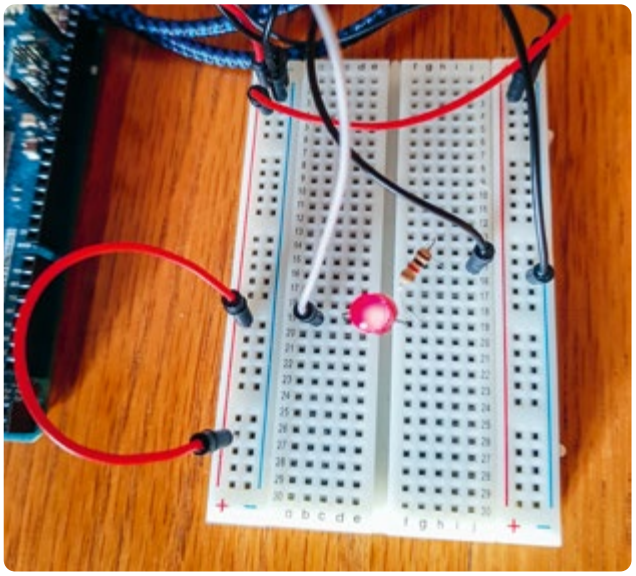
Преподавателят ще представи устройството на микроконтролера, показвайки микропроцесора, аналоговите I/O пинове (връзки) и цифровите входно / изходни пинове (връзки).



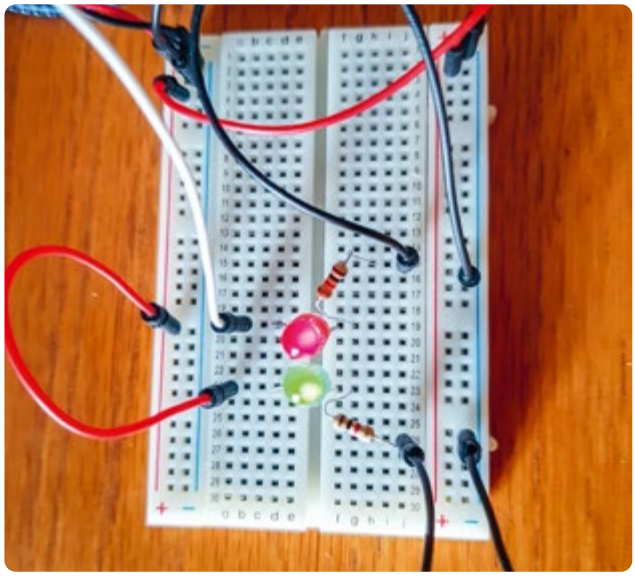
4: Arduino, прототипна платка и сензори

Учениците ще научат, че всеки сензор/задвижвано устройство обикновено има множество връзки:

- ↳ към 5V или 3,3V Arduino изход, за да получите захранване;
- ↳ към GND (заземяване) сигнал, за да може да тече ток;
- ↳ към друг цифров или аналогов входен щифт, ако се използва за четене (получаване) на външни данни;
- ↳ към друг цифров или аналогов изходен пин, ако се използва за предприемане на действия, генериращи изходен сигнал, например за излъчване на звук или светлина или ако правите нещо друго, което сигнализира за дадено действие (изписване на текст).



5: Детайл на прототипната платка



6: Прототипна платка с LED

Софтуерна част

Преподавателят ще представи проста програма, която чете датчик и изписва информацията на изпълняващото устройство, което дава ясна връзка между физическите пинове и физическия адрес на платката на микроконтролера и може лесно да се проследи от учениците.

Примери за инструкции, готови за употреба, са на разположение онлайн („Примерна програма 1“[4]).

Учениците трябва да имат предвид, че процесорът изпълнява инструкциите на кода една след друга и в реда, в който са написани. Само инструкцията 'loop' променя този принцип, тъй като казва на процесора непрекъснато да повтаря инструкциите в скоби, докато микроконтролерът се захранва.

<Практическа част с Arduino>

Учениците ще добият практически умения за работа с микроконтролер, прототипна платка и сензори. Учителят трябва да представи структурата на прототипната платка, като покаже всички налични връзки и как учениците могат да получат 5V, GND и I/O сигнали от Arduino[1] към прототипната платка. Те ще бъдат подканени да копират дадения пример за програмиране и да изпробват, тестват и отстраняват грешки от кода със свързания I/Os.

Хардуерна част

Учениците се нуждаят от микроконтролер, сензори и къси кабели (10 см), които позволяват лесни връзки между пиновете на сензора и „дупките“ на дъската. Понякога може да се наложи да запоите допълнителни кабели към сензорите, но много сензори не изискват това.

Учениците ще използват късите кабели, за да свържат 5V захранването и GND сигнал към прототипната платка, а аналоговите/цифровите пинове на Arduino[1] към някои „дупки“ на платката. Това позволява датчиците лесно да бъдат поставени и да получават необходимите електрически сигнали (виж 6).

Софтуерна част

След като проверят дали връзките между микроконтролера и прототипната платка са правилни, с точното съответствие между логически номер на пина от микроконтролера и сензорния пин върху платката, учениците ще се опитат да напишат предоставения им прост код („Програма пример 1“[4]).

Алгоритми с Arduino

Подготвили сме няколко преобразувания на сигнали от една физична форма в друга.

Преобразуване на аналогов светлинен сигнал в цифров светлинен сигнал на светодиод (модулиран с PWM функция) и звук: излъчваната звукова честота нараства с интензитета на светлината. Практическо приложение: будилник, помощ за незрящи хора. (Вижте 7)

7: Сензор за светлина

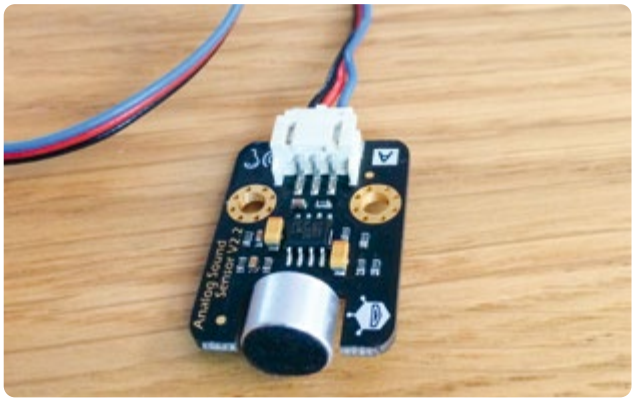
Преобразуване на сила, открита чрез сензор за сила, в цифров светлинен сигнал на светодиод (модулиран с PWM функция) и звук: интензитетът на светлината се увеличава с големината на силата. Практическо приложение: аларма за превишено тегло. (Виж 8)

8: Цифров зумер

Преобразуване на външен шумов сигнал в цифров светлинен сигнал на LED. Колкото по-висок е звукът, толкова по-висока ще бъде честотата на светлината. Практическо приложение: контрол на шумовото замърсяване.

Преобразуване на разстояние, измерено със сензор за разстояние, в звук. Техническо приложение: сензори за паркиране на автомобили. (Вижте 9)

Преобразуване на температурен сигнал в звуков и светлинен сигнал. Техническо приложение: контрол на температурата във фурната.



9: Датчик за разстояние

Преобразуване на концентрацията на влага в почвата в светлинен сигнал. Техническо приложение: напояване и контрол на напояване на растенията. (Вижте 10)

10: Сензор за влажност на почвата

Вижте „Програмен пример 2“[4] за кода, използван за създаване на тези приложения за Arduino.

Всички тези алгоритми служат, като системи за управление на сигнали и предупреждения, които наблюдават избрана среда и издават предупреждение въз основа на прочетения вход (стимул).

<Теоретична фаза с TI-Innovator Hub>

Хардуерна част

Учителят може да покаже на учениците колко лесно се свързват сензорите.

Софтуерна част

Основното програмиране може да се извърши само на калкулатора; учениците трябва да усвоят инструкциите TI-Basic, предоставени по-горе. Тогава хъбът може да бъде свързан и учениците ще се научат как да комуникират с него, т.е. да използват инструкциите „четене“ и „получаване“, за да получат данни и „настройка“, за да контролират изходите.

<Практическа фаза с TI-Innovator Hub>

Хардуерна и софтуерна част

Учениците ще започнат с основни примери, за да се запознаят с устройството, преди да работят върху собствени проблеми. Те ще се научат как да контролират различните изходи с помощта на малки упражнения, например: контролира цвета на светодиода, кара го да мига, контролира продължителността на мигането и издава звуци с дадена честота. Безкрайният цикъл отново ще бъде основната структура за продължаване на операциите за неопределено време.

Алгоритми с TI-Innovator Hub

Учениците ще решат два отворени проблема: създаване на автоматичен превключвател, който включва светлината само ако интензивността на околната светлина е по-ниска от определен праг, и създаване на будилник, който излъчва звук с все по-голяма честота с увеличаване на околната светлина. Допълнителни разработки са възможни, но ще трябва да се купят допълнителни сензори и да се включат в устройството.

<Закупуване на сензорите>

Информацията за това къде и как да закупите сензорите е достъпна онлайн.[4]

<Заключение>

В края на тези дейности забелязахме, че разбирането на нашите ученици за програмирането, общата програмна структура, както и логиката и алгоритмите се подобриха значително.

<Дейности по сътрудничество>

Сътрудничество може да бъде насърчаване чрез самостоятелното предприемачество. Учениците могат да измислят оригинален интерфейс за човешка машина (HMI). Тази HMI трябва да чете сигнали от средата (атмосфера, дом, човешко тяло и т.н.) и да реагира, като издава сигнал, който излъчва предупреждение за ситуация или извършва действие. Партньорските училища в чужбина могат да проведат пазарно проучване за търсенето и стойността на устройството в техните страни. Всяко училище, което участва в този обмен, може да измисли устройство или да направи пазарни проучвания за продукта на друго училище. В края на проекта най-популярното устройство може се произведе в малък мащаб от партньорска компания. Предприемачество е високо ценено, тъй като предлага чудесна възможност да преподаваме наука, технологии и финанси заедно.

<Препратки>

* Всички препратки са налични в английската версия.

CoALA–Програмирай малко животинче

<Автор> Мирек Ханчл
<Автор> Джулия Уинклер

<Инфо>

<Ключови думи> симулация, IPO модел (вход-процесизход), измерване, компютърно мислене, създаване

<Дисциплини> наука, биология, компютри

<Възраст на учениците> 9–13

<Хардуер> Calliope mini[1] или BBC micro:bit[2]
<Семинар А> щипки тип крокодил, червена пластмаса, USB кабели и батерия за мини Калиопа, самозалепваща се медна лента (5 мм), картон, лепило, ножица, малка чаша за вода, плакат със снимки на животни
<Семинар Б> щипки тип крокодил, USB кабели и батерия за мини Calliope, Grove сензор за влага, Grove I2C сензор за допир, Grove NFC, Grove I2C hub[3], картон, червена пластмаса, малка чаша за вода, плакат със снимки на животни.
<Език> MakeCode[4]

<Ниво на програмиране>лесно

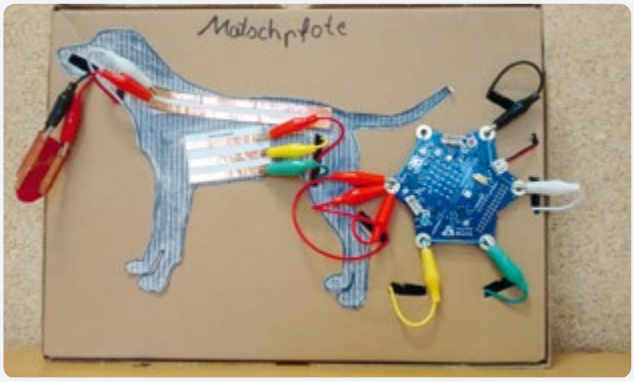
<Резюме>
Трудно ще намерите дете, което не иска да притежава домашен любимец. За да разберете кой е най-добрият, учениците ще изградят симулатор, който се управлява от едноплатков компютър и използва външни сензори, за да имитира нуждите на домашния любимец.

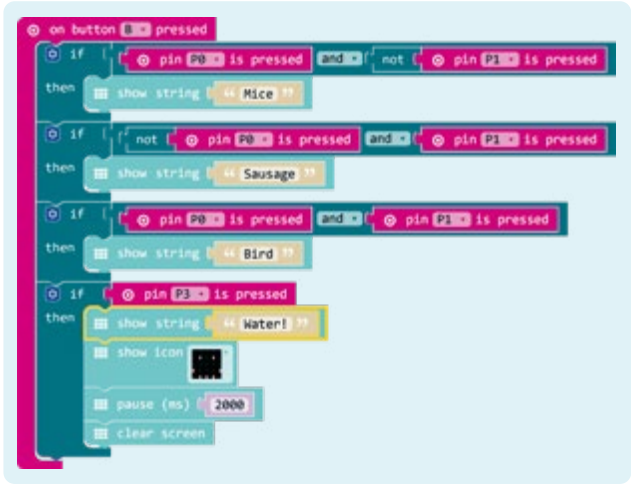
<Въведение>
Темата за "домашните любимци" е не само част от учебната програма в началните училища, но и по биология в средните училища, където учениците се учат как кучетата са произлезли от вълци; кои са основните нужди на домашния любимец; какви са изискванията, които собствениците трябва да спазват. Обикновено учениците анализират текстове в учебника или видеоклипове в интернет, защото училищата не могат да предоставят домашни любимци за тази цел. Ето защо електронен симулатор за представяне на основните нужди на домашния любимец (храна, вода, упражнения, галене и правилна телесна температура) би бил нагледен и поучителен.

Проектът CoALA не разчита на готови за употреба пособия от производители на учебни материали за масова търговия, които работят само с програми на производителя. Не се уповава и на прости играчки, като Тамагочи, които бяха световен бум през 90-те години на миналия век. Вместо това, учениците проектират, конструират и програмират свой собствен симулатор под формата на любимия си домашен любимец, включващ изображение на животното, еднобордов компютър (в нашия случай Calliope мини[1] или BBC micro:bit[2]) и консумативи като картон, медна лента и външни сензори. Учениците програмират алгоритъм за записване и оценка на основните нужди на избраното животно. В зависимост от алгоритъма, симулаторът на животни показва различни емотикони (за да покаже как се чувства животното) или свири съответно различни мелодии.

Идеята на проекта ‘CoALA–Програмирай малко животинче‘ е урокът да се проведе под формата на работилница/семинар. Отворените Образователни Ресурси (ООР) и свързаните с тях учебни материали се състоят от три части. Първата част запознава учениците с основите на алгоритмите и боравенето с еднобордов компютър. В част втора те изследват основните нужди на домашния любимец и как да ги оценят. В част трета от семинара учениците изработват любимия си домашен любимец от картон, инсталират еднобордовия компютър и подходящите сензори и създават необходимите алгоритми с помощта на графичен програмен език.

За да отговорим на изискванията на учебната програма по природни науки в началните училища и на учебната програма по биология в средните училища, предлагаме материалите на семинара в две версии. За началните училища (семинар А) се измерват и записват скоростта на хранене, пиене и взаимодействие на домашните животни, като за целта се използват проводими, самозалепващи се медни ленти. За средните училища (семинар Б) учениците използват външни сензори за измерване на влага (пиене), за допир (галене) и NFC (near-field communication) чип за безжичното комуникация (хранене). И в двете версии вградените сензори отчитат движение и температура.





Всички материали за семинарите плюс примери за използваните програми в среда MakeCode[4] са налични и могат да бъдат свалени онлайн.[5]

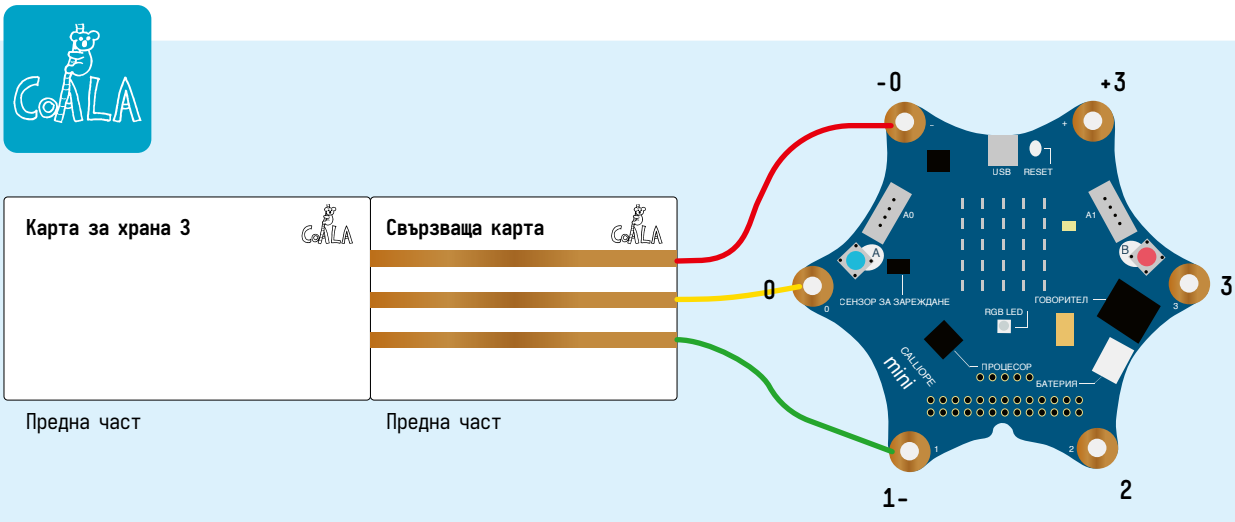
<Какво правят учениците/учителите>

За да се изградят симулатор на домашен любимец, учениците търсят изображение на любимия си домашен любимец или си правят собствена снимка. Печатното изображение е залепено върху картона и е снабдено с лепяща медна лента (семинар А) или външни сензори (семинар В) на подходящи места. Самозалепващата се медна лента или външните сензори са свързани с връзките на еднобордовия компютър, а на компютъра е написана подходяща програма, която прави животинчето "интелигентно". В следващия пример се използва основната нужда от "храна", за да се обясни по какво двете версии на семинара се различават и как се използва програмната среда.

Когато симулаторът на домашен любимец се храни, няма как да се използват реални сензори за вкус. Те са заменени със сензори за „разчитане" на предложената храна и алгоритъмът се контролира от подходящи комбинации, така че реакцията да съответства на очакваното поведение на животното. Така например симулатора на коте показва засмяно личице /емотиконче/, когато му се поднесе мишка и намръщена муцуна, когато му се поднесе кокал. Тези връзки са еднакви и за двата варианта на семинара.

Въпреки това, сензорите за храна в двете занимания са съвсем различни. При семинар А, снимки на различни видове храни са прикрепени към картонени карти. От другата страна на тези карти са залепени медни ленти, така че 'езикът' на симулатора на домашен любимец разчита двоично кодирани числа, когато картата се постави върху него. Когато връзките на четящото устройство – 'езикът', се свържат с различните пинове на компютъра, алгоритъмът може директно да провери дали пиновете са на късо, или не: различните карти с храна занулляват различни комбинации от пинове.

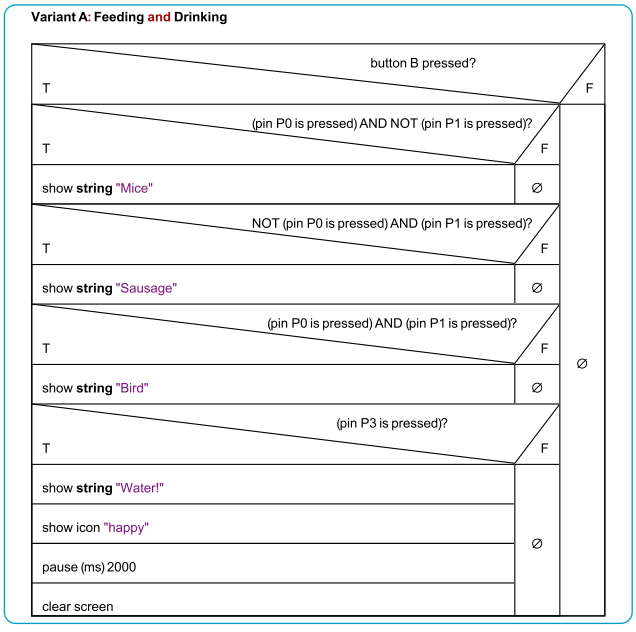
При семинар В се използва външен сензор с NFC чип и прикрепена радио антена за безжично предаване на



данни прочетени чрез NFC чипа (комуникацията между две устройства се извършва на разстояние не повече от 4 см). Този чип се закрепя на макета посредством чип карта или двойнолепящо тиксо. За разлика от семинар А, не се използва двоичен код, а вместо това се използва името на храната напр. 'риба' или 'кокал'. Това значително повишава възможните опции и съответно сложността. В алгоритъма, променливият компонент се контролира посредством сравняване на прочетената стойност с предоставените низове. В проекта CoALA, NFC чиповете се разчитат посредством приложение на смартфон. Прочетеното от NFC се свежда до единичен блок на кода в MakeCode[4] и след това се заражда като разширение /extension/ в програмната среда.

<Алгоритъм при други програмни езици>

Наличните кодове[5] могат да се въведат онлайн в MakeCode редактора[4], след което да се използват директно. При преминаване от блоков изглед / block view / към текстов изглед / text view /, кодът от източника се конвертира на JavaScript и по този начин лесно може да се използва при други програмни езици за Calliope mini[1] или за BBC micro:bit[2]. Програмните разширения за MakeCode, използвани в материалите на семинара за контролиране на мултитач / multitouch / и NFC сензорите, работят също и с BBC micro:bit.



И накрая, примерите за програмиране са предоставени на уебсайта като блок-схеми, така че алгоритмите да могат лесно да бъдат разбрани и пренесени към други платформи и програмни среди, като например Arduino.

<Заключение>

Проектът CoALA дава на учениците възможност да се запознаят с основните понятия в алгоритмите – твърдения, последователност, условно разклоняване, цикли и променливи. Те не ги учат с просто запаметяване и възпроизвеждане, а по-скоро чрез работа по вълнуващ образователен проект с реална приложимост. Учениците използват прости материали, за да построят собствен домашен любимец симулатор, който оживява с помощта на програмирането и собственото им въображение. Предоставените материали по семинара учат на компютърни умения в дидактически съкратена форма и в същото време предлагат различни нива на обучение, като по този начин отговарят на нуждите на хетерогенни учебни групи или по-големи ученици. И двата варианта на семинара може лесно да се комбинират.

Материалите за семинара са тествани успешно с Calliope mini[1] и еднобордов компютър от типа BBC micro:bit[2]. Необходима е евтина разширителна платка, за да се използват външни сензори за измерване на влага, за NFC или за мултитач в семинар Б с BBC micro:bit.[3]

<Сътрудничество>

При изпълнение на проекта CoALA се осъществи транснационално сътрудничество между две средни училища. Две групи ученици – една от Германия и една от Испания – обсъдиха опита си с домашните симулатори чрез видеоконферентна връзка. В допълнение към предложените им за решаване задачи, имената и основните нужди на техните домашни любимци, сътрудничеството се осъществи на английски език, както и на съответните им родни езици – кодиране в STEM обучението комбинирано с езиков курс.

<Препратки>

[1] https://calliope.cc/en
[2] www.microbit.co.uk/home
[3] Ако използвате BBC микро бит, ви е необходим и предпазител за микро бита.
[4] https://makecode.calliope.cc/?lang=en или https://makecode.microbit.org/?lang=en
[5] Допълнителни материали са налични на www.science-on-stage.de/coding-materials.

Данни за течностите

<Автор> Елефтерия Карагьоргу
<Автор> Севасти Цилики

<Инфо>

<Ключови думи> физика, компютри, киселинност, вода, течност, температура, pH, въвеждане на данни

<Дисциплини> химия

<Възраст на учениците> 16

<Хардуер> комплект Arduino за начинаещи[1], разширение за записване/регистриране на данни, сензор за температура, сензор за pH, SD карта

<Език> Arduino IDE – Програмиране на C[2]

<Ниво на програмиране> средно

<Продължителност на проекта> 7 учебни часа

<Резюме>

Този урок е базиран на интердисциплинарен подход, използващ програмиране на устройства и химия. Учениците поемат ролята на изследователи и провеждат експеримент, за да се определи дали има връзка между киселинността и температурата на водата. Това ще изисква използването на Arduino и химия.

<Въведение>

Тази образователна дейност е създадена, за да се демонстрира на учениците как програмирането на устройства може да се интегрира в STEM обучението и по-точно в учебния материал по химия, като се използват иновативни методи на преподаване. Тя е проведена като извънкласна дейност по време на занимания на училищните клубове по Роботика и STEM, които се събират за по два часа всяка неделя следобед.

Учениците влязоха в ролята на изследователи и получиха за задача да докажат важната роля, която температурата играе при измерването на pH. С повишаването на температурата се увеличават молекулните вибрации, което позволява на водата да йонизира и да образува повече водородни йони. Това води до спад на стойностите на pH на течността.

<Метод на преподаване>

Използване на изследователски подход в преподаването на науки: ние искахме да включим нашите ученици в проект за активно учене, въз основа на въпроси, които след това генерират допълнителни въпроси и така учениците напредват в познанието, докато правят проучвания по проекта. По този начин учениците стават изследователи и учат, като осъществяват практическа дейност.

<Предпоставки — необходими познания>

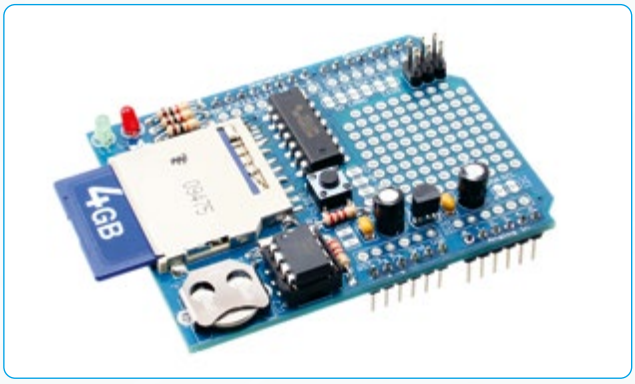
По приетата в Гърция учебна програма:

- сновни познания по програмиране, придобити през третата година на гимназиалния курс и през първата година от прогимназията;
- основни познания по киселинност и pH, придобити третата година от гимназиалния курс.

<Учебни материали / помещения>

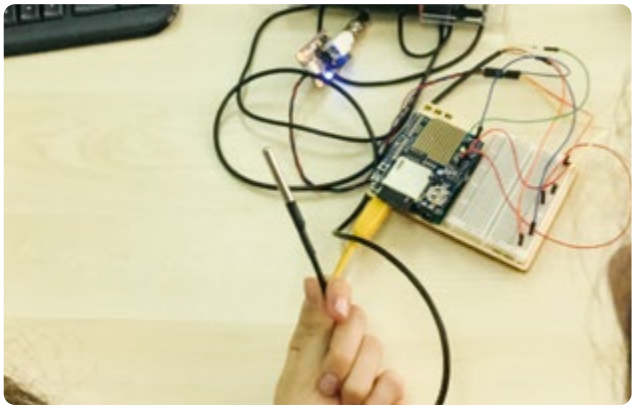
Лабораторията по Роботика и STEM в училище съдържа следния инвентар:

- Arduino[1] комплект за начинаещи (включващ Arduino, кабели, LCD екран, и т.н.);
- Adafruit Data Logger разширител за Arduino (📷1)
- Аналогов измервателен уред pH meter Pro Kit за Arduino (📷2)
- Водоустойчив сензор за температура (📷3)
- SD карта;
- Компютър с SD порт, тип лаптоп, който е необходим за програмиране и събиране и анализ на данните;
- Деминерализирана вода (дестилирана вода в която са премахнати повечето минерали и соли);
- Пакети за лед и охладител, където да се съхраняват ледените кубчета.



📷 1: Adafruit Data Logger разширител за Arduino[3]

📷 2: Аналогов pH метър за Arduino



3: Водоустойчив термосензор

<Изследователски въпрос>

Има ли връзка между киселинността и температурата на течността?

<Въпроси за разрешаване>

1. Как да свържем сензорите с Arduino плаката?
2. Как да въвеждаме и анализираме данните?

<Какво правят учениците/учителите>

<Подготвителна фаза: Въведение — теория – работа по групи>

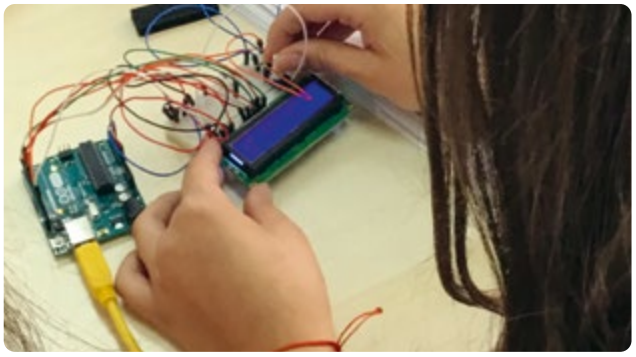
Продължителност: 1 час

Учениците ще бъдат разделени на групи и ще получат кратко въведение за Arduino[1], използваните сензори (pH сензор и температурен сензор), както и за начина им на работа. След това ще обсъдят теорията за киселинността, pH-метъра и връзката между киселинността и температурата. На учениците ще бъде поставена задача да проектират експеримент за измерване промяната на киселинността на течността, при промяна на температурата ѝ.

<Фаза 1: Въведение в Arduino и как се пише код>

Продължителност: 1 час

Учениците ще се запознаят с елекрическите вериги в Arduino и след това ще разберат за основните правила за програмиране в Arduino. Ще се научат да свързват LCD екрана към Arduino и как чрез програмиране да изведат съобщение на него. (4 и 5).



4/5: Свързване на LCD екрана

<Фаза 2: Свързване на сензорите>

Продължителност: 1 час

Учениците ще научат как работят pH сензора (6) и температурния сензор (3), като ги свържат към Arduino[1] и ги програмират, за да представят данните на LCD екрана. Това е подготвителна фаза, за да разберат входните и изходните параметри на сензорите.

6: Сензор за pH

<Фаза 3: Разширител за регистриране на данни>

Продължителност: 2 часа

Учениците ще запоят разширителя за регистриране на данни с SD карта на Arduino[1] платката, където ще се осъществява регистрирането на данни (7). Те ще програмират разширителя за регистриране на данни, който има свой собствен часовник в реално време (RTC). Ще започнат експериментът с използване на деминерализирана вода при 25 °C (неутрална) и след това ще измерват pH и температурата, като потапят сензорите в течността за 10 секунди.

7: Data logging разширителна платка запоена към Arduino[4]

<Фаза 4: Експериментът>

Продължителност: 1 час

Учениците ще проведат тестове с проби от деминерализираната вода, всички от които са с различна температура. Те ще започнат с деминерализирана вода (при стайна температура) в купа или чаша, поставена за охлаждане във водна баня с ледени кубчета (8 и 9). На всяка 1 минута ще потапят сензорите в течността за 10 секунди. Ще повтарят процедурата най-малко 6 пъти,

за да генерират голямо количество данни за етапа на изобразяването и онагледяването им. По този начин водната баня ще охлажда течностите за експеримента внимателно и постепенно.

8: Замерване на pH и температура

9: Водна баня с лед

<Фаза 5: Резултати>

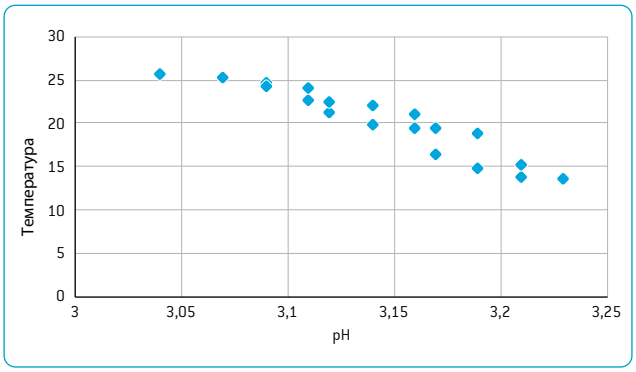
Продължителност: 1 час

Учениците ще изключат SD картата от разширителя за регистриране на данни и ще я включат в лаптопа, за да се прочете файла с данни и да се направи заключение, което ще илюстрира възможната връзка между температурата и киселинността с помощта на приложение за електронни таблици (например MS-Excel). Данните ще бъдат записани на SD картата като .csv файл, така че да може да се отваря като електронна таблица. Учителят ще обсъди резултатите от регистрирането на данни с учениците и ще прецени дали са успели да отговорят на поставените задачи. Учениците ще представят резултатите си пред класа и ще ги обсъдят със съучениците си.

<Заключение>

До края на дейността, от учениците се очаква да разберат връзките между STEM дисциплините чрез прилагане на теория от химията в програмиране на устройства за експеримент. Учениците с помощта на учителите си ще развият своето изследователско мислене и уменията за работа. Освен това в резултат на тези проектни дейности, те ще видят реално как знанията, които получават в училище, могат да се приложат практически в реалния свят. Меките умения като работа в екип, които са от съществено значение за тяхното бъдеще, също се развиват по време на работата по проекти и при намирането на решение по възникнали въпроси. Тази дейност дава и отлична възможност на учениците да подобрят работата си в STEM дисциплините и да разберат по-добре значението на такива интердисциплинарни проекти.

Експериментът може да бъде разширен като се включи голямо разнообразие от течности. Един пример е оцет в студена водна баня – с кубчета лед, така и в топла водна баня с гореща вода. (виж 10).



10: Данни за оцета

Разширението за регистриране на данни на Arduino за този проект, трябва много внимателно и прецизно да се запои за платката на Arduino. Това може да се окаже трудно за някои от учениците. Ето защо трябва внимателно да ги напътствате и дори да направите запояването сами, в случай че те не могат да се справят.

<Препратки>

[1] www.arduino.cc

[2] www.arduino.cc/en/Main/Software

[3] Снимка: oomlout (https://commons.wikimedia.org/wiki/File:ARSH-09-DL 03.jpg), „ARSH-09-DL 03“, CC BY-SA 2.0, https://creativecommons.org/licenses/by-sa/2.0/legalcode

[4] Снимка: oomlout (https://commons.wikimedia.org/wiki/File:ARSH-09-DL_(5703636953).jpg), „ARSH-09-DL (5703636953)“, CC BY-SA 2.0, https://creativecommons.org/licenses/by-sa/2.0/legalcode

Далечният капитан

<Автор> Имакулада Абад Небот
<Автор> Пери Компт Джове

<Инфо>

<Ключови думи> отдалечен контрол, 2D и 3D дизайн, моделиране, запояване на електронна платка, програмиране на чип, програмиране на приложение, 3D принтер

<Дисциплини> технологии, инженерни науки

<Възраст на учениците> 14–16

<Хардуер> Arduino[1], модул с Bluetooth, материали за построяване на модел на лодка

<Език> Arduino, ArduinoBlocks[2], AppInventor[3]

<Ниво на програмиране> средно

<Резюме>

Учениците трябва да проектират и изградят модел на лодка, която след това да навигират в басейн. След като приключат с това първоначално предизвикателство, те ще използват платка Ардуино за дистанционно управление на лодката с таблет или смартфон.

<Въведение>

Учениците ще проектират своя модел на лодка и ще се научат как да подходят към тази задача от гледна точка на инженерната наука. Проектът ще започне с анализ на различните видове лодки, като за целта се използва Интернет. След това малка група ученици ще изградят модел, който да е стабилен във водата.[4]

Тя ще работи с превключвател за обръщане, направен от учениците, който позволява да се контролират два мотора (всеки от които задвижва лодката напред и назад).

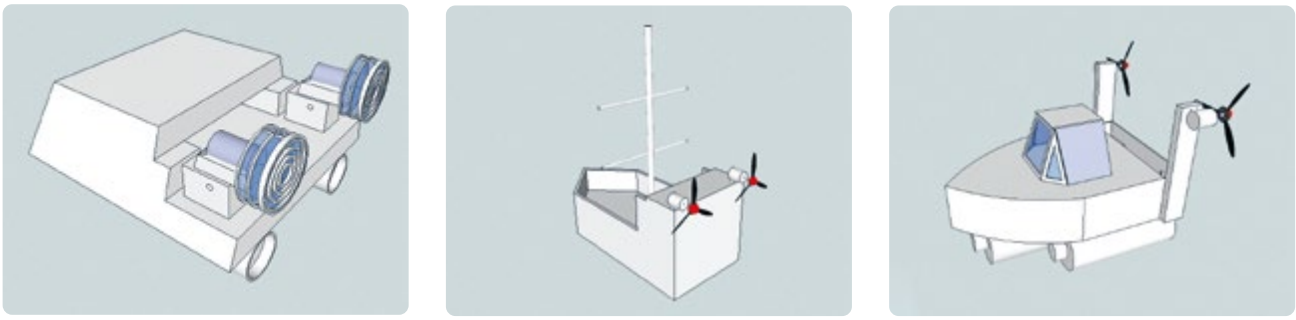
Учениците ще вградят в модела Bluetooth устройство, така че да могат да управляват лодката дистанционно със смартфон. С помощта на AppInventor[3], те ще програмират мобилно приложение за различни системи за управление, например с бутони, с гласово управление или с акселерометър (лодката ще промени посоката си според позицията на ръката на лицето, което я контролира).

И накрая, учениците ще имат възможност да представят своите проекти пред експерти на изложба за моделиране на морски кораби (📷1) и да обсъдят всички възможни недостатъци на моделите си с тях. Този експертен принос ще позволи на учениците да подобрят бъдещите си корабни модели.

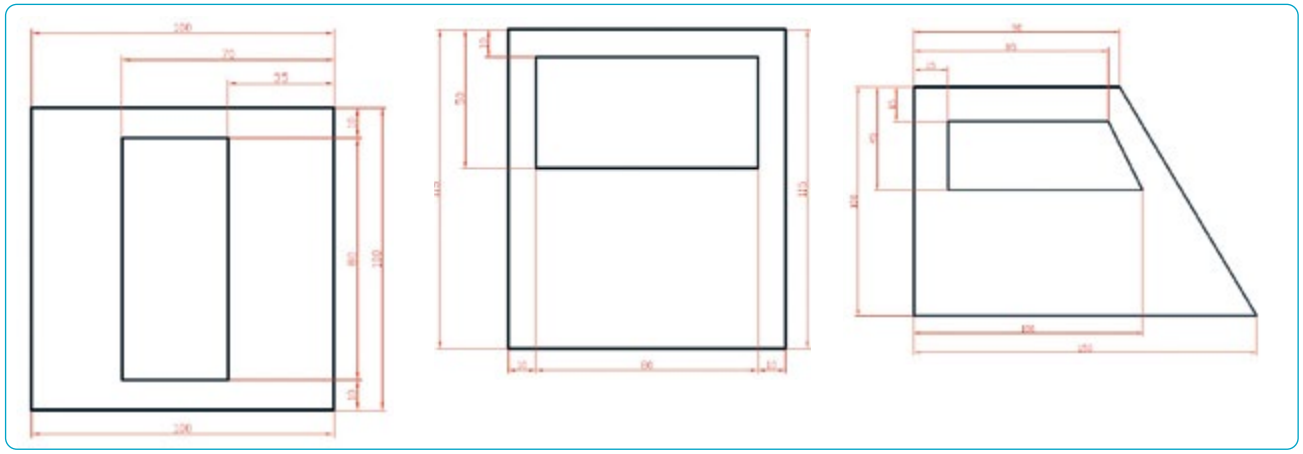
<Какво правят учениците/учителите>

Учениците ще създадат своята лодка, след като проучат различни модели и форми на лодки в Интернет. Те може

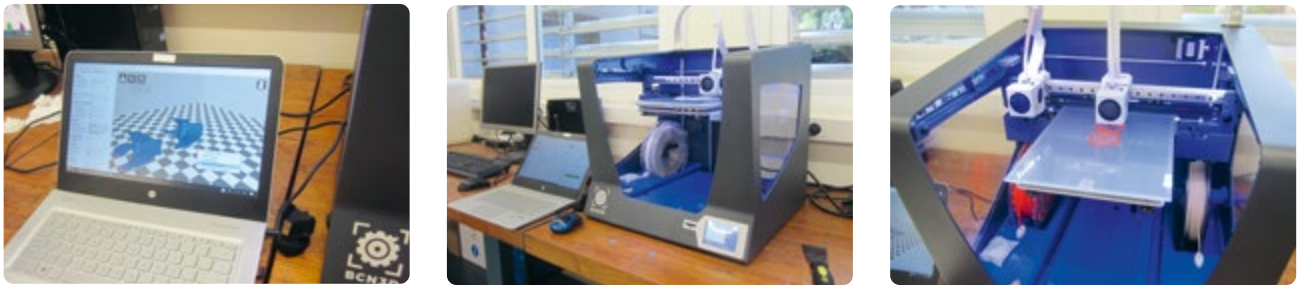
📷 1: Презентация на различни модели лодки на морско изложение в Испания



2: Различни 3D модели на лодки



3: Врата, предна и странична на кабината на лодката

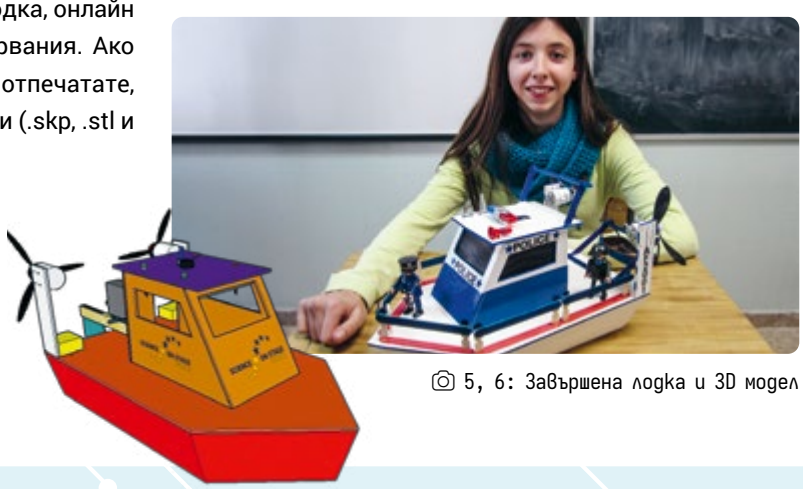


4а-с: Принтиране на витла в 3D принтиращ център Cesire Aulatec, Барселона

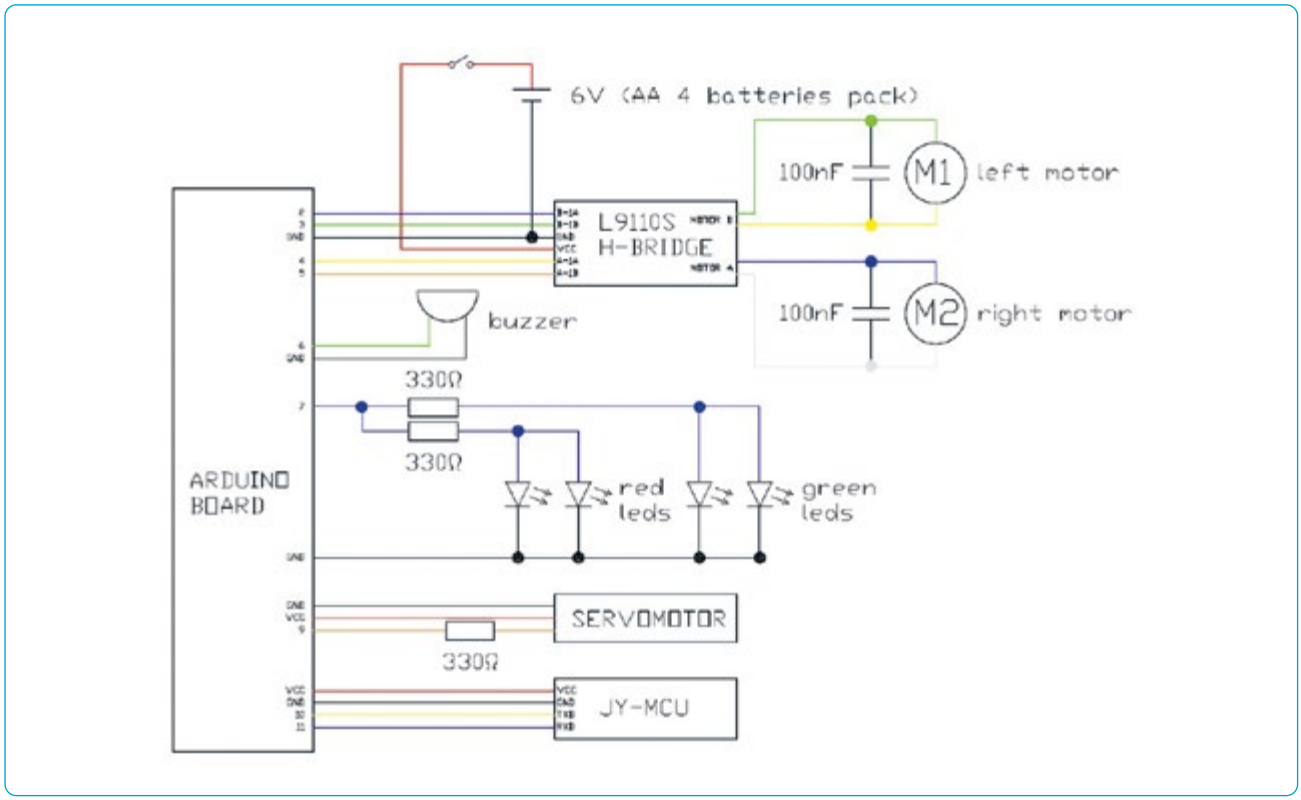
да начертаят и проектират лодката с помощта на специален 3D софтуер за дизайн, като например sketchUp[5] или Tinkercad[6]. Основното, което трябва да имат предвид е стабилността на лодката във вода, за да се избегне евентуалното и преобръщане и потъване. (2)

Предлагаме плановете за един модел на лодка, онлайн (3), включително и необходимите измервания. Ако искате да промените този дизайн или да го отпечатате, можете да го изтеглите в различни формати (.skp, .stl и .gcode).[4]

Ако в училище нямате 3D принтер, е възможно да си сътрудничите с други институции, като например университети, компютърни клубове и т.н. В нашия случай учениците отпечатаха 3D частите в 3D печатен център (4а–с). Инструкции как да се построи лодката стъпка по стъпка са налични онлайн.[4]



5, 6: Завършена лодка и 3D модел

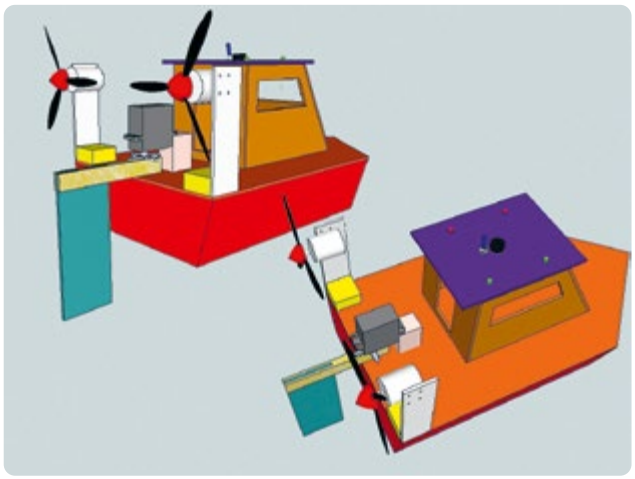


7: Диаграма на електрическите потоци

<Електрически вериги>

За да свържете аксесоарите с Arduino платката[7], използвайте схемата показана на 7 и следвайте съответните инструкции:

1. Можете да инсталирате сирена (Arduino пин 6) и светлини (Arduino пин 7) на върха на лодката. (8)
2. Можете да свържете сервомотор в задния край на лодката, за да управлявате руля (Arduino пин 9). (8)
3. Свържете Bluetooth модула в съответствие със схемата TXD (Arduino пин 10) и RXD (Arduino пин 11).
4. Свържете L9110S платката на контролера на мотора за Arduino с външните батерии. (7)



8: Лодка с аксесоари

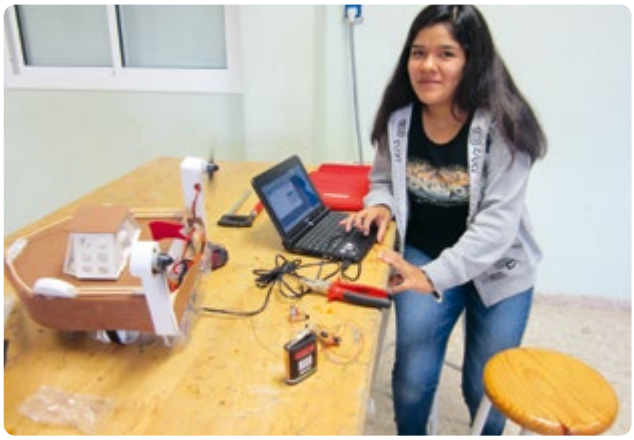
9: Сглобяване на лодката

<Как да управляваме мотора и другите компоненти на лодката>

Препоръчваме ви да програмирате с помощта на Arduino IDE[1], ArduinoBlocks[2] или друга подобна програма. Учениците могат да програмират следните задачи:

1. Захранване и прекъсване на светлините на върха на лодката.
2. Подаване на звук от сирената.
3. Контрол на серво позицията (40° дясно, 20° дясно, по средата, 20° ляво и 40° ляво).
4. Двата мотора се включват и лодката се движи напред.

5. Двата мотора се включват и лодката се движи назад.
6. Лодката трябва да завие надясно (левият мотор се включва напред, а десният мотор се включва назад).
7. Лодката трябва да завие наляво (десният мотор се включва напред, а левият мотор се включва назад).
8. Контролиране на всички програми посредством Bluetooth.



10: Програмиране на лодките

<Програмиране на приложението за контрол на лодката със смартфон с помощта на AppInventor[3]>

1. Програмиране на приложението да използва Bluetooth, за да се свързва с лодката.
2. Контрол на различните елементи на лодката с бутони.
3. Управление на руля с плъзгач.
4. Управление на лодката с помощта на акселерометъра на таблета или смартфона; накланянето на смартфона напред, назад, надясно или наляво ще доведе до задвижване на лодката в съответната посока.
5. Да се използва и опцията за гласови команди, за да се управлява лодката с команди, изречени на глас.
6. Комбинация на всички тези програми.

11: Потребителския интерфейс на приложението

<Списък с необходими материали и оборудване>

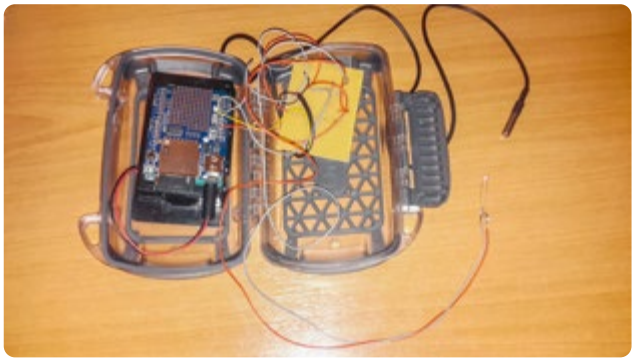
Списъкът, който съдържа всички необходими материали, може да намерите онлайн[4]. Той включва подробности като необходими количества, ценови диапазон, както и къде могат да бъдат намерени.

Материалите за изграждане на лодката струват приблизително 21 евро. Ние похарчихме приблизително 15 евро за електроника и 6 евро за останалите материали.

<Сътрудничество>

Докато редактирахме този материал, ние си сътрудничихме с Елефтерия Карагиоргу и Севасти Цилики от 7-ма гимназия в гр. Трикала, Гърция, за да реализираме Ардуино верига в хидробот, който може да се движи и да бъде направляван под вода. В този случай е много важно да се защитят моторите с восък и да се използва защитен водоустойчив корпус за Arduino платката, за да се предотврати навлизането на вода в двигателите. Ние оборудвахме хидробота „Арголит"[8] с микроконтролера Arduino UNO, за да му осигури електронен „мозък", който да записва осветеността, както и да измерва температурата под водата. „Мозъкът" също включваше разширител за регистриране на данни, който има часовник в реално време и възможност за запис на SD карта, където запазихме измерените данни.

12: Конструиране на хидробота



13: Arduino с водоустойчиво покритие

Онлайн материалът съдържа видео, снимано под вода, на „Арголит“ хидробота при тестването му в река Трикала, Гърция.[4]

<Препратки>

[1] www.arduino.cc
[2] www.arduinoblocks.com
[3] http://appinventor.mit.edu
[4] Всички стъпки от този проект и допълнителна информация: www.science-on-stage.de/coding-materials.
[5] www.sketchup.com
[6] www.tinkercad.com
[7] www.arduino.cc/en/Reference/Board
[8] Наръчник за построяване на плавателното тяло е наличен на http://seaperch.mit.edu/build.php.

Как да програмираме?

<Автор> Бернар Шриек

Тази статия разглежда използването на програмни езици в науката. Може да се нуждаете от специален хардуер и софтуер, за да използвате учебните единици посочени в този материал. Също така е полезно да имате основни познания за фундаменталните принципи на програмирането. Тази глава има за цел да ви предостави ясен преглед на необходимата информация.

<Хардуер>

Програмирането в науката се занимава основно с измерването на физични и химични величини с датчици и сензори и контрол на определени изходни данни. Типичните сензори измерват температура, шум, светлина, разстояние, pH, а бутоните се контролират с натиск, докосване и т.н. Типичните изходи са LED, зумер, говорител, мотор и др.

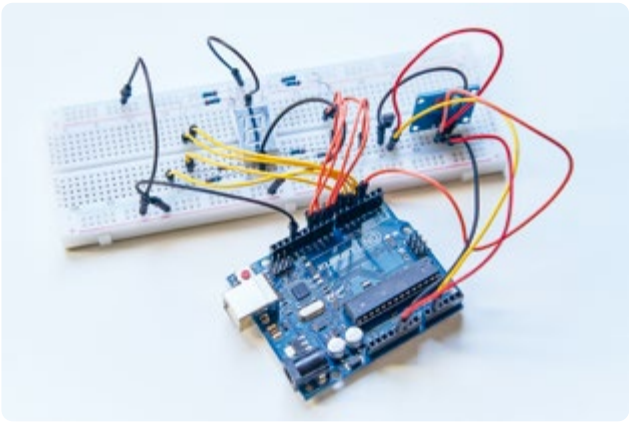
<Arduino>

Тези сензори и изходи могат да бъдат свързани към или да са вградени в микроконтролер или еднобордов микрокомпютър. Те могат да бъдат повече или по-малко сложни. Най-малката платка е Arduino[1]. Той е наличен в различни версии, но Arduino UNO се използва най-широко в училищата (а също и в тази публикация). Платката съдържа 8-битов микроконтролер, който работи доста бавно. Въпреки това, скоростта на процесора не е от значение за повечето приложения, тъй като платката комуникира единствено чрез бутон за нулиране и LED индикатор.

Платката може да бъде свързана с кабели към различни други сензори и изходи. Програмите за Arduino се пишат на компютър и се изпращат до Arduino чрез USB връзка. Когато програмата се зареди на Arduino, тя може да бъде стартирана (а по-късно да бъде прекъсната и рестартирана) чрез натискане на бутона за нулиране. Когато Arduino е свързан чрез USB, той получава захранване чрез USB връзката, но се нуждае от отделен източник на захранване, когато е изключен от компютъра.

Един много практичен начин да разширите обхвата на Arduino е чрез използването на разширители (shields), които са платки за разширяване на веригата и се включват директно в изходите на Arduino. Например често се използват платки за записване на данни, които имат часовник в реално време. Данните се записват върху SD карта, която се поставя в слота за SD карта на съответната платка.

В Интернет можете да намерите добре описани примери за програмиране за почти всяко разширение (екранни платки, сензори и изходи).



1: Arduino модул

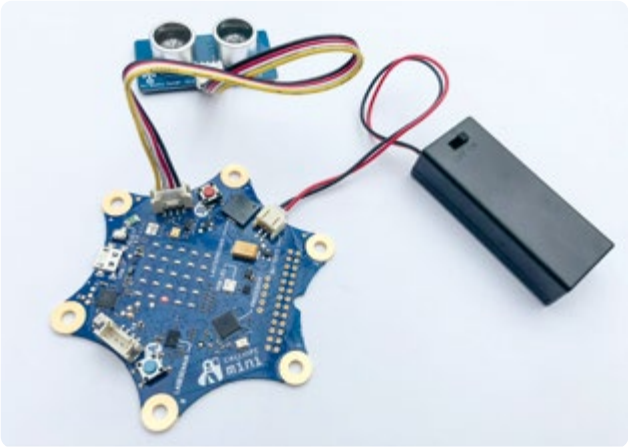
Компютри, работещи под Windows, macOS или Linux може да се използват за програмиране на Arduino. Софтуерът може да бъде свален от Интернет.[2]. Ще откриете и множество задълбочени примери за това как да се програмира на Arduino. Програмите за Arduino се наричат 'скечове/sketches'.

<Calliope mini и BBC micro:bit>

Друг еднобордов компютър е Calliope mini[3], който е съпоставим с BBC micro:bit[4]. Разликата между тях и Arduino е, че Calliope mini има множество вградени сензори и превключватели, така че за много проекти не се нуждаете от външни такива. Calliope mini има дори вграден Bluetooth за комуникация с външни устройства и смартфони. Процесорът също така е по-стабилен и по-бърз от този на Arduino, както и наличната памет е значително повече. Програми, написани на компютър, може да се прехвърлят с помощта на USB към Calliope mini. Програмите се задействат автоматично, когато бъдат прехвърлени, но може да се рестартират с помощта на бутона „reset". Както вече споменахме Calliope mini разполага с много сензори, но може да бъдат прикачвани и допълнителни с помощта на стандартни връзки[5].

Този миникомпютър има 5×5 LED матрица, която може да се използва за скролване на текстове. Calliope mini е малко по-скъп от Arduino, но предимството на множеството вградени сензори и устройства си струва вложението за всяко едно училище.

Calliope mini използва JavaScript за програмиране, но може да се използват и други програмни среди, които са по-подходящи за малки деца. На тях ще се спрем по-късно в тази статия.

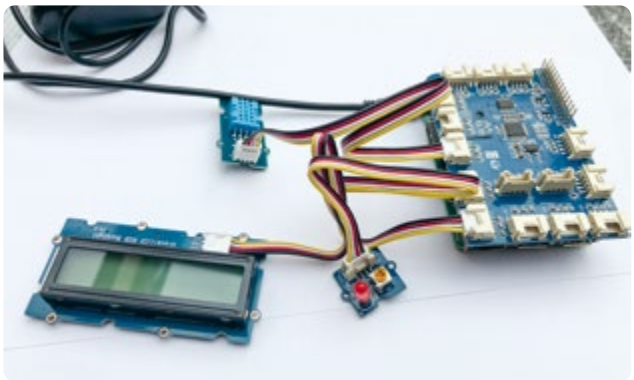


2: Calliope mini модул с ултразвуков сензор за разстояние и батерия

<Raspberry Pi>

Raspberry Pi[6] е известен, пълнофункционален еднобордови компютър, който работи под Linux и Windows като операционни системи. Може да се свърже с екран посредством HDMI кабел, а мишка и клавиатура могат да се свържат чрез USB. Цената му варира между €50 и €60, което е два пъти повече от тази на Calliope mini. Почти всеки програмен език може да се използва с Raspberry Pi. Вместо хард диск, той използва SD карта за запаметяване на програми и данни. Може да бъде надграждан с различни платки за различни цели. Снабден е с Wi-Fi устройство и може да се връзва към локални Wi-Fi мрежи. Както всички други компютри, той се нуждае от външно захранване или с батерии, или със захранващ адаптер. Въпреки че Raspberry Pi е цялостен компютър, той не е толкова бърз, колкото нормален настолен компютър или лаптоп, а и има само SD карта като външна памет. Има хиляди проекти за него в интернет. Трябва да имате опит с компютрите, за да работите с Raspberry Pi.

3: Raspberry Pi модул



4: Raspberry Pi модул с разширителна платка

<LEGO Mindstorms>

LEGO Mindstorms[7] е доста по-скъпа микрокомпютърна система. Повечето ученици имат известен опит с LEGO, така че им е по-лесно да изградят контролирани от програма инструменти и машини. LEGO има собствена система за програмиране, базирана на различни иконки,

която се нарича EV3 софтуер и е базирана на LabView[8] (професионален софтуер за измервания и контрол). Тук подреждате програмни блокове, за да направите работеща програма. Използването и програмирането на мотори е много важно, когато програмирате EV3-Робот. Освен оригиналния LEGO софтуер, може да използвате и lejOS[9], специално приложение на Java виртуална машина. Поради това, че в Eclipse съществуват много плъгини за LEGO /plug-in/, повече хора използват Eclipse като развойна среда за Java на компютър. Голям брой датчици и градивни елементи могат да бъдат свързани към главния процесор, но като всички материали на LEGO, те са много по-скъпи от сензорите и елементите за гореспоменатите микрокомпютри.

Има стотици други еднобордови компютри, много от които са проектирани да работят с роботи за всякаква цел. Ако търсите проектни идеи, ние препоръчваме да посетите hackster.io[10]. Моля, имайте предвид обаче, че трябва да се регистрирате безплатно, за да получите достъп до много описания на проекти.

<Софтуер>

През 21-ви век уменията за програмиране и писане на код ще стават все по-важни във всички области на живота. Една от целите на тази статия е да помогне на учителите да насърчават интереса на учениците си към програмирането. Учениците обикновено се интересуват от програмиране и се нуждаят само от правилните инструменти и програмни езици, за да бъдат успешни и по-мотивирани. Изборът на език за програмиране зависи от възрастта на учениците. По-малките ученици ще се нуждаят от повече визуална помощ по време на процеса на програмиране и отстраняване на грешки. Ето защо следващите редове ще очертаят начините за прилагане на няколко важни концепции за изграждане на умения за програмиране, като се използва блоково и текстово програмиране.

<Променливи>

Променливите се използват за запазване на дадени стойности, които да бъдат използвани на по-късен етап. Добър начин да възприемем по-добре променливите е да си ги представим като кутийки. Всяка кутийка съдържа някаква стойност, формата й се определя от съдържаната стойност (цяло число, флаг, низ, поредица, и т.н. вярно или невярно). Към нея също така се прикрепва и съответен етикет, съдържащ името на променливата. Предлагаме да се използват обяснителни имена за променливите, като например температура вместо буквата t, така че хората да могат по-лесно да разбират написания код. В програмните езици, ориентирани към блокове, а не текстове, като (Scratch[11], Snap![12], MakeCode[13]), променливите изглеждат като етикети и можете да видите името и стойността на дадена променлива докато програмата работи (📷5). Това е много полезно за отстраняване на грешки. Има и блокове за настройка или промяна на стойността на променлива, които са видни от само себе си.

📷 5

<Задания>

В някои текстови програмни езици променливите трябва да бъдат обявени. Декларирането им включва информация за типа на променливата (цяло число, низ, ...). Други текстово ориентирани езици чакат първото присвояване, за да решат безусловно какъв тип е променливата.

И самата задача съдържа трудности: написана, като математическо уравнение. “температура = 23” е присвояване и означава, че променливата температура получава стойност 23 (📷6). Ако искате да сравните температурата с стойността 25, пишете “температура == 25”, като се използват два знака за равенство. Тази разлика е причина за много грешки в разработката на софтуер.

📷 6

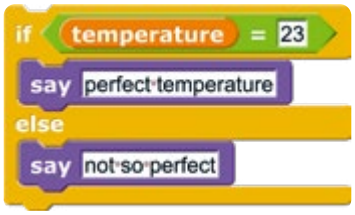
<Последователност на програмата>

Повечето компютърни програми се състоят от елементи, които се обработват последователно, т.е. един след друг. Разбира се, много езици включват паралелна обработка, при която две или повече програми се изпълняват едновременно в отделни нишки/разклонения – дори Scratch[11] или Snap![12] предлагат тази паралелна обработка.

Но ние разглеждаме единен процес. Задачите се изпълняват от първия блок или ред до последния блок или ред. Това се нарича последователност. Но нещата обикновено не са толкова прости. Искате програмата да изпълнява различни команди въз основа на решения, взети в определени точки от програмата? Това се нарича разклонение.

<Разклонения>

Има три вида разклонения: едностранни, двустранни и многостранни, и всички те се нуждаят от условие, за да посочат какво искате да направи програмата. Това условие обикновено зависи от сравняването на стойностите. Резултатът е (Boolean) истина / неистина. Едностранното разклонение само добавя няколко допълнителни операции към програмния поток, които се изпълняват, когато условието е изпълнено и е истина. Двустранното разклонение добавя два допълнителни набора от инструкции, от които се изпълнява само единият, в зависимост от резултата на условието. Многостранният клон приема променлива и в зависимост от стойността на тази променлива се изпълняват различни набори от инструкции. Блоково ориентираните езици осигуряват добра визуализация на състоянието и наборите от инструкции. Текстовите езици използват ако- (if-) или ако-тогава (if-then) инструкции за първите два вида разклонения и твърдения за случаите на многостранните разклонения.



7

8

<Цикли>

Циклите се използват в случай на многократно повтарящи се операции и са друг важен инструмент, който се използва за контролиране на последователността от инструкции. Циклите се разграничават по условието, което контролира или задава цикъла. Условието обикновено е сравнение между дадени стойности и може да бъде в началото или в края на цикъла. Ако е в началото и е неистина, самият цикъл не се осъществява. Ако условието е в края на цикъла, то цикълът ще се осъществи поне веднъж. Броящи цикли се използват, когато е ясно колко пъти трябва да бъде повторен даден цикъл.

9

10

Поредиците от условията, циклите и разклоненията обикновено са вложени. Това означава, че един цикъл може да бъде вътре в разклонение, както и разклонението може да бъде вътре в даден цикъл. Има правила за това как да се документира програмният поток в диаграма. В тази книга използваме диаграми на Наси-Шнайдерман[14], за да обясним потока от инструкции в нашите програми.

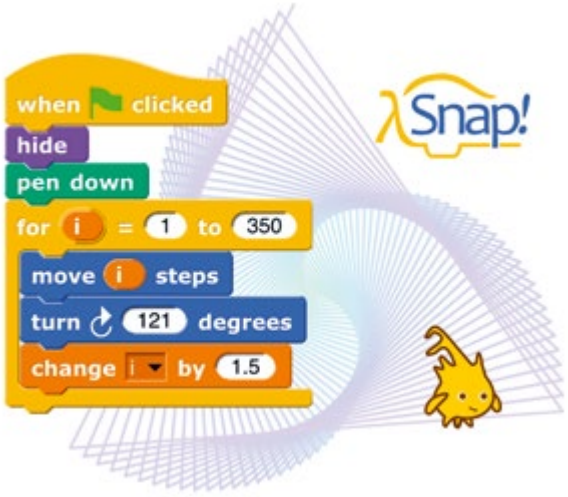
<Препратки>

[1] www.arduino.cc
[2] www.arduino.cc/en/Main/Software
[3] https://calliope.cc/en
[4] www.microbit.co.uk/home
[5] http://wiki.seeedstudio.com/Grove_System/
[6] www.raspberrypi.org
[7] www.lego.com/en-us/mindstorms
[8] https://en.wikipedia.org/wiki/LabVIEW
[9] www.lejos.org
[10] www.hackster.io
[11] https://scratch.mit.edu
[12] https://snap.berkeley.edu
[13] https://makecode.calliope.cc/?lang=en
[14] https://en.wikipedia.org/wiki/Nassi-Shneiderman_diagram

<Компютърно обучение със Snap!>

snap.berkeley.edu/run

Креативността, както и компютърната и медийната грамотност, се считат за важни умения в продължаващата цифрова революция. Snap! е инструмент, който подкрепя хора от всяка възраст или произход, да усвоят компютърните науки и активно да се включат в новия технологичен свят.



<Какво е Snap!?>

Snap! – Построй собствените си Блокове е визуален, базиран на блокове програмен език. Той е създаден така, че приканва обучаемите да дадат живот на своите идеи, докато се запознават с компютърните науки под формата на игри и експерименти. Очарователно за *Snap!* е стремежът да се осигури лесно и достъпно ниво за навлизане в езика, без да се намалява неговата изразителност. Тя позволява на начинаещи и опитни програмисти визуално, по привлекателен и разбираем начин, да задълбочат своите знания и да напреднат в понятия от компютърните науки, като например: произволни структури от данни, функции от по-висок ред и дори потребителски структури за контрол.

Snap! е разработена от SAP съвместно с изследователи от Калифорнийския университет Бъркли. Днес програмата е достъпна на над 40 езика и се използва за обучение по компютърни науки в много страни по света. *Snap!* е програмен език от отворен тип /open-source/ и е съвместим със всички съвременни уеб-браузъри.

<Знаете ли че, Snap!...>

- е един от топ 100 програмни езика в индекса на TIOBE;
- се използва за създаване на изкуствен интелект от учени в Университета Оксфорд;
- се използва от производителите на 3D-печат, бродерии и роботи;
- е лесен за преподаване, например с безплатния курс, наличен на следната уеб-страница – open.sap.com/courses/snap1 (на английски език, а скоро и на други езици).

<Запознайте се и програмирайте>

Вземете подкрепата, която ви е необходима, за да организирате и популяризирате програмирането!

„Нямах представа, че програмирането е толкова лесно!“ *„Запознайте се и програмирайте 2018“* беше истинска изненада за Филип. „Наистина искам да продължа“, каза Луиза, след като присъства на хакатон. И точно това иска да покаже инициативата – програмирането е забавно и лесно за учене.

През втората година на „Запознайте се и програмирайте“ повече от 52 000 момчета и момичета участваха в над 1100 събития в 22 държави. Както винаги, събитието се проведе по време на Седмицата на програмирането, организирана от Европейския съюз (ЕС) през октомври.

© Peter Böhmer

Всяко одобрено за финансиране събитие получи средства до 500 евро. Подкрепят се различни мероприятия, свързани с програмирането и всяка нестопанска организация може да кандидатства със свое събитие.

Инициативата възнамерява да продължи успехите от миналата година и през следващите години: многобройни събития, проекти и работни срещи ще се проведат по време на Седмицата на програмирането на ЕС в 22 държави. Целта е децата и младежите на възраст

© Dietrich Bechtel

© Dietrich Bechtel

между 8 и 24 години да опознаят света на технологиите и програмирането. Събитията показват на младите хора колко забавно може да бъде програмирането и как то може да помогне за реализирането на техните идеи. Чрез проучване на широк спектър от технологии и дигитални теми и творческо програмиране, участниците ще бъдат насърчавани да развиват цифрови умения, от които се нуждаят в днешния свят.

Мюнхенската организация Haus des Stiftens gGmbH със своя IT портал Stifterhelfen и страните – партньори от мрежата TechSoup Europe застават зад инициативата. „Запознайте се и програмирайте“ се осъществява благодарение на подкрепата на SAP.

През 2019 г. отново ще бъдат обявени награди за най-креативни идеи за събития и най-оригинални реализации. Наградите на инициативата „Запознайте се и програмирайте“ ще бъдат раздавани в най-малко три категории, включително иновации и приобщаващо образование.
Всички дейности, информация и регистрация за участие може да намерите на:

www.meet-and-code.org

Следвайте ни и се присъединете към дискусията:
@stifter_helfen @TechSoupEurope
#meetandcode #codeEU #SAP4Good

<Допълнителни материали>

Авторите са създали допълнителни ресурси и материали за учебните раздели. Можете да ги намерите безплатно онлайн на **www.science-on-stage.de/coding-materials**

<Събития по проекта в рамките на програмирането в STEM обучението>

<2 Юни 2016>
Брейнсторминг за темите по време на финалната среща по проект *„Футболът в обучението по науки“*, Брюксел, Белгия.

<20–22 Октомври 2017>
Първа работилница във връзка с финала на европейската *STEM лига* в Немския Футболен Музей, Дортмунд, Германия.

<13–15 Април 2018>
Втора работилниза в Берлин, Германия

<1 Март 2019>
Представяне на изданието във Виена, Австрия

<Март 2019–Дек 2020>
Състезание *Лига по програмиране*

<Следващи дйности 2019 и 2020>
Курсове за обучение на учители в различни европейски страни

<Пролетта на 2021>
Финал на *Code лига*

CODE LEAGUE
<Coding in STEM>

SCIENCE ON STAGE EUROPE

Науката на сцената – европейската мрежа на учителите по наука

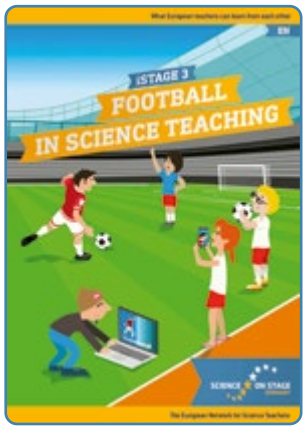
… е мрежа от и за преподаватели по науки, технологии, инженерство и математика (STEM) от всички училищни нива.
… предоставя европейска платформа за обмен на идеи за преподаване.
… подчертава значението на науката и технологиите в училищата и сред обществеността.

Федерацията на германските асоциации на работодателите в индустрията за металообработване и електротехника (GESAMTMETALL) с нейната инициатива think ING. подкрепя Наука на сцената.

Присъединете се – открийте вашата държава на
www.science-on-stage.eu
www.facebook.com/scienceonstageeurope
www.twitter.com/ScienceOnStage

Запишете се за нашия бюлетин
www.science-on-stage.eu/newsletter

<Допълнителни материали>



Футболът в обучението по науки

↳ Обучителни модули за различните аспекти на STEM във футбола
↳ Глави: Биосфера, тяло, топка, голяма информация

Смартфоните в обучението по наука

↳ Насоки и експерименти за обучение на базата на изследвания със смартфони

Къщата на Лилу – Езикови умения чрез експерименти

↳ Учениците в началното училище откриват природни научни явления в банята, хола и кухнята, докато говорят, пишат и четат.

Свалете материалите безплатно от
www.science-on-stage.eu/teachingmaterials

Проект на

SCIENCE ON STAGE GERMANY

Основна подкрепа на Наука на сцената

think ING.

Die Initiative für Ingenieurnachwuchs

С подкрепата на

SAP®

www.science-on-stage.de