Name_____

Обучение по програмиране с EdPy

Name_____

Съдържание

Name_____

Урок 1: Глава 1.1 - Запознайте се с робота Едисон

Едисон е малък програмируем робот. Едисон използва сензори както и задвижващи механизми за да се впише в света. Също така може да бъде конструиран по какъвто начин желаете чрез използване на LEGO части. Вижте снимката по-долу за да се запознаете със сензорите, бутоните и превключвателите на Едисон.

Това е изглед на робота от горе

Старт (триъгълният) бутон – Натискането на триъгълния бутон стартира робота.
Стоп (квадратният) бутон – Натискането на квадратния бутон спира робота.
Запис (кръглият) бутон – 1 натискане = изтегля програмата, 3 натискания = сканира баркод.

Това е изглед на робота от долу

Сензорът за проследяване на робота е създаден от две части: червен светлодиод и светлинен сензор.
Сензорът за проследяване може да разчита баркодове, които активират зададени преди това програми.

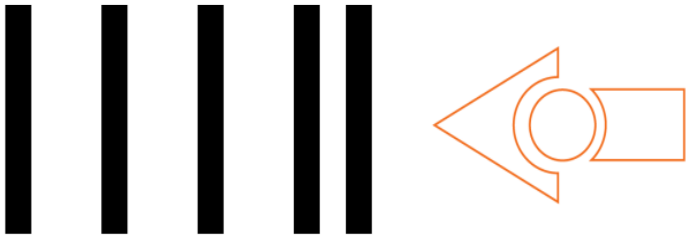
Name_____

Урок 1: Глава 1.2 – Баркод/Програмируем баркод

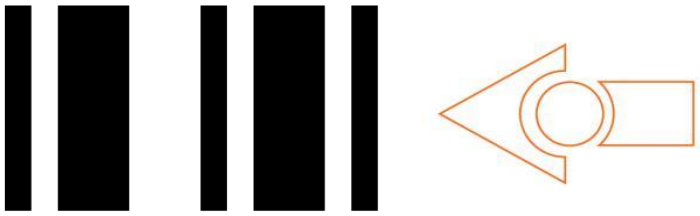
Към Едисон вече има заредени програми, който се активират чрез неговото поставяне над специални баркодове.

Разчитане на баркод.

1. Поставете Едисон с лице към баркода от дясната страна, над стрелката
2. Натиснете записващият бутон 3 пъти
3. Едисон се придвижва и сканира баркодът
4. Натиснете стартиращият бутон



Баркод – Ръкопляскащо задвижване (чрез звук)



Баркод – Избягва препятствия

Баркод – Проследява линии

Баркод – Проследява светлина

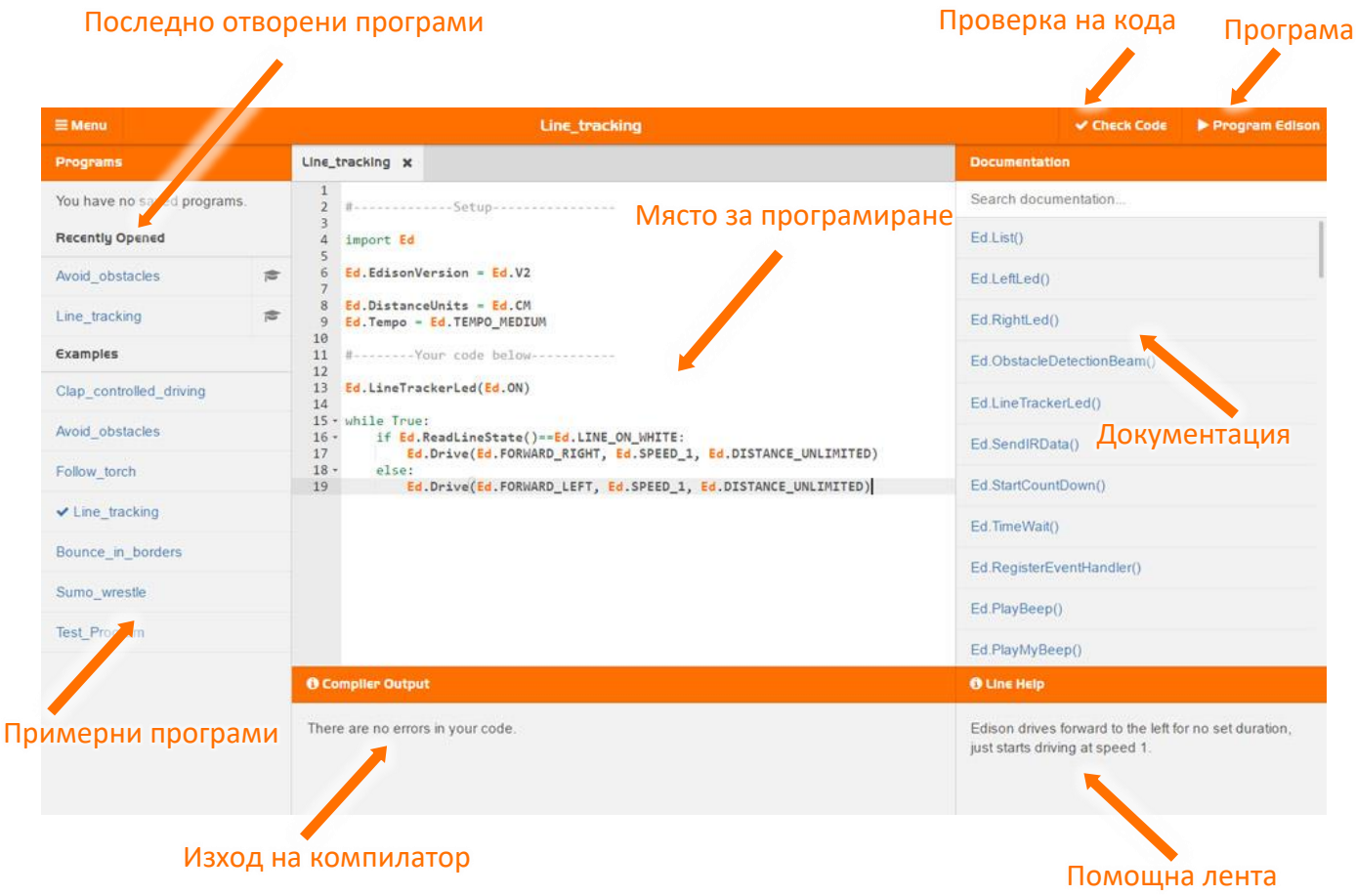
Ваш ред е:

За всеки баркод опишете действията на роботът и защо ги предприема.

Name_____

Урок 1: Глава 1.3 - Запознайте се с приложението EdPy

В тази част ще се запознаете с приложението EdPy софтуера, който използваме за да програмираме Едисон.
За да се запознаете с приложението, отворете някои примерни програми. Вижте как някои функции работят като потърсите в прозорецът – „Документация“. Всичко, от което имате нужда за да научите командите на приложението, може да намерите там. Също можете да намерите и помощ в помощната лента.



Name_____

Ваш ред е:

1. Какво трябва да промените в следният ред, ако използвате версия 1 на Едисон?
   Ed.EdisonVersion = Ed.V2 _____
2. Колко входни параметри има всяка команда?
   Ed.PlayBeep() _____
   Ed.TimeWait() _____
   Ed.LeftLed() _____
   Ed.DriveRightMotor() _____

3. Можете да промените кода за настройка, за да използвате инчове вместо сантиметри като измервателна единица за цялата програма. Как трябва да бъде записан този ред от кода за настройка?

_____
_____

4. Ако съществуват грешки във вашия код след като сте натиснали бутона за проверка на кода „Check Code“, в кой прозорец ще бъдат изписани грешките?

_____
_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 1: Глава 1.4 – Изтегляне на тестова програма

Отворете програмата с наименование „**Test_Program**” от прозореца примери „Examples” в приложението EdPy.

Така изглежда тестовата програма:

Test_Program

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.TIME
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13
14
15 while True:
16     Ed.PlayBeep()
17     Ed.LeftLed(Ed.OFF)
18     Ed.RightLed(Ed.ON)
19     Ed.Drive(Ed.SPIN_RIGHT, 5, 350)
20     Ed.TimeWait(20, Ed.TIME_MILLISECONDS)
21     Ed.PlayBeep()
22     Ed.LeftLed(Ed.ON)
23     Ed.RightLed(Ed.OFF)
24     Ed.Drive(Ed.SPIN_LEFT, 5, 350)
25     Ed.TimeWait(20, Ed.TIME_MILLISECONDS)
26
```

Едисон ще разчете всеки един ред от програмата един след друг, след това ще изпълни командата. Има редове, които Едисон ще пропусне.

Обърнете внимание на втория ред, той започва със символа наричащ се „хаштаг“ (#). Когато един ред започва с този символ той е коментиран. Едисон ще игнорира всичко след този символ и ще продължи на следващия ред. В програмирането, използваме тези „редове за коментар“ за да документираме кода си, това също ни позволява да следим какво се случва в програмата както и да помогне на други хора да разберат нейното значение.

Name_____

Добавете програмата към Едисон

За да изтеглите програма към робота, свържете кабела за програмиране към входа за слушалките на вашия компютър и усилете звука докрай. Свържете другият край на кабела в Едисон както е указано на снимката.

За да изтеглите тестова програма, следвайте следните стъпки:

1. Включете Едисон, после натиснете бутона за записване веднъж.
2. Свържете робота с компютъра използвайки кабела и се уверете, че звука е усилен докрай.
3. Натиснете бутона за програмиране на Едисон „Program Edison” в горния десен ъгъл в приложението EdPy.
4. Следвайте стъпките в излезлия прозорец, след това натиснете „Program Edison”

Щом програмата приключи свалянето, премахнете кабела. Натиснете стартиращият бутон за да задействате програмата.

Ваш ред е:

1. Какво действие извърши роботът след като натиснахте бутона за стартиране?

_____
_____
_____
_____

2. Разгледайте Python командите в програмата и помислете над това което, Едисон направи след като пуснахте програмата. Опишете как действията са свързани помежду си.

_____
_____
_____

Name_____

Урок 2: Глава 2.1 – Задвижване на робота напред

В този урок ще трябва да напишете програма, която да накара Едисон да се движи напред. Погледнете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
14
```

Стъпка 1:

Започнете да записвате указаната по горе програма в приложението EdPy като впишете “Ed” в ред 13.

Когато започнете да пишете „Ed“ в ред 13 ще видите изскачащо поле за показване. Полето за подкана показва списък от възможни команди, които да изберете.

Това е функция на приложението EdPy, наречено „завършване на командния ред“, което го прави по-бързо за програмиране.

Стъпка 2:

Напишете „Ed.Drive„ в ред 13 и изберете функцията „Ed.Drive “. Ed.Drive е функция в Python, която се импортира от модула на Edison "Ed" на библиотеката от кода за инсталиране. Функцията е част от код, който изпълнява определена роля или задача, в зависимост от параметрите, които се въвеждат. Всички функции, които са импортирани от библиотеката „Ed“, трябва да започват с „Ed.“, Което указва на коя библиотека да отиде, за да намери тази функция.

Стъпка 3:

Въведете входните параметри. Когато функцията има входни параметри, трябва да въведете стойности за всеки един от тях.

Функцията Drive има три входни параметъра:

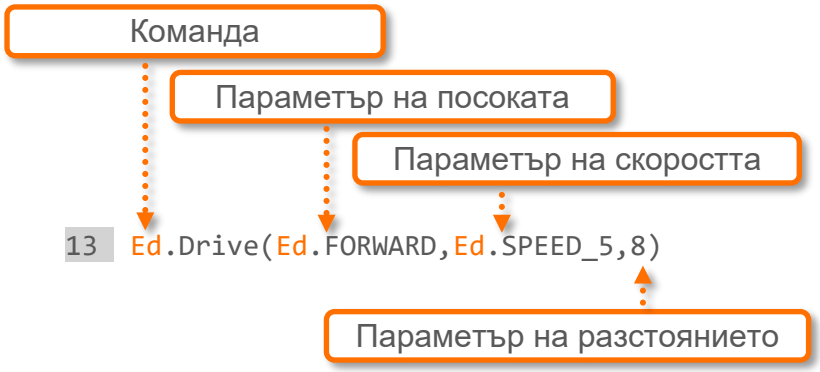
- **Посока** – Посоката в която Едисон ще се движи
- **Скорост** – Скоростта с която Едисон ще се движи
- **Разстояние** – Разстоянието което Едисон ще измине

Различните входни параметри имат различни стойности: Например за скоростта Ed.SPEED_a стойността се изменя от 1 до 10 като 10 е крайната точка.
Вида разстояние се контролира от константа Ed.DistanceUnits, която е зададена в настройващия код. Има три вида мерни единици, които може да използвате:

- Сантиметри, пише се Ed.CM
- Инчове, пише се Ed.INCH
- Време, пише се Ed.TIME

Ако имате робот Едисон V1, трябва да използвате време, затова се уверете, че Ed.DistanceUnits = Ed.TIME. Единица за разстояние е милисекунди, което означава, че да управлявате за 2 секунди, трябва да зададете на разстояние входния параметър до 2000 . (от 2000 милисекунди = 2 секунди).

Стъпка 4: Прегледайте програмата
Нека погледнем подробно на функциите за управление и техните параметри.



След като сте написали пълната програма, натиснете „Провери код“ и проверете прозореца „Изход на компилатор“ за да се уверите, че не сте направили грешки докато сте писали.
Грешките при въвеждане се наричат синтактични грешки. Ако напишете дума, която не е част от езика на програмата EdPy, също се нарича синтактична грешка, тогава компилатора на EdPy не може да разбере командата. Това създава синтактични грешки.
Ако имате грешки в кода, поправете ги така, че кода ви да съвпада с този посочен в примера.

Стъпка 5: Изтегляне и тестване на програмата
Щом проверите за грешки кода си, изтеглете програмата към Едисон.

1. Включете Едисон, след това натиснете бутона за запис веднъж
2. Свържете роботът с компютъра използвайки кабела и се уверете, че звука е усилен докрай.
3. Натиснете бутона за програмиране на Едисон „Program Edison” в горния десен ъгъл в приложението EdPy.
4. Следвайте стъпките в излезлия прозорец, след това натиснете „Program Edison”

Name_____

След като програмата се свали, изключете кабела. Използвайте указаното в урок 2 глава 1 или цветна лента за да маркирате началото и края върху маса като ваша зона за тестване. Натиснете старт бутона веднъж за да задействате програмата и вижте какво ще се случи.

Стъпка 6: Експериментирайте с програмата
Измерете разстоянието между старта и края. Опитайте се да подобрите вашата програма като накарате Едисон да спре точно преди крайната точка. Експериментирайте, за да видите кое как работи.

Ваш ред е:

1. Каква константа зададохте във вашият настройващ код ‘Ed.DistanceUnits’?

   _____

2. Каква единица ви бе необходима да въведете като разстояние, за да накарате Едисон да се придвижи от начало до край?

   _____

3. Експериментирайте с управлението на Едисон в различни скорости. Какво прави роботът когато му зададете максимална скорост 10? Забелязвате ли някакви промени в точността на Едисон докато се придвижва на максимална скорост?

_____
_____
_____
_____

Name_____

Урок 2: Глава 2.2 – Задвижване на робота назад

В тази дейност, вие ще напишете програма, която да придвижи Едисон назад.
Когато пишете програма за Едисон в EdPy, винаги следвайте основните стъпки:

- **Стъпка 1:** Проверете кода за инсталация като използвате желаните константи.
- **Стъпка 2:** Напишете програмата избирайки функциите, които искате да използвате, и попълнете входните параметри с каквито стойности желаете.
- **Стъпка 3:** Проверете програмата си за грешки чрез опцията „Check Code”
- **Стъпка 4:** Свалете и тествайте програмата на вашия робот.

Помнете, ако имате Edison V1 робот, уверете се че, Ed.DistanceUnits = Ed.TIME. Параметърът за Ed.TIME е в милисекунди.

Ваш ред:

Задача 1: Движете назад
Напишете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.Drive(Ed.BACKWARD,Ed.SPEED_5,8)
14
```

Използвайте указаното в урок 2.1 за да маркирате старта и края на линиите, по които Едисон ще се движи. Експериментирайте за да видите дали може да задвижите Едисон назад от начало до край.

Задача 2: Използвайте константата: ‘**Ed.DISTANCE_UNLIMITED’**
Има няколко начина да програмирате Едисон да се движи напред и назад.
Един начин е да използвате константата ‘**Ed.DISTANCE_UNLIMITED’** за определение на разстояние. Тази константа включва и двата мотора на задвижване на Едисон.
За разлика когато използвате точна стойност за параметър на разстояние, константата ‘**Ed.DISTANCE_UNLIMITED’** не определя точен параметър, за който моторните функции да прекратят работа. Вместо това ги включва и след това продължава към следващия ред на кода. Стопирането на моторите ще бъде необходимо по нататък в кода.

Name_____

Използвайки константата **Ed.DISTANCE_UNLIMITED** може да се окаже полезно когато искате да напишете програма, в която моторите прекратяват работа когато се намеси външен фактор, например ако е открито препятствие на пътя. Вижте следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.Drive(Ed.BACKWARD, Ed.SPEED_6, Ed.DISTANCE_UNLIMITED)
14 Ed.TimeWait(200, Ed.TIME_MILLISECONDS)
15 Ed.Drive(Ed.STOP, Ed.SPEED_10, 0)
16
```

Тази програма включва моторните функции на Едисон за задвижване назад, след което изчаква 200 милисекунди преди да ги изключи.
Напишете програма използвайки константата **Ed.DISTANCE_UNLIMITED** за да задвижите робота назад. Използвайте указаното в урок 2.1 за да маркирате старта и края на пътя, по който Едисон ще се движи. Експериментирайте за да видите дали може да задвижите Едисон назад от начало до край.

1. Задайте скоростта във вашата програма на Ed.SPEED_6. Колко милисекунди ще е нужно да въведете към TimeWait() функцията за да задвижите Едисон назад от начало до край?

   _____
2. Изпробвайте различни скорости и TimeWait() входни параметри. Запишете колко бързо и колко бавно можете да накарате Едисон да се придвижи назад от начало до край?

   Най-бързо: _____ Най-бавно: _____

Name_____

Урок 2: Глава 2.3 – Напред, после назад

В този урок, трябва да напишете програма, която да движи Едисон напред, след това назад.
Помнете и следвате четирите основни стъпки за програмиране на Едисон в EdPy:

- **Стъпка 1:** Проверете кода за инсталация като използвате желаните константи.
- **Стъпка 2:** Напишете програмата, избирайки функциите които искате да използвате и попълнете входните параметри с каквито стойности желаете.
- **Стъпка 3:** Проверете програмата си за грешки чрез опцията „Check Code”
- **Стъпка 4:** Свалете и тествайте програмата на вашият робот.

Помнете, ако имате Edison V1 робот, уверете се че, Ed.DistanceUnits = Ed.TIME. Параметърът за Ed.TIME е в милисекунди.

Ваш ред е:

Задача 1: Напишете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.Drive(Ed.FORWARD,Ed.SPEED_5,8)
14 Ed.Drive(Ed.BACKWARD,Ed.SPEED_5,8)
15
```

Използвайте указаното в урок 2.1 за да маркирате старта и края на пътя, по който Едисон ще се движи.

1. Кои са правилните входни параметри за разстояние, за да може Едисон да се движи напред после назад между стартът и стоп на пътя?
   Напред _____ Назад _____

Задача 2 : Напишете програма, която кара Едисон да се движи напред после назад между линиите на началото и краят. Този път използвайте **Ed.DISTANCE_UNLIMITED** параметъра и **Ed.TIMEWAIT()** функциите. Използвайте

Name_____

указаното в урок 2.1 за да маркирате старта и края на пътя, по който Едисон ще се движи.

2. Как изглежда новата ви програма? Напишете вашия код долу в полето.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 2: Глава 2.4 – Изрази в езика Python

В този урок, ще научите важен елемент на кода, който използваме когато програмираме на Python-изразите.

Какво са изразите?

Израза е въпрос, който може да бъде оценен като „вярно “ или „невярно.“ Например: А по малко ли е от Б? или А равно ли е на Б? В кода, изразите се пишат с помощта на математически означения вместо с думи.
В кода за настройка, сте виждали да бъде използван символа равно „=“. Например **Ed.DistanceUnits = Ed.CM**. Нотацията А=Б означава че, А е със същата стойност като Б. Изразите също използват означения. Има някой основни означения в изразите на Python, които можем да използваме:

Израз	Значение
A == B	А равно ли е на В?
A != B	А различно ли е от В?
A > B	А е по-голямо ли е от В?
A >= B	А по-голямо или равно на В?
A < B	А по-малко ли е от В?
A <= B	А по-малко или равно на В?

Изразите сравняват лявата страна с дясната страна.
Може да замените А или В с всеки израз или функция, които връщат стойност. Също може да извършвате действия върху тези изрази. Например, ‘(A + 2) > B’, което означава „когато А плюс 2 е по-голямо от В?“
В кода, изразите работят в специфичен ред. Когато вашия израз съдържа математически действия или извика функция, изразът ще разреши или действието или функцията първо. После ще сравни лявата страна с дясната на израза и ще реши дали е „вярно“ или „невярно“.

За какво се използват изразите в Python?

Изразите се използват съвместно с други оператори в кода, като например цикъл „**while**“ и **„if“,** за да промени потока на кода. Тези елементи позволяват на кода да се изпълнява по различен начин, отколкото само в последователен ред от ред 1 → ред 2 → ред 3.

Name_____

Ваш ред е:

Практикувайте решаването на изрази. Първо, напишете какво означава всеки израз, след това го оценете с вярно или невярно
Ако A = 2 и B = 4 какво значат следните изрази и как се оценява всеки (вярно или невярно)

1. (A*2) == B
   Значение: _____
   Решение: _____
2. A >= B
   Значение: _____
   Решение: _____
3. (A+A) != B
   Значение: _____
   Решение: _____
4. (A-1) < (B-3)
   Значение: _____
   Решение: _____

Name_____

Урок 2: Глава 2.5 – Активиране на движението чрез бутон

В този урок, трябва да напишете програма за да задвижвате Едисон само напред чрез натискане на кръглия бутон или триъгълния. За да направим това ще използваме изрази и цикъла „**while’**“.
Вижте следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ReadKeypad()
13 while Ed.ReadKeypad() == Ed.KEYPAD_NONE:
14     pass
15 Ed.Drive(Ed.FORWARD, Ed.SPEED_6,8)
16
```

Тази програма използва цикъла „**while’**“ заедно с израз.

В езика Python, операторът за цикъл „**while’**“ повтаря твърдения или група твърдения докато даденото условие е ВЯРНО. Той тества условието, което е написано като израз, преди да изпълни тялото на цикъла. Докато израза е оценен като „ВЯРНО“, програмата повтаря командите си в цикъла. Когато израза се оцени като „НЕВЯРНО“, програмата продължава към следващият ред от кода извън цикъла.

Използване на отстъп

Python използва отстъп за групиране на изрази или команди. В Python, всички изречения с един и същ брой символи и интервали се считат за единичен блок код.
Обърнете внимание на 14 ред в програмата. Тъй като „**pass**“ е с отстъп, той е вътре в цикъла. Ред 15 в програмата обаче не е отстъп, така че линия 15 не е вътре в цикъла.

Name_____

Функции и константи в програмата

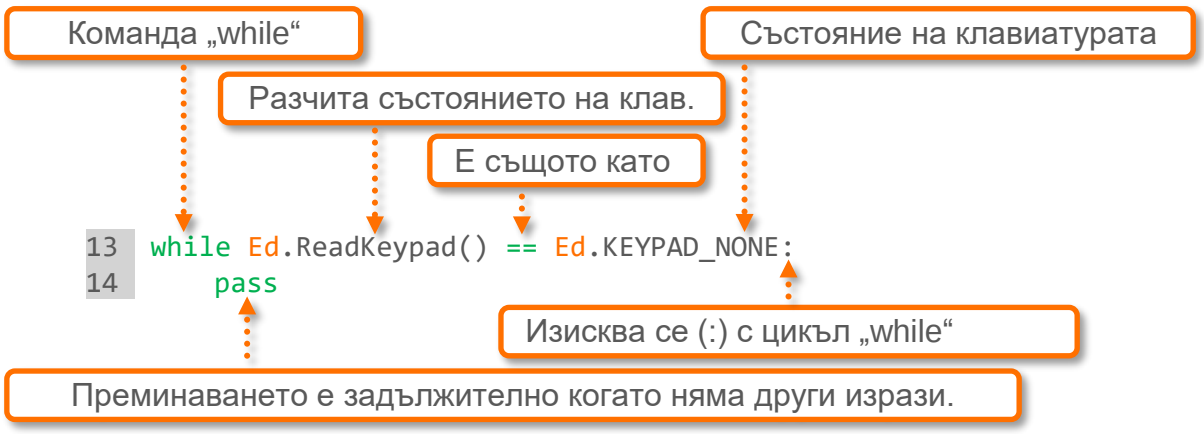
Ed.ReadKeyPad() – Тази функция разчита клавиатурата на Едисон. С други думи определя дали някой от бутоните на Едисон е натиснат или не. **Ed.ReadKeyPad()** връща стойност показваща кой бутон е натиснат: **Ed.KEYPAD_NONE, Ed.KEYPAD_TRIANGLE** или **Ed.KEYPAD_ROUND.** Тази функция не работи за квадратния бутон. Това е защото той е предназначен за спиране на програмата. Бутонът винаги ще спре програмата когато е натиснат.

Важно: използване на функцията „read“ в контура.

Някой видове данни временно се запазват в паметта на Едисон. Така функцията Ed.ReadKeyPad() може да разчете натискане на бутон, преди функцията да се извика в кода.
В тази програма, ние искаме да се уверим че функцията Ed.ReadKeyPad() в цикъла, ще изчака докато не бъде натиснат бутон, като не обръща внимание на натисканията на бутони, които се случват преди цикъла. Затова вписваме Ed.ReadKeyPad() в реда над цикъла (ред 12). Това ще изчисти натискания преди цикъла. Трябва да следите процеса когато използвате функцията “read” в цикъл.

Ваш ред е:

Напишете програмата.
Уверете се, че докато вписвате цикъла „while’“ ще включите двоеточие и че



правилно ще направите отстъп:

Изтеглете програмата и натиснете триъгълния бутон за да я стартирате.
Изчакайте малко, след това натиснете един от двата или триъгълния или кръглия бутон.

Name_____

1. Какво направи роботът след като натиснахте триъгълния бутон няколко секунди след стартирането на програмата?

_____
_____

2. Пуснете програмата отново и опитайте да натиснете квадратния бутон вместо триъгълния. Какво се случи? Защо се случи? Обяснете.

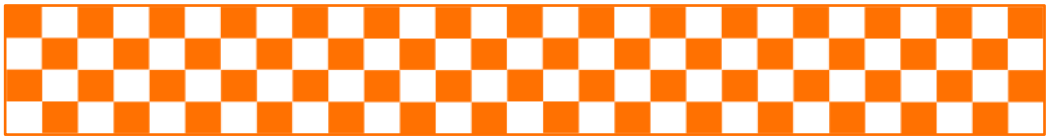
_____
_____
_____
_____
_____
_____
_____
_____

3. Сега се опитайте да добавите още код в края на програмата. Новия код който ще напишете трябва да накара Едисон да се движи назад след като натиснете или кръглия или триъгълния бутон отново. С други думи, вашата програма ще накара Едисон да се движи напред след първото натискане на бутон, след това назад когато е натиснат бутон отново. Запомнете да включите двоеточие и да отстъпите вашият код в цикъла „while“. Как изглежда новата програма? Напишете вашият код по-долу.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 2: Упражнение 2.1



FINISH LINE

START LINE

Name_____

Урок 3: Глава 3.1 – Завий надясно

В тази дейност, трябва да напишете програма, която да накара робота да завие надясно. Обърнете внимание на следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 degreesToTurn = 90
13 Ed.Drive(Ed.SPIN_RIGHT,Ed.SPEED_6,degreesToTurn)
14
```

Погледнете ред 13. Запомнете, че функцията Ed.Drive има 3 входни параметъра: посока, скорост и разстояние. В тази програма, параметъра за разстояние не е зададен в число, а в градус на завоя „**degreesToTurn“.** Това е променлива.

В Python, променливите са места в паметта зададени да запазват стойности. Това означава, че когато създадете променлива, вие запазвате някакво място в паметта на програмата.

Променлива представлява стойност, която е зададена някъде във вашата програма.

Обърнете внимание на ред 12 в програмата. Там се задава стойността на функцията „**degreesToTurn“.** Това се нарича назначаване на стойност към променлива. В Python, знака за равно (=) се използва за присвояването на стойности към променливи.

Може да използвате променливи за съхраняване на стойности, които се използват на няколко места в програмата. Това може да бъде много полезно, особено ако стойността на променливата се променя. Като използвате променливи трябва само да направите промяна в един ред от кода, за да регулирате стойността навсякъде, където тази променлива се използва във вашата програма.

Name_____

Ваш ред е:

Напишете програмата, след това я приложете и стартирайте с вашия робот. Използвайте указаното в урок 3.1 или цветно тиксо за да маркиране старта и края както и ъглите, по които ще се движи Едисон.

1. Опишете какво прави роботът и защо го прави когато стартирате програмата.

_____

_____

_____

Въведете нов ред в програмата (след ред 13) и въведете следният код:

```
Ed.Drive(Ed.SPIN_LEFT,Ed.SPEED_6,degreesToTurn)
```

Отново приложете и стартирайте програмата с Едисон.

2. Опишете какво прави роботът след като сте обновили кода.

_____

_____

_____

Сега поправете програмата си така, че Едисон да се завърти на 180 градуса вдясно, след това 180 градуса наляво.

Бележка: Запомнете може да сменяте стойностите на променливата degreesToTurn

Приложете обновената програма към Едисон.

3. В кой ред или редове програмата ви се промени? Запишете обновения ред.

_____

_____

_____

Променливите имена трябва да изпълняват определени правила в Python. Например, без символи от сорта на **хаштаг (#)** те не са позволени. Опитайте се да смените името на променливата "**degreesToTurn".** Експериментирайте с различни възможни имена и използвайте опцията „**Check Code**" за да разберете кои са позволени и кои не.

4. Дайте два примера за позволени и непозволени имена, които сте открили.

_____

_____

_____

_____

Name_____

Урок 3: Глава 3.2 – Ляв завой на 180°

В този урок вие ще трябва да напишете две различни програми за вашия робот, да може да завие наляво точно 180°

Ваш ред е:

Задача 1: Завой наляво с точност 180°

Напишете програма, която ще накара Едисон да завие наляво с точност на 180°

Бележка: За начало използвайте програмата от глава 3.1

Използвайки указаното в глава 3.1 приложете и тествайте програмата. Помнете, опитвайте се да експериментирате с вашата програма. Ако Едисон не завие точно на 180°, тогава променете входните параметри и тествайте отново.

1. Какви входни параметри използвахте, за да накарате робота да завие точно на 180°? Ако сте използвали променлива, добавете каква стойност използвахте за нея.

_____
_____
_____
_____
_____

Задача 2: Използвайки функцията: **Ed.DriveRightMotor(),** завийте точно на 180°

EdPy има следната команда '**Ed.DriveRightMotor()**', която задейства само десния двигател на Едисон. Ако само десния е задействан в каква посока ще завие Едисон? Вземете робота в ръце и имитирайте какво ще се случи ако само десния двигател е задействан.

За да откриете командата **Ed.DriveRightMotor()** потърсете в менюто документация или „Documentation", в приложението EdPy и вижте как работи функцията.

След това напишете програма която кара робота да се завърти на 180° наляво използвайки **Ed.DriveRightMotor()** функцията

2. Какви са входните параметри, които използвахте, за да накарате робота да се завърти 180° наляво използвайки командата **Ed.DriveRightMotor()?**

_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 3: Глава 3.3 – Завой надясно после наляво

В този урок ще трябва да напишете програма, която да накара Едисон да се завърти щом е натиснат триъгълния бутон.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 degreesToTurn = 90
13 Ed.ReadKeypad()
14 while Ed.ReadKeypad() != Ed.KEYPAD_TRIANGLE:
15     pass
16 Ed.Drive(Ed.SPIN_RIGHT,Ed.SPEED_6,degreesToTurn)
17
```

Напишете следната програма:
Приложете и тествайте програмата използвайки указаното в глава 3.1

Ваш ред е:

Напишете програма, която ще накара робота да се завърти точно 90° надясно щом е натиснат триъгълният бутон веднъж, след това да се завърти наляво точно на 270° щом триъгълния бутон е натиснат втори път.
Помнете да добавите командата **Ed.ReadKeyPad()** в реда над всеки цикъл „while“ за да изчисти всички натискания преди цикъла.

1. Как изглежда програмата ви? Напишете своят код.

_____
_____
_____
_____
_____
_____

Name_____

Урок 3: Глава 3.4 – Мини лабиринт

В тази глава, ще напишете програма, която ще позволи на Едисон успешно да се придвижи през лабиринт.

Ваш ред е:

Напишете програма, чрез която роботът ще се придвижи през мини лабиринт на базата на глава 3.2 щом натиснете старт бутона (триъгълния).
За успешно завършване на лабиринта, вие трябва да:

- Стартирате Едисон зад старт линията
- Да спре движение след преминаването на финалната линия
- Да задържите Едисон в границите на лабиринта.

Използвайте наученото досега да напишете програма, която използва няколко функции, които позволяват на Едисон да премине през завоите на лабиринта.

Ed.Drive()	Ed.SPIN_RIGHT	Ed.FORWARD	Ed.SPIN_LEFT

Важно:

1. Опишете последователността от движения, които роботът извърши за да премине през лабиринта.

_____
_____
_____

2. Какво ви затрудни при писането на тази програма?

_____
_____
_____
_____
_____

Предизвикателство 1: надбягване

Кой ще премине най-бързо през лабиринта, без да мами?
Помнете: робота ви трябва да стартира зад стартовата линия, както да финишира след финалната линия, и да не прекрачва границите за да спечели.

3. С кой се надбягвахте? Кой спечели?

   Опонент: _____

   Победител: _____

Name_____

4. Какво беше времето на спечелилия робот, минавайки през лабиринта?
   _____

Предизвикателство 2: проектирайте своят лабиринт

Проектирайте свой лабиринт, който е по-предизвикателен и има повече завой. Напишете програма чрез, която Едисон ще премине лабиринта успешно. Или, разменете лабиринт с вашият партньор и напишете програма за да завършите техните препятствия успешно

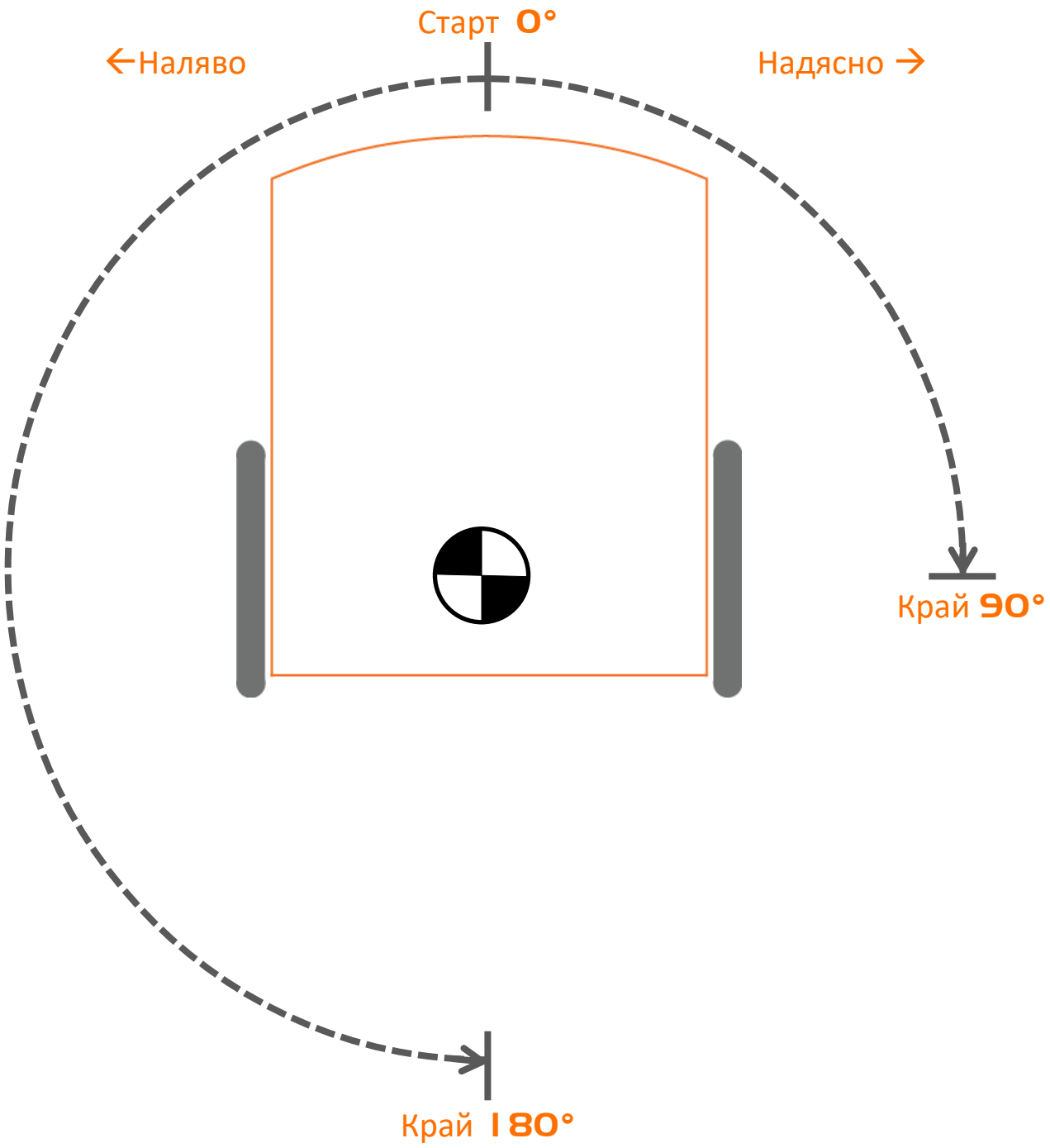
Помнете, че вашият робот трябва да стартира зад стартовата линия, както и да финишира след финалната и не трябва да излиза от граничните линии за да победи.

5. Нарисувайте малък образец на вашия лабиринт

Name_____

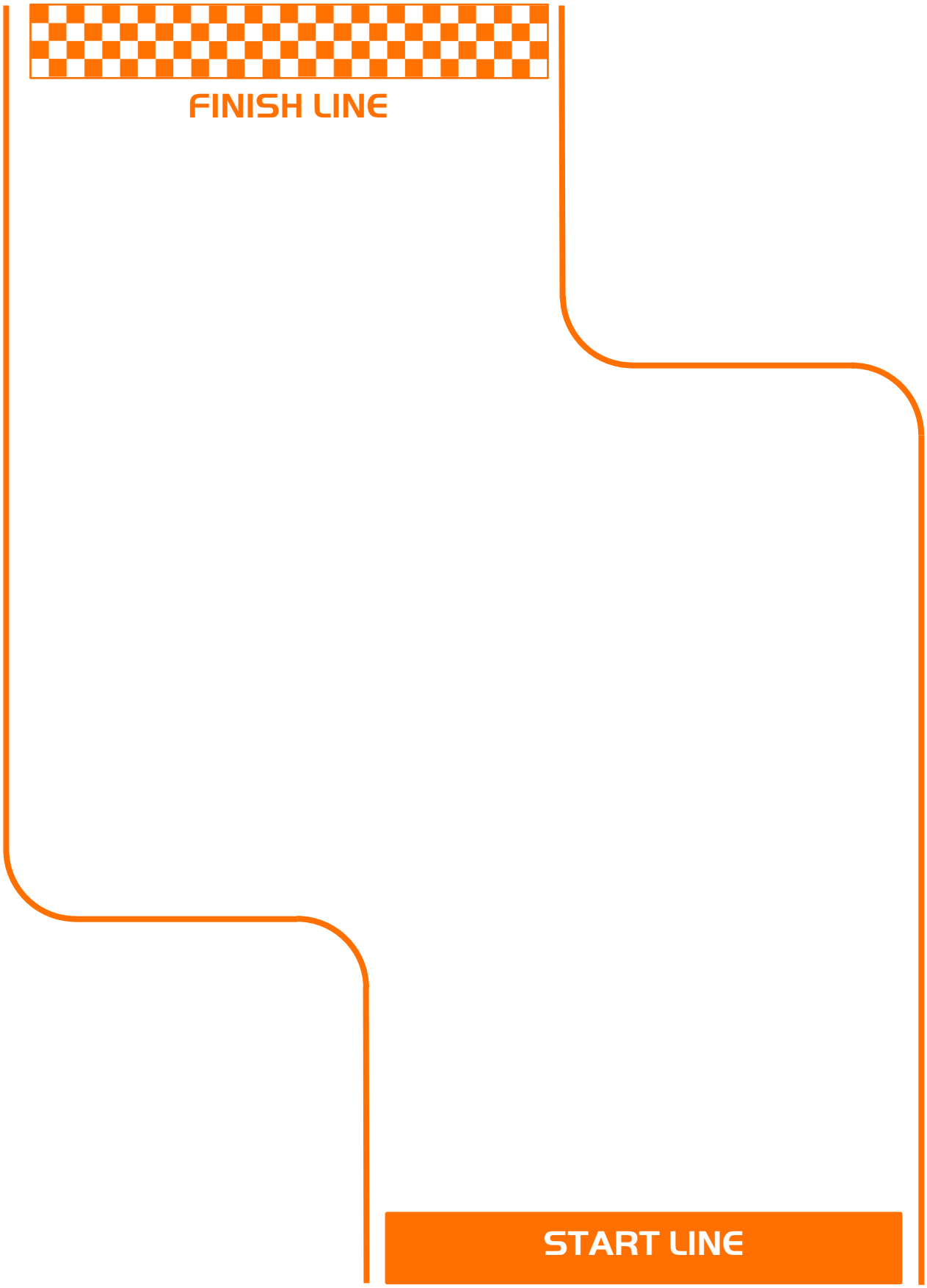
Урок 3: Глава 3.5 – Завъртане

Поставете Едисон в очертанията както е показано. Винаги стартирайте робота от стартовия маркер (0°).



Name_____

Урок 3: Глава 3.6 – Мини лабиринт



Name_____

Урок 4: Глава 4.1 – Движение в квадрат

В този урок, ще трябва да напишете програма, която ще позволи на Едисон да се движи по формата на квадрат.

Ваш ред е:

Напишете програма, така че когато Едисон се движи да е в квадрат. Използвайте командите, който сте научили досега включително командата **Ed.Drive()** и променлива. Уверете се, че програмата завършва с това, че Едисон ще приключи от мястото от което е започнал движение.

Приложете и тествайте вашата програма използвайки указаното в 4.1, и отново слагайки вашия робот на стартовата линия като следва останалите. Също така може да направите по-голям квадрат използвайки цветно тиксо, за да маркирате линиите по които ще се движи робота.

1. Как изглежда вашата програма? Напишете вашият код.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

2. Колко извикващи функции използвахте в програмата?

_____
_____
_____

3. Имахте ли повтарящи се редове в кода си? Ако да, какви са те и колко пъти ги използвахте?

_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 4: Глава 4.2 – Цикъл за движение в квадрат

В този урок трябва да напишете програма, която да накара Едисон да се движи в квадрат. Използвайки указаното в 4.1, написахте програма, която използва едни и същи команди няколко пъти. Трябваше да използвате командата **Ed.Drive ()** с параметър за посока **Ed.FORWARD** четири пъти, по един път за всяка страна на квадрата. Също така трябваше да използвате функцията **Ed.Drive()** с параметър за посока **Ed.SPIN_LEFT** отново четири пъти за да завие след всеки ъгъл.
Може би е малко отегчително да пишете едни и същи команди.
Пишейки една и също програма отново и отново не е проблем за един компютър, но по този начин не е ефективно относно вашето развитие, по-добрия вариант е да се използва цикъл.
Можем да напишем програма, която да накара Едисон да се движи в квадрат, с по-малко код използвайки „**for**“ цикъл. Това ще бъде по-полезно при писането. Като използваме по-малко редове от код, използвайки цикъла „for“, така ще намалим шансовете за правописни или синтактични грешки при писане на програмата.

Цикълът „for“ и „range()“, функцията в Python

В Python, „for“ цикъл контролира структурата, което може да се използва за да повтори команди или изрази колкото пъти желаете.
Използването на цикъл „**for**“ ви позволява да повтаряте блок от команди колкото пъти искате.
В Python цикъла „**for**“ често върви заедно с функцията „range()“.
Функцията „**range()**“ връща набор от стойности в определен диапазон.
В EdPy, **range()** има само един входен параметър. Този входен параметър определя горната граница на набора, а долната граница винаги е 0
Функцията range() връща стойност от 0 до (входен параметър - 1)
Пример:
range(4) → има 4 стойности в набора си: 0, 1, 2, 3.
Нека погледнем следният пример:

Name_____

В този пример, цикъла „for“ се повтаря четири пъти карайки променливата „i“ да има следните стойности 0, 1, 2, и 3. Всеки път когато се повтаря цикъла изпълнява блока на изразите **Ed.PlayBeep()** и **Ed.TimeWait().**
Резултата е следния, чуващия се звук се възпроизвежда четири пъти с по едносекундно забавяне между всеки звук.

Ваш ред е:

Напишете програма използвайки цикъла „**for**“ и функцията **„range()“** , така че вашия робот да се движи в квадрат. Трябва да можете да завършите програмата използвайки два пъти функцията **Ed.Drive(),** един път за напред един за завъртане.
Не забравяйте да включите двоеточие и правилно отстояние за цикъла.
Приложете и тествайте програмата си, сложете вашия робот на стартовата линия. Уверете се, че Едисон ще завърши на началната си точка. Може също да направите по-голям квадрат използвайки цветно тиксо.

1. Как изглежда програмата ви? Напишете своят код.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 4: Глава 4.3 – Движение в триъгълник и шестоъгълник

В този урок, трябва да напишете две различни програма, за да накарате Едисон да се движи във формата на триъгълник и шестоъгълник.

Ваш ред е:

Задача 1: Движение в триъгълник

Напишете програма, така че когато Едисон се движи да прави триъгълник. Отново приложете и тествайте програмата си.

1. Колко пъти вашият цикъл „**for**“, се изпълни за триъгълната ви форма?

   _____

Задача 2: Движение в шестоъгълник

Сега напишете програма, за да може робота да се движи в шестоъгълник. Приложете и тествайте програмата си.

1. Колко пъти вашият цикъл „**for**“, се изпълни за шестоъгълната ви форма?

   _____

2. Забележете модел, който се появява между броя на страните на формата и броя на изпълненията на цикълът “**for**”. Опишете го.

_____

_____

3. Коко пъти ще трябва да използвате цикълът „for“ за да начертаете обикновена 12-странична фигура (всички страни са равни)?

   _____

Name_____

Урок 4: Глава 4.4 – Предизвикателство! Движение в кръг.

Този път трябва да напишете програма, която да накара Едисон да се движи в кръг.

Ваш ред е:

Напишете програма, чрез която роботът ще се движи в кръг. Вашият Едисон трябва да се движи в кръг, не просто да се завърта на едно място.
Приложете и тествайте програмата си.
Бележка: Форма с много малки стотици страни също може да прилича на кръг.

1. Колко пъти цикъла ви се извършва, за да направите вашия кръг?

_____
_____
_____

2. Колко далеч стига вашия робот всеки път щом извършите цикъл?

_____
_____

3. Роботът ви движи ли се в перфектен кръг? Ако не, може ли да обясните защо?

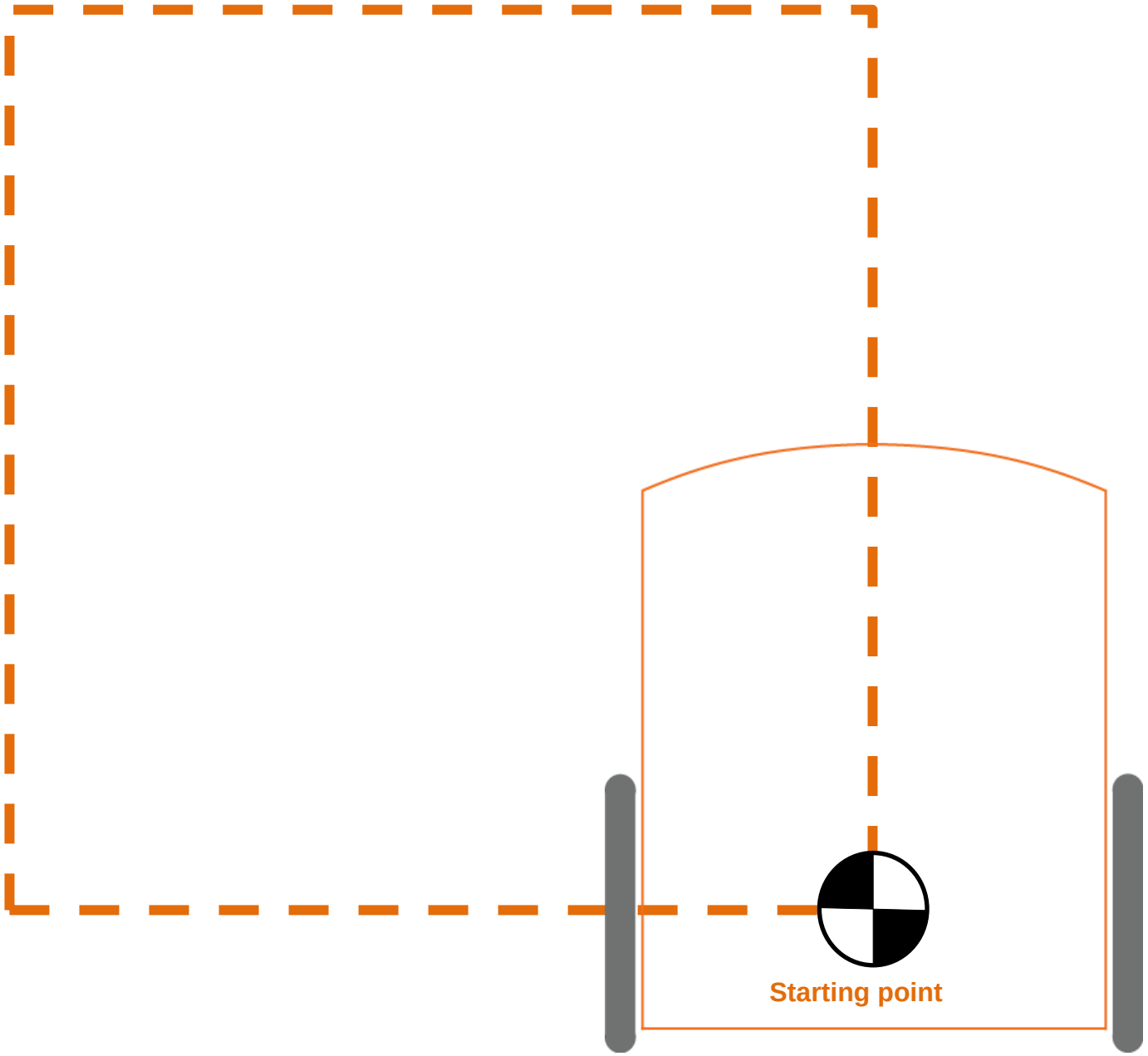
_____
_____
_____

Предизвикателство: Нарисувайте вашата форма.

Поставете към вашият робот цветен маркер използвайки LEGO блокчета. Поставете Едисон на лист хартия и изпълнете програмата си. Вижте формата, която се рисува докато Едисон се движи. Вижте дали робота ви ще нарисува цял кръг или не.

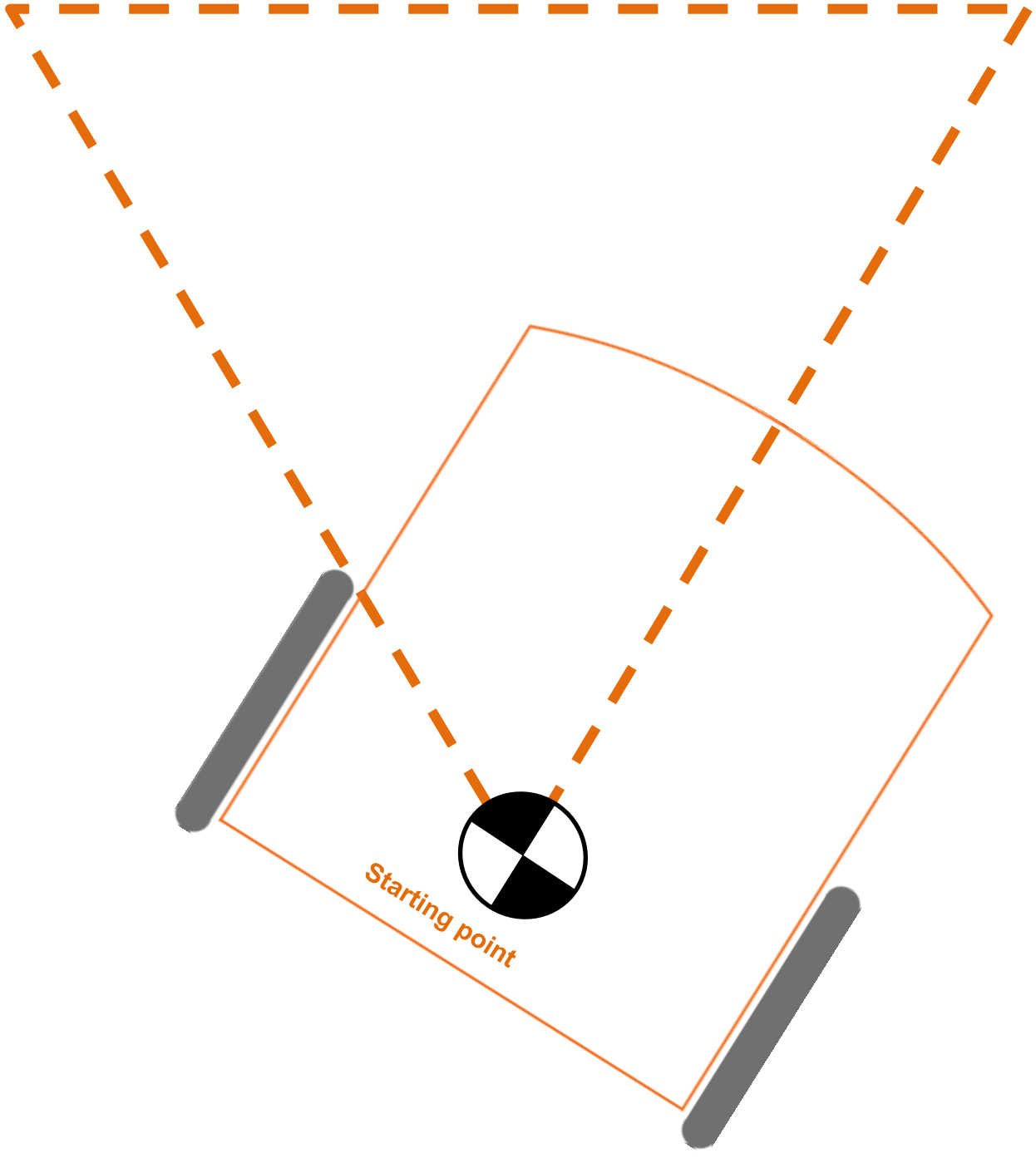
Name_____

Урок 4: Упражнение 4.1



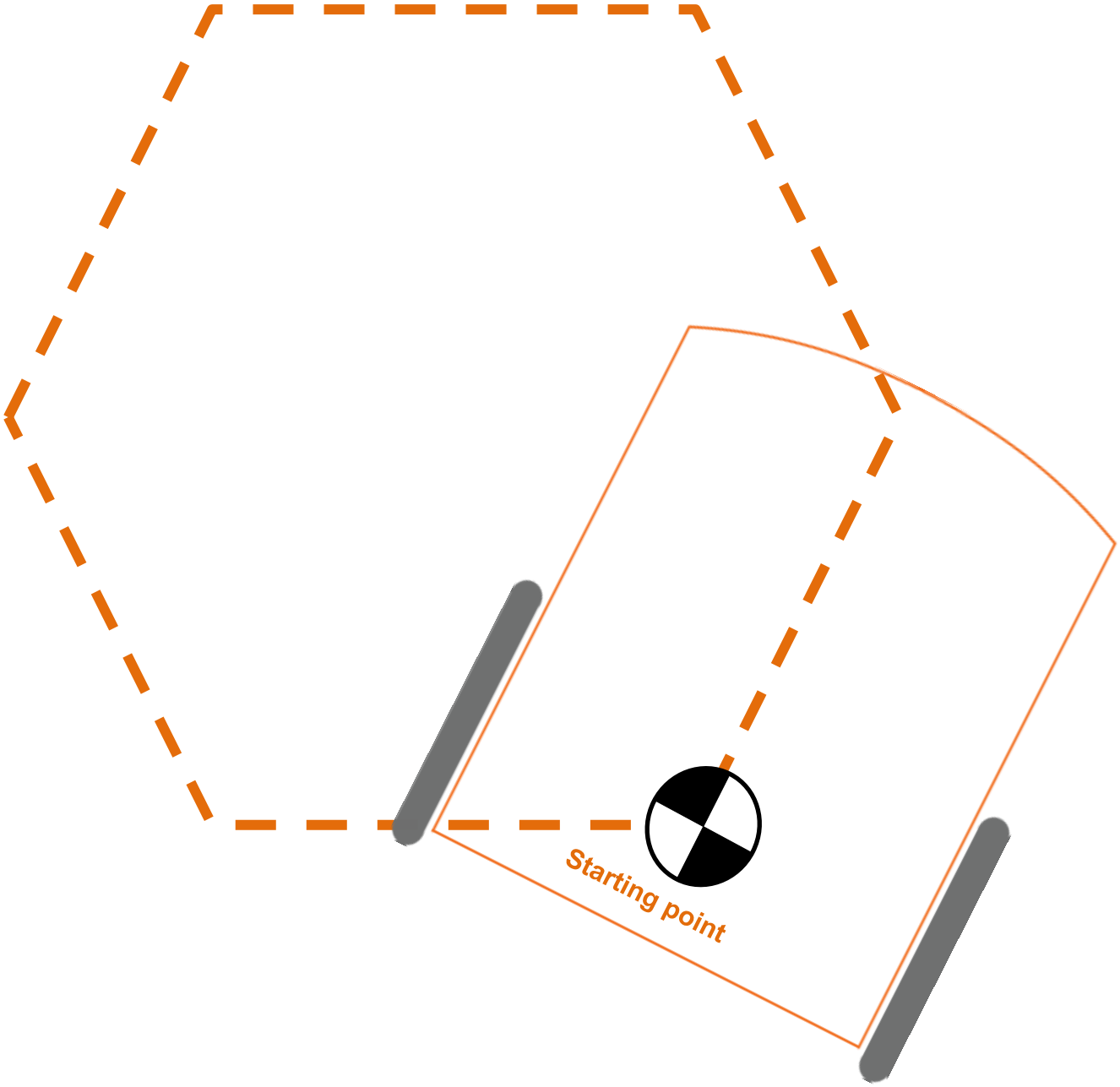
Name_____

Урок 4: Упражнение 4.2



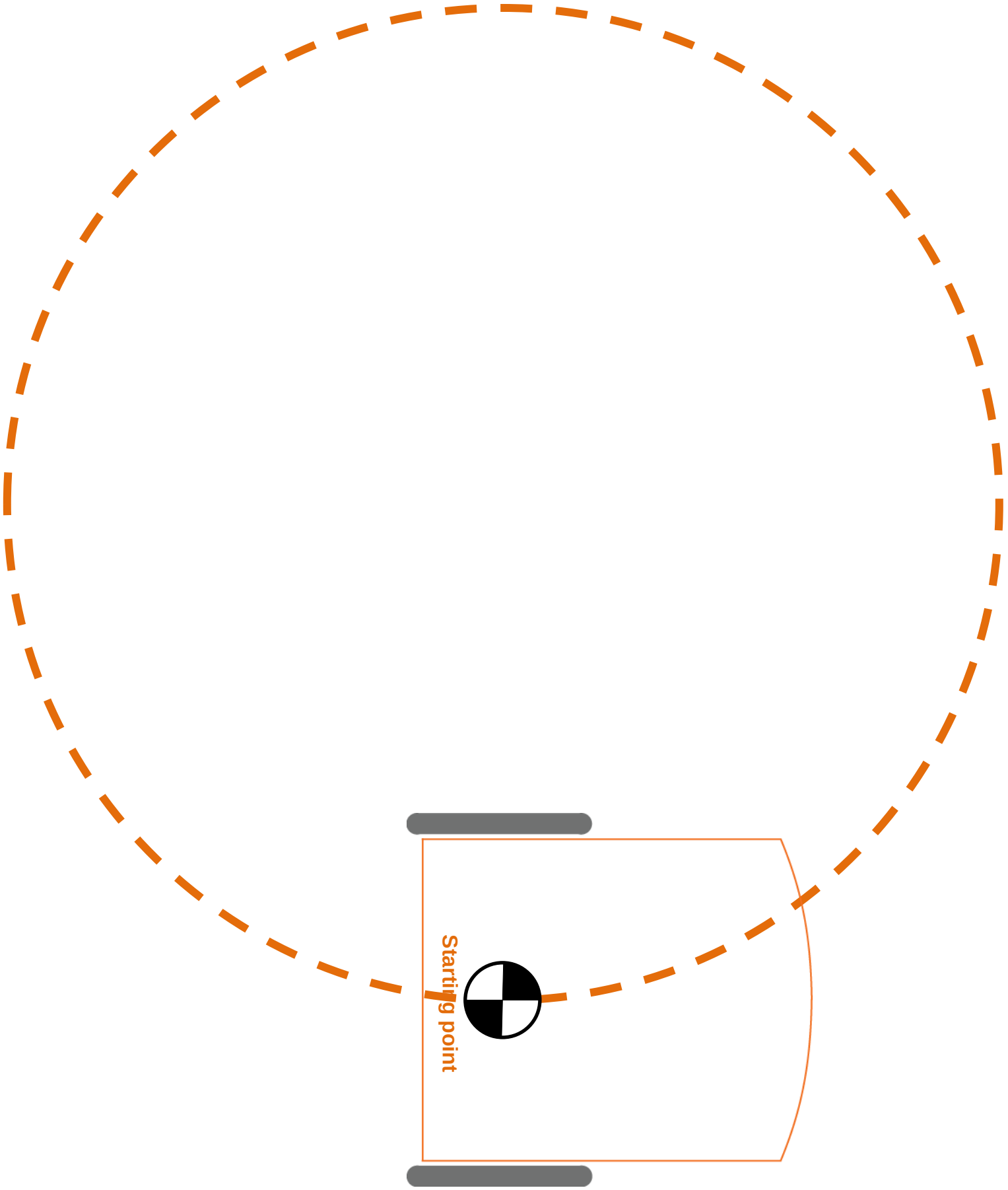
Name_____

Урок 4: Упражнение 4.3



Name_____

Урок 4: Упражнение 4.4



Name_____

Урок 5: Глава 5.1 – Възпроизвеждане на тонове

В този урок, ще трябва да напишете програма, която да накара Едисон да възпроизведе музикална нота и да научите как роботът възпроизвежда звуци.
Можете да възпроизведете индивидуални ноти чрез малкия говорител на робота използвайки функцията Ed.PlayTone() в приложението EdPy.
Функцията Ed.PlayTone() приема два входни параметъра: нотата и времетраене на възпроизвеждането. Тези параметъра включват: коя нота да бъда възпроизведена и както вече казахме за каква продължителност да бъде нейното възпроизвеждане.
Тук са указана възможните стойности за параметрите.

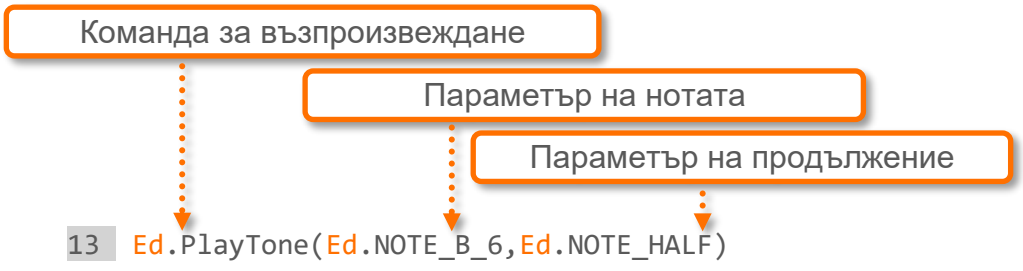
note

Parameter input options	Plays musical note
Ed.NOTE_A_6	low A
Ed.NOTE_A_SHARP_6	low A sharp
Ed.NOTE_B_6	low B
Ed.NOTE_C_7	C
Ed.NOTE_C_SHARP_7	C sharp
Ed.NOTE_D_7	D
Ed.NOTE_D_SHARP_7	D sharp
Ed.NOTE_E_7	E
Ed.NOTE_F_7	F
Ed.NOTE_F_SHARP_7	F sharp
Ed.NOTE_G_7	G
Ed.NOTE_G_SHARP_7	G sharp
Ed.NOTE_A_7	A
Ed.NOTE_A_SHARP_7	A sharp
Ed.NOTE_B_7	B
Ed.NOTE_C_8	high C
Ed.NOTE_REST	rest

duration

Parameter input options	Plays note for
Ed.NOTE_SIXTEENTH	125 milliseconds
Ed.NOTE_EIGHTH	250 milliseconds
Ed.NOTE_QUARTER	500 milliseconds
Ed.NOTE_HALF	1,000 milliseconds
Ed.NOTE_WHOLE	2,000 milliseconds

Да погледнем функцията за възпроизвеждане в програмата:



Name_____

Използвайки таблиците със стойностите, можете ли да отгатнете какво ще направи програмата? - Тази програма ще възпроизведе нисък тон за една секунда.

Ваш ред е:

Задача 1: Възпроизведете нота

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.PlayTone(Ed.NOTE_A_SHARP_7,Ed.NOTE_HALF)
14
```

Напишете следната програма:

Приложете програмата, за да чуете как звучи

Задача 2: Възпроизвеждане на нота, след това придвижване в посока? Или възпроизвеждане на звук докато робота се придвижва?

Когато Едисон издава звуци, ги прави на заден план. Тоест в момента, в който започне да издава звук, програмата преминава на следващия ред от кода. Звуците ще продължават да се възпроизвеждат на заден план, докато Едисон продължава напред с програмата.

Ако искате Едисон да изчака приключването на звукът, трябва да използвате функцията Ed.ReadMusicEnd() в цикъл „while“.

Напишете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.PlayTone(Ed.NOTE_C_8, Ed.NOTE_WHOLE)
13 while Ed.ReadMusicEnd()==Ed.MUSIC_NOT_FINISHED:
14     pass
15 Ed.Drive(Ed.FORWARD, Ed.SPEED_6, 5)
16
```

Name_____

Приложете програмата.

1. Опишете какво се случи след като стартирахте програмата.

_____

_____

_____

2. Обърнете внимание на ред 13 и 14 в програмата. Спомнете си, че изразите лявата страна с дясната в нотацията на израза. Какво върши цикъла?

_____

_____

_____

Напишете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.PlayTone(Ed.NOTE_C_8, Ed.NOTE_WHOLE)
13 Ed.Drive(Ed.FORWARD, Ed.SPEED_6, 5)
14
```

Приложете програмата.

3. Опишете какво се случи след като я стартирахте.

_____

_____

_____

4. Защо тази програма действа по различен начин за разлика от предишната?

_____

_____

Name_____

Урок 5: Глава 5.2 – Направете аларма

В този урок, вие трябва да напишете програма, с която Едисон да възпроизвежда аларма на каквато честота зададете.
Използвайки функцията **Ed.PlayTone(),** вие можете да променяте точната честота на звука, който се възпроизвежда от говорителя на Едисон използвайки числа и променливи.

Честота в акустиката

Както знаете, звука преминава чрез вълни, които се наричат звукови вълни. Акустиката, е раздел от физиката, който се занимава със звук и звукови вълни, като обхваща всичко свързано със звука, включително и как да го измерва.
Един начин за измерване на звук е чрез измерване на честотата. Честотата се съставя от броят на вълните, които преминават през определена точка за единица време.
Честотата най-често се измерва в цикли за секунда. Основната единица за честота е херц, съкратено (Hz).
Един херц е равен на една пълна вълна за секунда.

Знаете ли, че? Човешкият звуков обхват е: 20 Hz ~ 20000 Hz.

Честота и период

В допълнение към музикалните ноти, които са предефинирани в EdPy, също можем да програмираме Едисон да издава звук на различни честоти.
За целта, трябва да превърнем честоти в периоди, които Едисон да разчете.
Един период е колко дълго ще отнеме на акустична вълна да завърши пълен цикъл. След като използваме херцове, ние мерим честота в цикъл на секунда.
В акустиката, когато един период се увеличава, честотата намалява.
Нека погледнем няколко примера на това как честотата и периода се отнасят:

- Ако една вълна има период от 0.5 секунди, то тогава има честота от 2Hz защото може да завърши 2 цикъла за 1 секунда.
- Ако една вълна има период от 2 секунди, то тогава има честота от 0.5Hz защото може да завърши половин цикъл за 1 секунда.

Превръщане на честота в период за вашата програма.

За да може Едисон да възпроизведе персонализирана честота, трябва да изведем стойността на периода. Това е стойността, която вписваме в “note” параметъра в Ed.PlayTone().
За да превърнем честота в период, разделете числото 8, 000, 000 на желаната честота. Например, за да възпроизведем 1kHz (1000 цикъла в секунда):

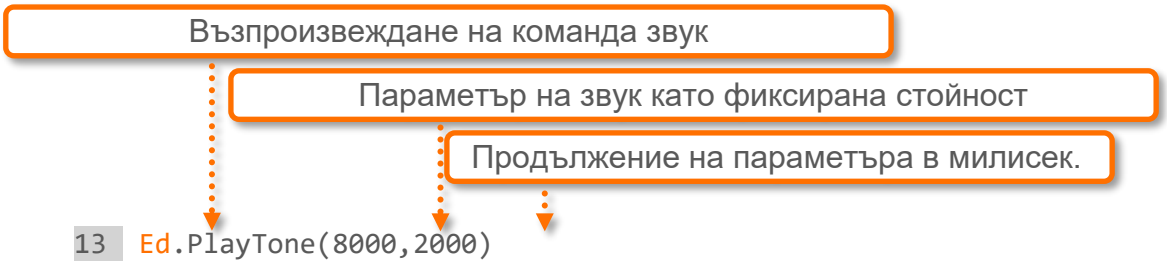
Name_____

$$\frac{8000000}{1000} = 8000$$

Ваш ред е:

Задача 1: Възпроизвеждане на персонализиран звук.

Напишете следната програма:



Приложете програмата, за да чуете как звучи.

Задача 2: Издаване на аларма

С тази програма, ние искаме Едисон да издава звуци с нарастващ период.

За да създадем аларма в програмата, трябва да използваме цикъл „for“, променливи и функцията range(). Също така трябва да използвате цикъл „while“.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 for i in range(33):
14     Ed.PlayTone(100+(i*100), 1000)
15     while Ed.ReadMusicEnd()==Ed.MUSIC_NOT_FINISHED:
16         pass
17
```

Напишете следната програма:

Приложете програмата, за да чуете как звучи.

1. Какво чухте от робота? Защо се случва това?

_____

_____

_____

Name_____

Важно умение в програмирането е да бъдете способни да проследявате какво се случва в програмата. Програмистите изпълняват проследяване на кода като метод за ръчно симулиране на изпълнението на техния код, за да проверят дали работи правилно преди да го компилират.
Това включва проследяването на програмата ред по ред, записвайки важни стойности. Често се прави за помагането при намирането на грешки или „бъгове“ в кода, също така е полезно, за да разберете какво се случва в програмата.
Опитайте се да проследите какво се случва в програмата и отговорете на следните въпроси.

2. Попълнете следната таблица, като изчислите периодичният параметър за всяка дадена стойност на „i“ в кода по-горе. Първата стойност е вече е попълнена.

Стойност на i	Периодичен параметър Първият входен параметър за PlayTone()
0	100
1	
2	

3. Каква е максималната стойност на i?

   _____

4. Каква е максималната стойност на периодичния параметър към функцията PlayTone()?

   _____

Опитайте и вие!

Приложението в акустичната технология се нарича акустично инженерство. Опитайте собственото си акустично инженерство. Експериментирайте чрез модифициране на параметрите на PlayTone(), за да накарате програмата да възпроизведе различни комбинации от звуци.

Name_____

Урок 5: Глава 5.3 – Издаване на мелодия

В този урок, трябва да напишете програма, която да накара Едисон да възпроизведе мелодия.

Това може да се случи чрез помощта на функцията **Ed.PlayTune()** и специален тип вход наречен „**string**“.

Използване на „string“ за произвеждане на мелодия

В Python, „string” е серия от символи в ред. Символ е всичко, което можете да напишете на клавиатурата като буква, число или специален знак от сорта на # или $. Например „Meet Edison“ е „string“ от 11 символа (10 букви и разстояние)

В приложението EdPy, трябва да използваме „string“ за възпроизвеждането на мелодия. Наричаме това „tune string“

Това са специални „стринг“ символи, които представят определени звуци. Те са съставени от ноти и продължителни входове, които са репрезентирани от символи.

Поредица от мелодии изглежда така: “ndndndnd…ndz”, където n е стринг символ от таблицата **Notes Table**, а d е продължителност на тона от таблицата **Duration Table**:

Нотна таблица

Символ на низ	Музикална нота
m	долно A
M	долно A #
n	долно B
c	C
C	C #
d	D
D	D #
e	E
f	F
F	F #
g	G
G	G #
a	A
A	A #
b	B
o	горно C
R	rest
z	Край на мелодия

Продължителност на нотата

Символ на низ	Времетраене
1	Цяла нота
2	1/2 нота
4	1/4 нота
8	1/8 нота
6	1/16 нота

Всички настройки на стринговете трябва да завършат с „Z“.

За да създадете стринг за мелодия, трябва да извикате функцията Ed.TuneString(), която има два входни параметъра. Размера на стринга (тоест

колко символа съдържа той), е първият параметър, а самият стринг, който искате да възпроизведете е втория параметър.
Можете да променяте скоростта, с която се изпълнява, като промените променливата **Ed.Tempo** в кода за настройка.

Ваш ред е:
Напишете следният код, за да възпроизведете мелодията „Mary Had a Little Lamb“:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 maryLamb = Ed.TuneString(53,"e4d4c4d4e4e4e2d4d4d2e4g4g4e4d4c4d4e4e4e4e4d4d4e4d4c1z")
13
14 Ed.PlayTune(maryLamb)
15 while Ed.ReadMusicEnd() == Ed.MUSIC_NOT_FINISHED:
16     pass
17
```

Тази мелодия е стринг в програмата:
“e4d4c4d4e4e4e2d4d4d2e4g4g4e4d4c4d4e4e4e4e4d4d4e4d4c1z”
Експериментирайте като променяте стойността на Ed.Tempo в кода на настройка.

1. Какви са различните стойности, който Ed.Tempo може да възприеме?

Бележка: Запомнете да използвате командата за автоматично довършване в EdPy. Опитайте да напишете „Ed.TEMPO“ и забележете всички възможни стойности за Ed.TEMPO, които автоматичното довършване допринася.

_____

_____

2. Кои стойности Ed.TEMPO ще възпроизведе най-бързо?

_____

3. Променете програмата за да възпроизведе само част от мелодията. Опишете промените, които трябваше да направите в програмата си, за да възпроизведете само част от мелодията.

_____

Урок 5: Глава 5.4 – Накарайте своят робот да танцува

В този урок, трябва да напишете програма, която да накара вашия робот да танцува.
В повечето добри танцови представяния, има стъпки които се повтарят.
Можете да накарате Едисон да повторя действия в танцова рутина използвайки цикъл „**for**“.
„**shimmy**“ е танцов ход, в която стоите неподвижно и мърдате раменете си напред и назад.
Погледнете програмата, която ще накара вашия робот да танцува в същия танцов ход.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 #Set up some variables
14 turnSpeed = Ed.SPEED_9
15 degreesToTurn = 20
16 numberOfTwists = 3
17
18 #Now shimmy!
19 Ed.Drive(Ed.SPIN_RIGHT,turnSpeed,degreesToTurn/2)
20 for i in range(numberOfTwists):
21    Ed.Drive(Ed.SPIN_LEFT,turnSpeed,degreesToTurn)
22    Ed.Drive(Ed.SPIN_RIGHT,turnSpeed,degreesToTurn)
23 Ed.Drive(Ed.SPIN_LEFT,turnSpeed,degreesToTurn/2)
24
```

Тази програма използва променливи, за да е по- лесно да се промени скоростта на завъртане, броят на обратите в танца и градусите на Едисон.
Погледнете ред 19 и 23. В тези редове извършваме математически изчисления в нашият код, за да накараме Едисон да се завърти наполовина на зададеният градус.

Name_____

Ваш ред е:
Напишете програма за танцуване.

Приложете програмата към вашия робот.

1. Колко пъти Едисон се обърна наляво?
   _____

2. Колко пъти се обърна надясно?
   _____

3. Първото обръщане надясно, е половината от разстоянието на всички обръщания в цикъла „**for**“, защото този ред има входен параметър „**degreesToTurn/2**“. Защо искаме този ред в програмата? Опитайте се да премахнете математическите естества **(/2)** и задействайте програмата отново. Какво забелязахте? (Бележка: вижте колко повече се движи наляво спрямо стартовата точка.)

_____
_____
_____
_____

Опитайте и вие!
Опитайте да промените променливите, за да промените начина по който Едисон танцува. Променете градусите на завоите, скоростта и броят на обръщанията.

Name_____

Урок 5: Глава 5.5 – Предизвикателство! Танцувайте с музика

Танците са по-забавни когато има музика! В този урок, ще напишите програма, съчетавайки танцови стъпки и мелодия.

Ваш ред е:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 #Set up my variables
14 turnSpeed = Ed.SPEED_9
15 degreesToTurn = 60
16 numberOfTwists = 3
17
18 #Now dance to the music!
19 Ed.Drive(Ed.SPIN_RIGHT, turnSpeed, degreesToTurn/2)
20 Ed.PlayTone(Ed.NOTE_C_7, Ed.NOTE_SIXTEENTH)
21 for i in range(numberOfTwists):
22     Ed.Drive(Ed.SPIN_LEFT, turnSpeed, degreesToTurn)
23     Ed.PlayTone(Ed.NOTE_A_7, Ed.NOTE_SIXTEENTH)
24     Ed.Drive(Ed.SPIN_RIGHT, turnSpeed, degreesToTurn)
25     Ed.PlayTone(Ed.NOTE_C_7, Ed.NOTE_SIXTEENTH)
26 Ed.Drive(Ed.SPIN_LEFT, turnSpeed, degreesToTurn/2)
27 Ed.PlayTone(Ed.NOTE_A_7, Ed.NOTE_SIXTEENTH)
28
```

Напишете следната програма, която съчетава танцова стъпка и мелодия:

Сега направете свой танц за Едисон, добавяйки мелодии или използвайте стринг. Можете ли да го синхронизирате, така че Едисон да танцува в такт с музиката?

1. Опишете движенията на робота. Има ли нещо в програмата ви, което харесахте? Ако да, опишете го.

_____

_____

2. Каква комбинация от тонове или ноти използвахте да се възпроизвежда в такт с вашият танц?

_____

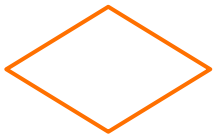
Name_____

Урок 6: Глава 6.1 – Задействайте светодиод чрез ръкопляскане

В този урок, ще трябва да напишете програма използвайки детектора за звук на Едисон, за да задействате светодиод в момент на пляскане.
Първо планирайте програмата си.

Блок-схеми в програмирането

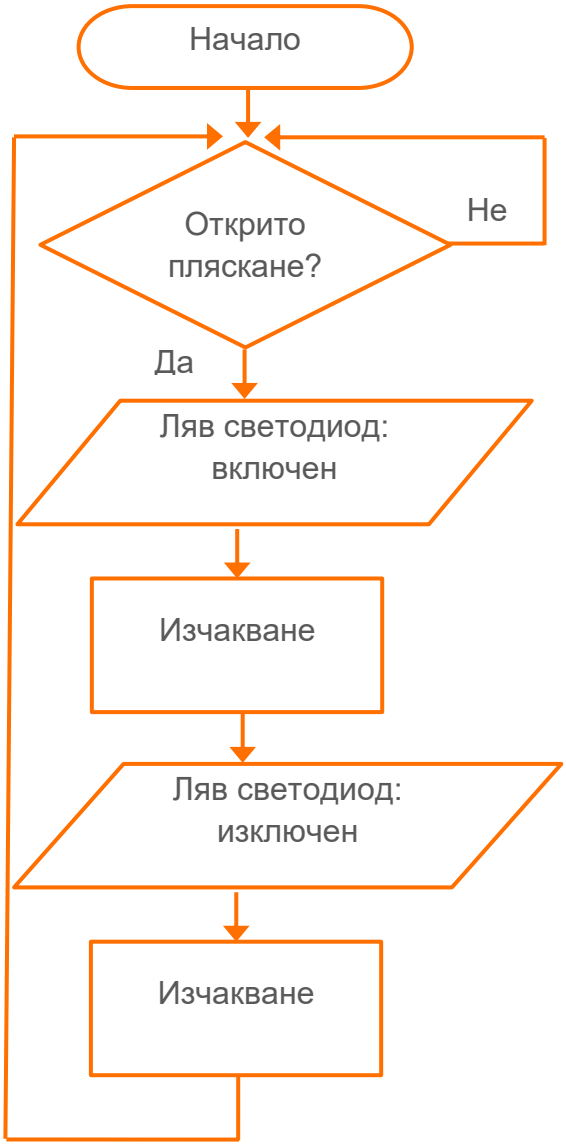
Професионалните програмисти обикновено планират своите програми преди да започнат да пишат своят код. Използването на блок-схемови диаграми е начин на програмистите да организират и планират своите програми.
Идеята на блок-схема е графично да се резюмира какво се случва в кода без подробности. Блок-схемите позволяват на програмиста да визуализира и съобщи как ще работи „потокът“ на програмата.
Блок- схема в програма се репрезентира чрез различни форми и стрелки. Всяка форма репрезентира различен елемент в програмата, а стрелките показват как елементите работят заедно.
Имат пет основни символа, които се използват в блок-схема:

Символ	Име	Функция
	Терминатор (начало/край)	Овалът представлява начална Или крайна точка
	Стрелка	Линия, която действа като свързващ елемент, показващ връзките между представителните форми
	вход/изход	Паралелограма представлява вход или изход
	Процес	Правоъгълникът представлява процес или действие
	Решение	Успоредникът репрезентира решение.

По-сложните блок-схеми също може да използват допълнителни форми с различни значения.
При правенето на блок-схема, за планирането на програма, думите често се добавят във формите или до стрелките. Тези дума са къси резюмета на процесът или решението.

Name_____

Да погледнем примерна блок-схема резюмираща програмата, която искаме да направим.
Имаме блок-схема за програма, която ще накара Едисон да изчака пляскане, след това да светне левият светодиод:



Тази програма ще използва звуковият детектор, за да реши дали има ръкопляскане или не. Резултата решава как ще продължи кода.
Когато погледнете блок-схемата, може да забележите, че няма край. Това е защото програмата използва цикъл “while”, за да я накара да продължи.

Name_____

Създаване на безкраен цикъл

Ако искате да напишете програма, която няма край се използва безкраен цикъл. В програмирането това се казва безкраен цикъл.
Може да използвате безкрайният цикъл да създадете планираната програма в блок-схемата.
Погледнете следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 while True:
13     Ed.ReadClapSensor()
14     while Ed.ReadClapSensor() != Ed.CLAP_DETECTED:
15         pass
16     Ed.LeftLed(Ed.ON)
17     Ed.TimeWait(50, Ed.TIME_MILLISECONDS)
18     Ed.LeftLed(Ed.OFF)
19     Ed.TimeWait(50, Ed.TIME_MILLISECONDS)
20
```

Тази програма е репрезентирана от нашата блок-схема и съдържа безкраен цикъл.
Погледнете ред 12, който се използва цикъл.
‘While’ циклите се нуждаят от условие. Цикъла ще повтори всеки код докато условието се установява като вярно.
Ако искаме „while“ цикълът да бъде безкраен, вместо да даваме условие, програмата трябва да даде оценка, която да заменим условието с вярно.
Вярно е винаги „вярно“. Като зададем условието на „Вярно“, сме закодирали състоянието на нашият цикъл за да бъде вярно.

Name_____

Ваш ред е:

Напишете кодът по-горе, за да програмирате левия светодиод на Едисон да свети при пляскане.

1. До какво разстояние роба прихваща ръкопляскането?
   _____

2. Каква е целта да има функцията TimeWait()? Какво би се случило ако я нямахте? Опитайте да стартирате програмата без нея.

_____
_____
_____

3. Погледнете програмата и блок-схемата и ги сравнете. Обяснете какво се случва в кодът когато крайния е резултат репрезентиран блок-схемата е „не“.

_____
_____
_____

Опитайте и вие!

Можете ли да промените програмата, така че и двата светодиода да светят когато има ръкопляскане?

Name_____

Урок 6: Глава 6.2 – Движение в отговор на ръкопляскане

В този урок, трябва да напишете програма, която да накара Едисон да се движи напред в отговор на ръкопляскане.

Ваш ред е:

Задача 1: Движение напред щом е доловено ръкопляскане

```
1
2  #------------Setup---------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ReadClapSensor()
13 while Ed.ReadClapSensor() == Ed.CLAP_NOT_DETECTED:
14     pass
15 Ed.Drive(Ed.FORWARD,Ed.SPEED_8,10)
16
```

Напишете следната програма:

1. Защо е необходимо да се извърши първоначално разчитане на сензора за ръкопляскане в ред 12? Какво върши? Бележка: Погледнете глава 2.5

_____

_____

_____

Задача 2 : Движение напред, след това назад щом е разчетено ръкопляскане.

Звуковите детектори на Едисон не разчитат само ръкопляскания. Сензорите могат да отговарят на големи шумове, затова може да почукате близо до говорителят на роботът, за да задействате звуковият сензор.

Задвижващите елементи на Едисон също издават звук, което също може да задейства звуковият сензор. За да се избегне това, трябва да промените програмата.

Ще добавите функцията TimeWait() с входен параметър от около 350 милисекунди, така ще се даде време на мотора да спре.

Също трябва да използвате ReadClapSensor() за да изчистите сензора.

Name_____

Напишете следната програма:

```
#-------------Setup----------------

import Ed

Ed.EdisonVersion = Ed.V2

Ed.DistanceUnits = Ed.CM
Ed.Tempo = Ed.TEMPO_MEDIUM

#--------Your code below-----------
Ed.Drive(Ed.FORWARD,Ed.SPEED_8,10)

Ed.TimeWait(350,Ed.TIME_MILLISECONDS)
Ed.ReadClapSensor()
while Ed.ReadClapSensor() == Ed.CLAP_NOT_DETECTED:
    pass
Ed.Drive(Ed.BACKWARD,Ed.SPEED_8,10)

```

Приложете програмата.

2. Обикновено, ще изработим блок-схема, за да планираме нашият код. За упражнение, направете блок-схема, която да съвпада с кода, който написахте. Може да използвате програма като Google Slides за изработването на блок-схемата.

Name_____

Урок 6: Глава 6.3 – Създайте собствена функция

В този урок, ще създадете собствена функция и ще я използвате да напишете програма за Едисон.

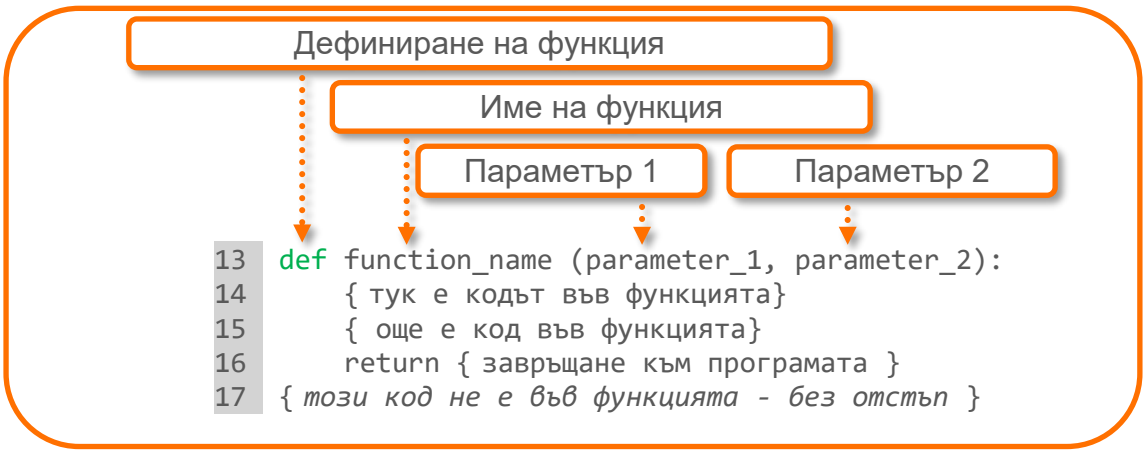
Какво по-точно са функциите?

След като програмите от известно време с Едисон и EdPy, сте използвали достатъчно различни функции от библиотеката на EdPy.
Както знаете, функцията е част от кода, който изпълнява определена роля или работа в програмирането, зависейки от какви входни параметри са използвани.
Може да не сте забелязали, че всички функции, които сте използвали досега всъщност са изпълнявали няколко редове от код, когато сте стартирали програмата.
Това е защото, функцията е блок от организиран код за повторно използване, който се използва за извършване на едно свързано действие.
Функциите са много полезни, защото те ни позволяват да програмираме по по-модулиран начин, използвайки един и същ блок от код в различни точки в програмата. За да накарате програмата да стартира всички тези редове от код, трябва да напишете следният ред: calling that function.

Гледайте това кратко видео, което обяснява как работят функциите:
https://youtu.be/8T5acEwfJbw

Създаване на собствени функции

До тук, всеки път щом сте използвали функция в EdPy сте извикали тази функция от библиотеката. Също така може да съставите собствени функции.
В Python, функциите изглеждат така:



Всичко това е част от функцията. Всичко което не е отстъпено не е част от функцията, а следващият ред в кода в програмата.
Важно е да се отбележи, че функциите, които създавате не са различни от тези използвани досега.
Когато извикате своята функция (със или без параметри), програмата преминава от повикването към кода във функцията (кода с отстъп). След това програмата изпълнява този код, преди да се върне към редът където сте направили извикването.
Ако зададете на функцията да върне стойност, когато програмата се върне в редът, в който сте направили извикването, извикването е заменено с върнатата стойност.
Това е важно, защото програмата винаги ще разрешава функции, след това математическите поръчки после изрази.

Организиране на вашият код

Конвенцията в EdPy е да напишете дефиницията на вашите функции в края на кода, след като вече сте използвали функцията си в програмата си.
Това е така, че вашият код е хубав и спретнат. Като организирате кода си по този начин, всичките ви функции, независимо дали използвате свои или извикващи функции от библиотека, могат да бъдат записани в основната част на вашата програма по визуално изчистен, организиран начин. Определенията на вашите функции могат да се намират извън това, по-долу в долната част на програмата.

Ваш ред е:

Задача 1: Упражнявайте се да дефинирате функция

Name_____

Вижте следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 directionToMove=Ed.SPIN_LEFT
13 speedToMoveAt=Ed.SPEED_8
14 distanceToMove=360
15
16 #call the function
17 moveOnClap(directionToMove, speedToMoveAt, distanceToMove)
18
19
20 #user defined functions
21 def moveOnClap(direction, speed, distance):
22     Ed.ReadClapSensor()
23     while Ed.ReadClapSensor()==Ed.CLAP_NOT_DETECTED:
24         pass
25     Ed.Drive(direction, speed, distance)
26
27
```

В ред 17 е извикването на функция. Редове 21 и 25 дефинират функцията. Помнете, че конвенцията е да дефинирате функцията в дъното на вашата програма, за да поддържате всичко чисто и организирано.
Напишете програмата и я приложете към Едисон. Тествайте, за да видите как работи.

1. Можем да напишем функция, която да накара Едисон да се движи в квадратна форма. Попълнете липсващите думи, за да допълните функцията.

```
def driveInaSquare():
  for i in range(___):
    Ed.Drive(_____,_____,_____)
    Ed.Drive(_____,_____,_____)
```

2. Напишете синтаксис за кода за извикване на функция. С други думи, какво ще напишете, за да извикате тази функция?

_____

Напишете програма, която ще накара Едисон да се движи няколко квадрата. Ще трябва да дефинирате driveInaSquare' функцията, и програмата ви ще

трябва да извика тази функция повече от веднъж. Приложете програмата за да видите как работи.

3. Програмата покри ли вашите очаквания? Ако не, опишете какво очаквахте и какво всъщност програмата извърши. Имахте ли проблеми при изработката на програмата?

_____
_____
_____
_____
_____

Задача 2: Определете своя функция
Напишете своя функция, която ще накара Едисон да върши нещо при пляскане. Може да го накарате да танцува, да свети светодиод, да се върти в кръг или каквото пожелаете!

Стъпка 1: Дизайн

Първо, напишете блок-схема, която описва вашата програма. Помнете да използвате правилните форми във вашата блок-схема, за да опишете началото на програмата, нейните процеси, решения и нейното протичане.
Формите които ще ви трябват зависят от вашата блок-схема. При всеки случай ще ви трябват овалът за стартова фигура, правоъгълникът за процесите и успоредникът за решенията.

Стъпка 2: Код

Преведете вашата идея в код в EdPy. Използвайте вашата блок-схема за ръководство и кодирайте вашите функции, за да включите всяка стъпка, която сте изложили във вашата блок-схема.

Стъпка 3: Тест

Напишете тестова програма, която включва вашата функция и допълнителен код, за да „упражнявате“ вашата функция. (Изпълнението на функция означава да го извикате в кода си.) Вижте дали вашата функция работи както сте планирали и очаквате да работи. Ако не, прегледайте отново вашата блок-схема и код, като коригирате при необходимост. Експериментирайте, за да видите как работи.

4. Опишете какво направи функцията ви.

_____
_____
_____
_____

Name_____

_____
_____

5. Опишете какви проблеми имахте.

_____
_____
_____
_____
_____

Name_____

Урок 7: Глава 7.1 – Калибриране откриването на препятствия

Можете да регулирате чуствителността на системата за откриване на препятствия на Едисон. По този начин Едисон ще открива препятствия по-напред. Намалявайки чуствителността ще има обратен ефект, и Едисон ще открива препятсвия само в близост. Използвайте това, за да промените чуствителността на системата на Едисон.

Стъпка 1: Разчитане на баркод

1. Поставете Едисон от дясната страна с лице към баркода.
2. Натиснете бутона за записване (кръглият) три пъти.
3. Едисон ще се придвижи напред и ще сканира баркода.

Баркод- Калибриране на откриването на препятствия

Стъпка 2: Задаване на максимална чувствителност

След сканирането на бардкода, поставете Едисон на маса и премахнете всички препятсвия пред него. След това натиснете стартиращият (триъгълният) бутон. Сега Едисон е в режим на калибриране.

Отляво чувствителността се калибрира първо.

1. Няколко пъти натиснете стартиращият бутон, което увеличава чуствителността, докато червеният светодио отляво пресветва.
2. Няколко пъти натиснете бутонът за записване, което намалява чувствителността, докато светодиода не спре да свети.
3. Натиснете стопиращият бутон, за да преминете към дясната страна.
4. Няколко пъти натиснете стартиращият бутон, докато дестният червен светодиод пресветва. Сега натиснете няколко пъти записващият бутон, докато светодиодът не спре да пресветва.
5. Натиснете бутонът за спиране, за да завършите калибрирането.

Бележка: Персонализирана чувствителност

Възможно е да зададете дистанцията от която ще бъдат открити препятсвия. За да направите това, сканирайте баркода за калибриране на препятсвия, поставете предмет пред Едисон на разстояние каквото пожелаете и натиснете старт бутона след което, повторете стъпките от 1 до 5.

Name_____

Урок 7: Глава 7.2 – Откриване на препятствие чрез инфраред

В този урок, ще научите повече за инфрачервената светлина и как Едисон може да я използва, за да открие препятствия.

Какво е инфрачервена светлина?

Има много видове светлина, някой от които са видими за човешкото око, а някой не са. Инфрачервената светлина не е видима.

> ***Знаехте ли?*** Въпреки, че хората не могат да я видят, тя е вид светлина. Следователно ще работи в тъмна обстановка. Затова когато включите телевизора чрез дистанционното устройство, то работи дори когато няма светлина в стаята.

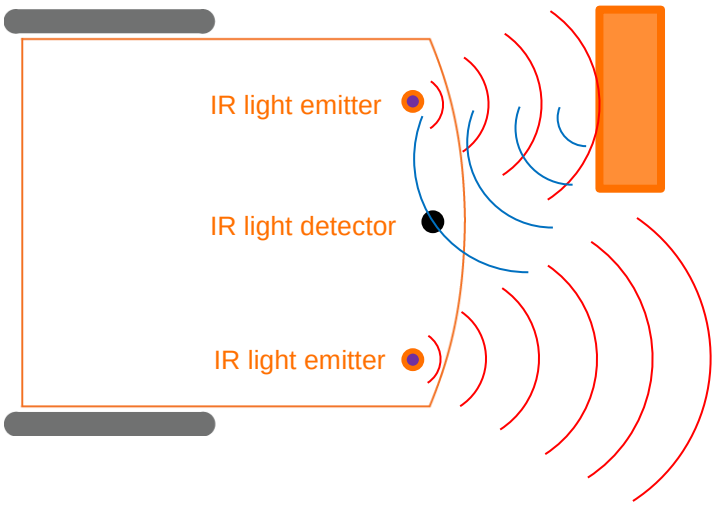
Едисон и инфрачервена светлина.

Роботът е оборудван с инфрачервена система, която дава на роботът вид зрение. Тази система позволява на Едисон да открива препятствия около него.

Инфрачервената система на Едисон е изработена от два светодиодни излъчватели отпред. Единият е отляво другият отдясно. Едисон също има инфрачервен сензор отпред точно по средата.

За да може Едисон да открива препятствия, инфрачервената светлина се излъчва от дясно и ляво на светодиодите. Ако инфрачервената светлина засече предмет, например стена, се връща сигнал към Едисон, след което сензора на робота открива отразената светлина.

Погледнете илюстрацията:



В тази илюстрация, има препятствие от лявата страна на Едисон, но не от дясната. Затова само лявата страна е отразена.

От полученият сигнал Едисон решава, че има препятствие отляво, но не отдясно.

Урок 7: Глава 7.3 – Откриване на препятствие и спиране

В този урок, ще трябва да напишете програма, която да накара вашият робот да се движи, докато не срещне препятствие и не спре.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.ObstacleDetectionBeam(Ed.ON)
14
15 Ed.Drive(Ed.FORWARD,Ed.SPEED_5,Ed.DISTANCE_UNLIMITED)
16
17 while Ed.ReadObstacleDetection() != Ed.OBSTACLE_AHEAD:
18     pass
19 Ed.Drive(Ed.STOP,1,1)
20
```

Погледнете следната програма:

Тази програма казва на Едисон да се движи докато не попадне на препятствие. Има няколко важни неща, които трябва да забележите за тази програма.
Обърнете внимание на ред 13. В този ред Едисон включва сензорния лъч за откриване на препятствия. Когато искате можете да използвате сензорният лъч в EdPy, винаги трябва да включвате сензорът преди да използвате програмата.
Сега обърнете внимание на ред 15. Този ред задава скорост на 5 в програмата. Когато използвате откриване на препятствия, трябва да използвате по-ниска скорост, за да позволите на робота да открие препятствието преди да се сблъсне с него. Ако скоростта е твърде висока, той ще се сблъска с него преди да го открие.

Name_____

Ваш ред е:

Задача 1: Изберете вашите препятствия

Трябва да изберете подобаващи препятствия, които можете да използвате с Едисон. Ако предметът е твърде малък или не излъчва достатъчно инфрачервена светлина, Едисон не може да го открие. Изберете предмет, който е непрозрачен, но не твърде тъмен (т.е. не черен), и да бъде висок поне колкото Едисон.

За тази програма, стена в стая ще бъде добро препятствие.

Задача 2: Подгответе своят робот.

Когато искате да стартирате програмата използвайки засичането на препятствия, добре е да се уверите, че засичането е калибрирано на разстоянието, което желаете. Използвайте указаното в глава 7.1, за да го калибрирате. Може да се наложи да направите персонализирано калибриране на чувствителността, за да се уверите, че робота ще открие и спре пред вашето препятствие.

Задача 3: Напишете и стартирайте програмата

Напишете програмата използвайки приложението EdPy, и я приложете към Едисон.

Експериментирайте, използвайки различни предмети, за да видите какво Едисон ще засече и какво не. Също може да променяте чувствителността на Едисон на различни разстояния, за да видите какво ще се случи.

1. Изтрийте редът ‘Ed.ObstacleDetectionBeam(Ed.ON)’, от програмата. Опитайте да я приложете. Какво се случи? Защо се случва това?

_____

_____

_____

2. Помислете къде сте виждали този тип невидимо засичане. Дайте пример.

_____

_____

_____

3. Къде мислите, че може да се използва този тип технология от разпознаване/засичане? Напишете поне една идея на това как бихте използвали тази технология.

_____

Name_____

Урок 7: Глава 7.4 – Избягване на препятствия

В този урок, ще напишете програма, която да накара вашият Едисон да се движи докато не открие препятствие, да завие и да го избегне.

```
#-------------Setup----------------

import Ed

Ed.EdisonVersion = Ed.V2

Ed.DistanceUnits = Ed.CM
Ed.Tempo = Ed.TEMPO_MEDIUM

#--------Your code below-----------

Ed.ObstacleDetectionBeam(Ed.ON)

Ed.Drive(Ed.FORWARD,Ed.SPEED_5,Ed.DISTANCE_UNLIMITED)

while Ed.ReadObstacleDetection() != Ed.OBSTACLE_AHEAD:
    pass
Ed.Drive(Ed.SPIN_RIGHT,Ed.SPEED_5,180)
Ed.Drive(Ed.FORWARD,Ed.SPEED_5,10)
```

Погледнете следната програма:
Тази програма кара Едисон да се движи до засичането на предмет. След като го засече, програмата казва на робота да завие на 180° и да се движи напред 10 cm.

Ваш ред е:

Задача 1: Напишете и стартирайте програмата.
Напишете програмата използвайки EdPy и я приложете.
След като я приложете, погледнете я отново и помислете как може да я промените. Опитайте да промените кодът, така че робота да реагира при засичането на предмет. (Бележка: опитайте да използвате звук и светлина.)

1. Помислете как може да подобрите оригиналната програма. Какво може да промените, така че програмата да върши повече от това само да се обърне в друга посока след засичането на препятствие?

_____
_____
_____

Name_____

Задача 2: Синтактични и логични грешки.
Програмистите често правят синтактични грешки. Важно е да намирате вашите синтактични грешки, за да направите корекция на вашият код.
Също може да получите логическа грешка, докато програмирате. В кодирането, всяка грешка е или синтактична или логическа.
Когато има логическа грешка в програмата, кодът не се изразява по начинът по който програмистите очакват. В EdPy, ако имате логическа грешка в програмата си, тя обикновено ще се приложи към Едисон. Когато я стартирате няма да се изпълнява по начинът, който сте предвидили.
Логическата грешка може да бъде доста проста, като да сте използвали грешната функция или изключване на функция. Пример на логическа грешка е да напишете програма, която използва откриване на препятствие, но да не включите сензорният лъч.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.ObstacleDetectionBeam(Ed.ON)
14
15 Ed.Drive(Ed.FORWARD,Ed.SPEED_5, Ed.DISTANCE_UNLIMITED)
16
17 While Ed.ReadObstacleDetection() != Ed.OBSTACLE_NONE
18 pass
19 Ed.Drive(Ed.SPIN_RIGHT,Ed.SPEED_5,145)
20 Ed.Drive(Ed.FORWARD, Ed.SPEED_5,20)
21
```

Вижте следната програма:

Програмата е предназначена да накара Едисон да се движи докато не попадне на препятствие, след това да се завърти на 135° и да се отдръпне от предметът с 20 cm.
Има пет грешки в програмата. Можете ли да ги откриете и да определите дали са синтактични или логически? Попълнете таблицата на следващата страница.
Бележка: Можете да напишете тази програма в EdPy и да използвате опцията за проверка на кода, за да ви помогне да откриете синтактичните грешки.

Name_____

Грешка #	Ред #	Вид грешка	Описание на грешката
1			
2			
3			
4			
5			

Name_____

Урок 7: Глава 7.5 – Засичане на препятствие като събитие

В този урок, ще напишете програма, управлявана от събития, за да накарате Едисон непрекъснато да се движи напред, като избягва препятствия.

Програмиране, управлявано от събития

В програмирането, събитието е нещо, което се случва извън програмния код, който засяга начина на изпълнение на програмата. Дадено събитие може да е натиснат бутон или да се предаде информация от датчик/сензор.

Много програмни езици включително Python, позволяват на програмистите да създадат код, който може да отговори на някой събития. Този тип програмиране се нарича „програмирано събитие“.

Когато пишете програма, която е задвижвана от събитие, ще трябва да напишете код, който да обработва тези събития. Има два основни начина да направите това, като програмата чака цикъл или чрез прекъсвания.

Прекъсваме този урок, за да говорим за… прекъсвания

Както вече знаете, програмите обикновено се движат последователно през кода, ред по ред. Има начини да позволи на кода да се движи по различен начин, като например чрез използване на цикли.

Може да промените начина, по който работи програмата чрез „прекъсване“.

Прекъсването е част от кода, който поставя на пауза основната програма и се изпълнява сама. След като кодът за прекъсване приключи, програмата се връща на мястото където е спряло в основната програма.

Прекъсванията са винаги функции дефинирани някъде в програмата. Обикновено те са кратки раздели на код, предназначени да извършват специфични задачи, без да създават сериозни смущения в потока на основната програма.

Програмистите използват прекъсвания, защото това позволява на програмата да реагира на даден случай във всеки момент, докато програмата работи. С други думи, чрез прекъсванията, програмистът няма нужда да предсказва кога по време на работа на програмата, ще се появи случай.

Обработване на събития и прекъсвания

Когато използвате прекъсвания в програмирането, управлявано от събития, ще трябва да използвате „манипулатори на събития“.

Манипулатор на събитие е начин на обвързване за прекъсване на конкретно събитие.

За да използвате манипулатор на събитие, трябва първо да зададете или да регистрирате, манипулатора на събитие във вашият код. Щом е регистриран, той ще следи за неговото събитие. Когато се появи събитие, манипулаторът ще

задейства прекъсване, което ще извика и стартира функцията, след което ще се върне към основната програма.
Погледнете следната програма:

```
1
2  #------------Setup---------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ObstacleDetectionBeam(Ed.ON)
13 Ed.RegisterEventHandler(Ed.EVENT_OBSTACLE_AHEAD, "avoidObstacle")
14
15 while True:
16     Ed.Drive(Ed.FORWARD,Ed.SPEED_5,Ed.DISTANCE_UNLIMITED)
17
18 def avoidObstacle():
19     Ed.Drive(Ed.SPIN_RIGHT,Ed.SPEED_5,180)
20     Ed.ReadObstacleDetection()
21
```

В EdPy, използваме функцията ‘**Ed.RegisterEventHandler’**, за да регистрираме манипулатора на събитие.
Функцията ‘**Ed.RegisterEventHandler’** има два параметъра. Първият е, събитието, което ще се появи, а вторият е функцията, която ще бъде извикана щом се появи събитието.
Обърнете внимание на ред 13. Този ред регистрира манипулатора, така че когато се появи ‘**EVENT_OBSTACLE_AHEAD’**, функцията ‘**avoidObstacle**’, ще бъде извикана.

Ваш ред е:
Напишете програма в EdPy, и я приложете към Едисон. Стартирайте я и вижте как работи.

1. Опишете какво прави вашият робот.

_____
_____
_____
_____

Name_____

2. Кой от редовете ще бъде изпълнен в програмата, щом е засечено препятствие? Защо програмата изпълнява тези редове?

_____
_____
_____
_____

3. Защо в програмата е включена функцията **Ed.ReadObstacleDetection**? С други думи, какво прави ред 20? Бележка: Върнете се към глава 2.5, за съвет, или се опитайте да премахнете ред 20 и опитайте да стартирате програмата.

_____
_____
_____
_____

Опитайте и вие!

Какво друго може да направи Едисон, когато засече препятствие? Опитайте се да промените кода, така че Едисон да извърши друго действие щом засече препятствие. Експериментирайте с програмата, за да видите какво работи и какво не.

Name_____

Урок 7: Глава 7.6 – Засичане на препятствие вдясно и вляво

В този урок, вие ще трябва да напишете програма за Едисон, чрез която той ще засича препятствия вляво и вдясно от него. За изпълнението на това, ще използваме изрази „**if**“.

Изрази „if“

Важна част от кодирането, е правенето на избори. Най-често това се прави чрез използването на изрази „**if**“.

Израз „if“, пита дали условието е вярно или невярно. Ако е вярно, тогава програмата изпълнява блока от изрази след това. Ако е невярно, тогава програмата игнорира израза и продължава към следващият ред на кода извън изразът „**if**“.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ObstacleDetectionBeam(Ed.ON)
13
14 while True:
15     if Ed.ReadObstacleDetection() != Ed.OBSTACLE_NONE:
16         Ed.PlayBeep()
17         Ed.ReadObstacleDetection()
18
```

Вижте следната програма:

Тази програма използва израз „**if**“, за да даде способността на роботът да прави решения без човешка намеса. Когато роботът може да прави сам решения, се нарича автономен робот.

Ваш ред е:

Задача 1: Звуков сигнал, ако има препятствие

Напишете програмата по-горе в приложението EdPy, приложете я към роботът и стартирайте програмата. След това опитайте да помръднете препятствие, като например ръката си, извън и във обсега на засичане на Едисон, за да видите какво ще се случи.

Name_____

1. Едисон ще може да реагира различно на различни стимули, като взема решение за това какво да прави. Това означавали, че Едисон има интелект? Бележка: Може да потърсите информация относно изкуствен интелект, за да ви помогне да отговорите на въпроса.

_____
_____
_____
_____
_____
_____
_____
_____
_____

Задача 2: Звуков сигнал, ако има препятствие или включване на светодиод. Освен, че може да казвате на програмата какво да прави ако изразът е верен, може да казвате какво прави и ако не е.

Ако, друго

Използвайки израз „**if**“, с „**else**“ ни позволява да пишем програми, които да извършват по-сложни решения. **if/else** изрази са начин на вземане на решение между две неща.

Гледайте Бил Гейтс, основател на Microsoft, обяснява изявленията if / else и вземането на решения в програмирането: https://youtu.be/fVUL-vzrIcM

В Python синтаксиса за **if/else** е:

```
if expression:
   statement(s)
else:
   statement(s)
```

Програмата се движи последователно от горе надолу, започвайки с условието „**if**“. Ако изразът „ако“ е верен, програмата стартира отстъпен код за израза „**if**“, и прескача израза „друго“. Ако израза „**if**“ е неверен, програмата прескача този участък с отстъпен код и вместо това изпълнява кода **else**.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ObstacleDetectionBeam(Ed.ON)
13
14 while True:
15     if Ed.ReadObstacleDetection() != Ed.OBSTACLE_NONE:
16         Ed.LeftLed(Ed.OFF)
17         Ed.PlayBeep()
18         Ed.TimeWait(100, Ed.TIME_MILLISECONDS)
19         Ed.ReadObstacleDetection()
20     else:
21         Ed.LeftLed(Ed.ON)
22
```

Напишете следната програма:
Приложете и стартирайте програмата. Опитайте да местите препятствие във и извън обсега на засичане на Едисон, за да видите какво ще се случи.
Тази програма има две пътеки: едната е препятствието да бъде засечено, а другата е да не бъде.

if, “elif”, else

Можете да съставите програма, която взема решение, използвайки повече от две условия. За да стане това, използваме друг синтаксис в Python:

```
if expression:
   statement(s)
elif expression:
   statement(s)
else:
   statement(s)
```

‘Elif’ е начин на изказване на “освен ако” в Python. Може да използвате ‘Elif’, за да напишете програма с многократни условия „ако“.
Програма, която използва **if/elif/else**, се движи последователно отгоре надолу. След като програмата изпълни отстъпен код във всяка част от структурата „**ако**“, тя ще пропусне останалата част от структурата и ще премине към следващият ред на кода извън структурата.

Това означава, че ако изразът „**ако**“, най-отгоре е верен, прогамата стартира отстъпният код за изразите „**ако**“ и пропуска **elif** секциите както и **else** секциите ако има такива. Ако изразът „**if**“ е неверен, програмата пропуска тази част и преминава на секцията **elif** от кодът.
Ако условието **elif** е вярно, програмата стартира отстъпният код и пропуска всичко под него намиращ се структурата на израза „**if**“. Ако **elif** е невярно, програмата продължава към следващата част от структурата на израза „**if**“ и т.н.
Задача 3: Засичане на препятствие вляво и вдясно

Вижте следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.ObstacleDetectionBeam(Ed.ON) #turn on obstacle detection
13
14 while True:
15     Ed.Drive(Ed.FORWARD, Ed.SPEED_1, Ed.DISTANCE_UNLIMITED)
16
17     obstacle=Ed.ReadObstacleDetection()
18
19     if obstacle>Ed.OBSTACLE_NONE: #there is an obstacle
20         Ed.Drive(Ed.BACKWARD, Ed.SPEED_5, 7)
21         if obstacle==Ed.OBSTACLE_LEFT:
22             Ed.Drive(Ed.SPIN_RIGHT, Ed.SPEED_5, 90)
23         elif obstacle==Ed.OBSTACLE_RIGHT:
24             Ed.Drive(Ed.SPIN_LEFT, Ed.SPEED_5, 90)
25         elif obstacle==Ed.OBSTACLE_AHEAD:
26             Ed.Drive(Ed.SPIN_RIGHT, Ed.SPEED_5, 180)
27         Ed.ReadObstacleDetection() #clear any unwanted detections
28
```

Тази програма има три различни начина по които може да се изпълни, ако е засечено препятствие базирано на това къде е засечено препятствието спрямо Едисон.
Напишете програмата в EdPy и я приложете към роботът.

2. Когато стартирате програмата, обяснете със свой думи, какво прави роботът:

Засечено препятствие отпред:

_____

Name_____

Засечено препятствие вляво:

_____

Засечено препятствие вдясно:

_____

Name_____

Урок 8: Глава 8.1 – Сензор за следене на контур

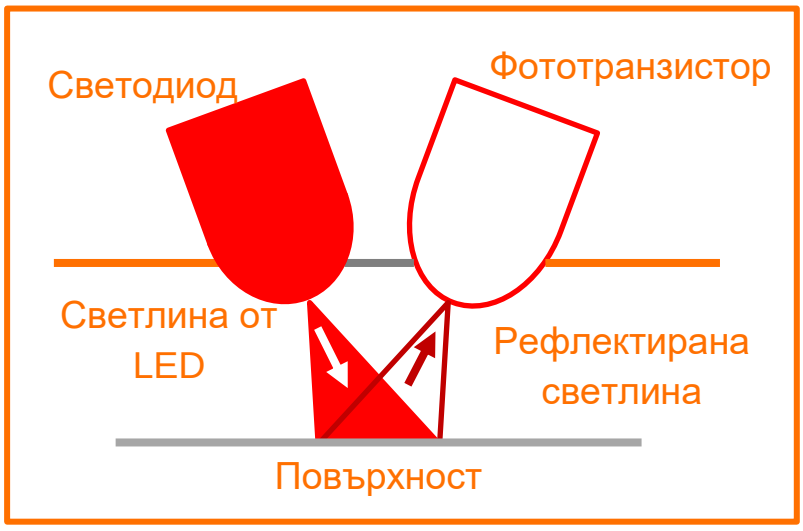
В този урок ще научите за сензорите на проследяване на контур на Едисон, и как той ги използва да стигне до заключение, дали се намира на отразяваща или не отразяваща повърхност.

Как работи сензорът за проследяване на контур на Едисон?

Вашият робот е оборудван със сензор на следен на контур, намира се до бутона за включване на робота, отзад на робота. Този сензор е направен от два основни електронни компонента:

1. Червена светлина излъчвана чрез диод (LED)
2. Фототранзистор (светлинен сензор).

Това изображение представлява напречен изглед на сензора за проследяване на Едисон:



Светодиодът на сензора за проследяване на контур излъчва светлина върху повърхността, по която се движи Едисон.
Фототранзисторът е сензор за светлина. Той измерва количеството светлина, рефлектирана обратно към повърхността под Едисон.

Ваш ред е:

Задача 1: Черно срещу бяло
Когато има повече рефлектирана светлина към фототранзистора на Едисон, му дава по-голямо разчитане на светлина
Експериментирайте, за да видите дали една бяла или черна повърхност ще отразява повече светлина.

Name_____

Използвайте указаното в 8.1 или част от бяла или черна хартия. Включете Едисон и натиснете кръглият бутон два пъти, за да се включи светиодиодът за проследяване. Повдигнете леко Едисон от хартията и погледнете отблизо кръглото петно на светлината, което светодиодът свети на повърхността. Сравнете колко ярка светлина се появява, когато се постави черна повърхност и след това бяла.

1. Кога светодиодът се отразява повече на черната или бялата повърхност?
   _____

Задача 2: Червено, зелено и синъо

Както видяхте в задача 1, има повече отразена светлина на бялата от колкото на черната повърхност. Това е защото светлината е по-ярка на бяла повърхност.

Когато фототранзисторът е на бяла повърхност, това дава по-голямо разчитане на светлина от това на черната повърхност. Черната повърхност се счита за не рефлектираща, а бялата се счита за рефлектираща.

Способността на фототранзисторът да определи дали Едисон се намира на рефлектираща или не рефлектираща повърхност, позволява на роботът да бъде програмиран да разпознава повърхността, по която се движи.

2. Помислете как проследяващият сензор ще отговори на следните повърхностни цветове. Дали ще ги определи всичките за рефлектиращи или не? Помнете, Сензора на Едисон излъчва червена светлина.

   Червена повърхност _____
   Зелена повърхност _____
   Синя повърхност _____

Урок 8: Глава 8.2 – Движение докато се стигне до черна повърхност

В този урок, вие ще напишете програма, чрез която Едисон да се движи напред на бяла рефлектираща повърхност, докато не премине на черна не рефлектираща.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 Ed.LineTrackerLed(Ed.ON)
14
15 Ed.Drive(Ed.FORWARD,Ed.SPEED_6,Ed.DISTANCE_UNLIMITED)
16
17 while True:
18     if Ed.ReadLineState() == Ed.LINE_ON_BLACK:
19         Ed.PlayBeep()
20         Ed.Drive(Ed.STOP,Ed.SPEED_6,0)
```

Вижте следната програма:

Обърнете внимание на ред 13. Този ред извиква функцията **Ed.LineTrackerLed(),** което включва сензорът за проследяване.
Както със сензорът за засичане на обекти, за да го използвате първо трябва да го включите. Включването му също ще включи неговите светодиоди.
Сега обърнете внимание на ред 19. В този ред извикваме фунцкията **Ed.PlayBeep().** Този ред не се отразява на работата на сензорът за проследяване. Същността на този ред е, че се използва дебъгване.

Дебъгване

Дебъгването е процес, който намира бъгове или грешки във вашата програма. Често програмистите ще добавят редове като ред 19 в програмата, към техният код, за да следи течението на програмата.
Да кажем, че статирате програмата, но рботът не спира на черната повърхност. Има две възможни причини: (1) роботът може да не разпознава черната повърхност или (2) може да има грешка в последната команда **Ed.Drive().**

Ако чуем звученето ще знаем, че черната повърхност е засечена. Следователно знаем, че грешката е в следващата команда. Този допълнителен код за дебъгване ни помага да открием грешката по-лесно.
Други функции може да се използват за дебъгване също, като командата **Ed.LeftLed().** Например, може да използвате тази команда, за да включите левият светодиод за показ, че в програмата е достигнала до определена точка.

Ваш ред е:
Напишете програма чрез EdPy, и я приложете към вашият робот. Използвайте черна повърхност за тестване на програмата. Може да нарисувате черна линия на бяла хартия или използвайте черен изолирбанд на бяла повърхност.

> **Бележка:** Когато използвате сензорът за проследяване в програма, винаги стартирайте роботът на бяла (рефлектираща) повърхност – нилкога на не рефлектираща.

Поставете Едисон на бяла повърхност и придвижете напред до черна.
След което, опитайте се да стартирате програмата отново, използвайки всяка от цветни повърхности описани в 8.1 една след друга. Придвижвайте Едисон към всяка повърхност, за да откриете дали роботът ще ги засече и ще спре.

1. Има ли цветове, които Едисон не може да засече? Ако да, кои са?

_____

2. Според вас защо получихте такъв отговор на първият въпрос? Защо Едисон не може да засече цветовете?

_____

3. Предствете си, че програмирате роботът да се движи по слалом трасе, с три слалом флага. Опишете как ще използвате функциите **Ed.PlayBeep(),Ed.LeftLed() и Ed.RightLed()** във вашата програмата за дебъгване и какво ще изпълнят те.

_____

Name_____

Урок 8: Глава 8.3 – Движение в граници

В този урок, трябва да напишете програма, която задържа Едисон в границите на черна повърхност използвайки сензорът за проследяване.
Първо трябва да планираме програмата си чрез псевдокод.

Псевдокод

Планирането на програмата си преди да започнете да я пишете е важна и полезна способност.
Един начин за това е чрез блок-схема, за да определите потока на програмата, което вече научихте в урок 6. Псевдокод е друг начин да репрезентирате програмата си преди да започнете нейното писане.
Псевдокодът е начин да напишете програмат си по по-лесен начин за разчитане. Прилича на опростен език за програмиране, но не е базиран на някой спецефичен език така че, е на свободен синтаксис. Вместо това псевдокодът използва Английски език, за да опише какво правят програмите. Затова псевдокодът е още наричан „структориран Английски“.
Когато планирате програмата си чрез псевдокод, би следвало да бъде по същата последователност по която бихте написали кодът за програмиране, за да може да бъде лесно за разбиране.
Ето пример за псевдокод описвайки програма, която кара Едисон да се придържа към границите на не рефлектираща повърхност, използвайки сензорът си за проследяване:

```
Включете проследяване на линия
Цикъл завинаги
        Задвижване напред
        Ако роботът открие черна линия тогава
СТОП
        Възпроизвеждане на звуков сигнал
        Завъртане надясно 135°
```

Ето съответния код в EdPy:

Name_____

Сравнете псевдокода с програмата. Виждате ли как псевдокода е преведен в програмата?

```
11  #--------Your code below-----------
12
13  Ed.LineTrackerLed(Ed.ON)
14
15  while True:
16      Ed.Drive(Ed.FORWARD, Ed.SPEED_3, Ed.DISTANCE_UNLIMITED)
17      if Ed.ReadLineState() == Ed.LINE_ON_BLACK:
18          Ed.Drive(Ed.STOP, Ed.SPEED_3, Ed.DISTANCE_UNLIMITED)
19          Ed.PlayBeep()
20          Ed.Drive(Ed.SPIN_RIGHT, Ed.SPEED_5, 135)
21
```

Ваш ред е:

```
1
2   #-------------Setup----------------
3
4   import Ed
5
6   Ed.EdisonVersion = Ed.V2
7
8   Ed.DistanceUnits = Ed.CM
9   Ed.Tempo = Ed.TEMPO_MEDIUM
10
11  #--------Your code below-----------
12
13  Ed.LineTrackerLed(Ed.ON)
14
15  while True:
16      Ed.Drive(Ed.FORWARD, Ed.SPEED_3, Ed.DISTANCE_UNLIMITED)
17      if Ed.ReadLineState() == Ed.LINE_ON_BLACK:
18          Ed.Drive(Ed.STOP, Ed.SPEED_3, Ed.DISTANCE_UNLIMITED)
19          Ed.PlayBeep()
20          Ed.Drive(Ed.SPIN_RIGHT, Ed.SPEED_5, 135)
21
```

Напишете следната програма:

Приложете програмата и я стартирайте, за да видите как работи.
Използвайте указаното в 8.2 като граница за тестване на програмата. Може да съставите ваша граница, като използвате голям лист хартия и черен маркер или изолирбанд на бяла повърхност, за да създадете голяма граница за роботът.
Модифицирайте програмата:
Опитайте се да премахнете стоп командата като изтриете ред 18. Експериментирайте с програмата като използвате различни скорости. Тествайте модифицираната си програма, за да видите резултата.

Name_____

1. Колко бързо може да се движи роботът, преди да настъпят проблеми?
   _____

2. Какво се случва, когато роботът се движи твърде бързо?

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Name_____

Урок 8: Глава 8.4 – Следване на линия

В тази част, трябва да използвайте сензорът за проследяване, за да напишете програма, която да накара Едисон да следва дадена черна линия.
За изпълнението на това, трябва да съставим алгоритъм за следване на черна линия.

Какво е алгоритъм?

Един алгоритъм представлява широк набор от инструкции за решаване на набор от проблеми. Един алгоритъм излага процес или набор от правила, които трябва да се следват, за да се реши всеки проблем в набора.

Алгоритми в програмирането

В програмирането често имаме набор от проблеми, които искаме да разрешим. Например в този урок, искаме да накараме Едисон да следва дадена черна линия. Следователно нашият набор от проблеми е „следва дадена черна линия“. Всяка специфична линия за следване е нов проблем в нашият набор.
Да кажем, че нарисувате черна линия, която Едисон да следва. За да накарате Едисон да следва вашата линия, вие трябва да напишете програма, която да каже на Едисон да следва точно тази линия. Ако нарисувате нова линия обаче, ще трябва да напишете изцяло нова програма за нея.
Вместо това, може да съставите алгоритъм.
Алгоритъмът произвежда програма, която ще работи за всички проблеми в наборът. По този начин не е нужно писането на нова програма за всеки проблем.
За разрешаването на нашият набор от проблеми, алгоритъмът трябва да издаде инструкции, които ще работят за всяка черна линия.
Може да планирате алгоритъм чрез псевдокод или блок-схема, точно както когато планирате програма.
Ето алгоритъм написан в псевдокод, който ще позволи на Едисон да следва всяка черна линия:

```
Включете проследяване на линията
Цикъл завинаги
        Ако роботът открие черна линия, карайте напред и
надясно
        иначе
                карайте напред наляво
```

Този алгоритъм казва, че ако сензорът за проследяване е на не рефлектираща повърхност, тогава Едисон ще се движи напред и вдясно, придвижвайки сензорът към бяла повърхност. Ако сензорът е на рефлектираща повърхност,

тогава роботът трябва да се движи напред и вляво, придвижвайки го към черна повърхност. По този начин роботът постоянно се движи напред, следейки границите на всяка линия.
Забележете, че няма споменати скорости или разстояния в псевдокодът. Този тип детайли обикновено са оставени за етапът на кодиране.

Ваш ред:
Преведете псевдокодът на алгоритъма в програма в Python, за да накарате роботът да следва всяка черна линия. Експериментирайте с различни скорости и разстояния.

1. Каква беше най-добрата комбинация от скорост и разстояние, за да постигнете гладко проследяване на линиите?

_____

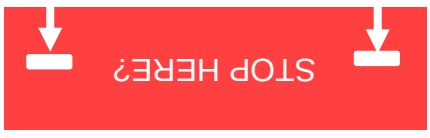
2. Как изглеждаше програмата ви в Python? Напишете вашият код.

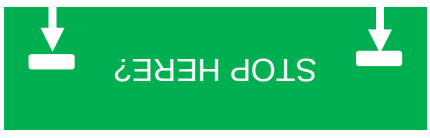
_____
_____
_____
_____
_____
_____
_____
_____

Опитайте и вие!
Направете ваша линия с черен маркер на бял лист или черно тиксо. Може ли Едисон да следва вашата линия?

Name_____

Урок 8: Упражнение 8.1

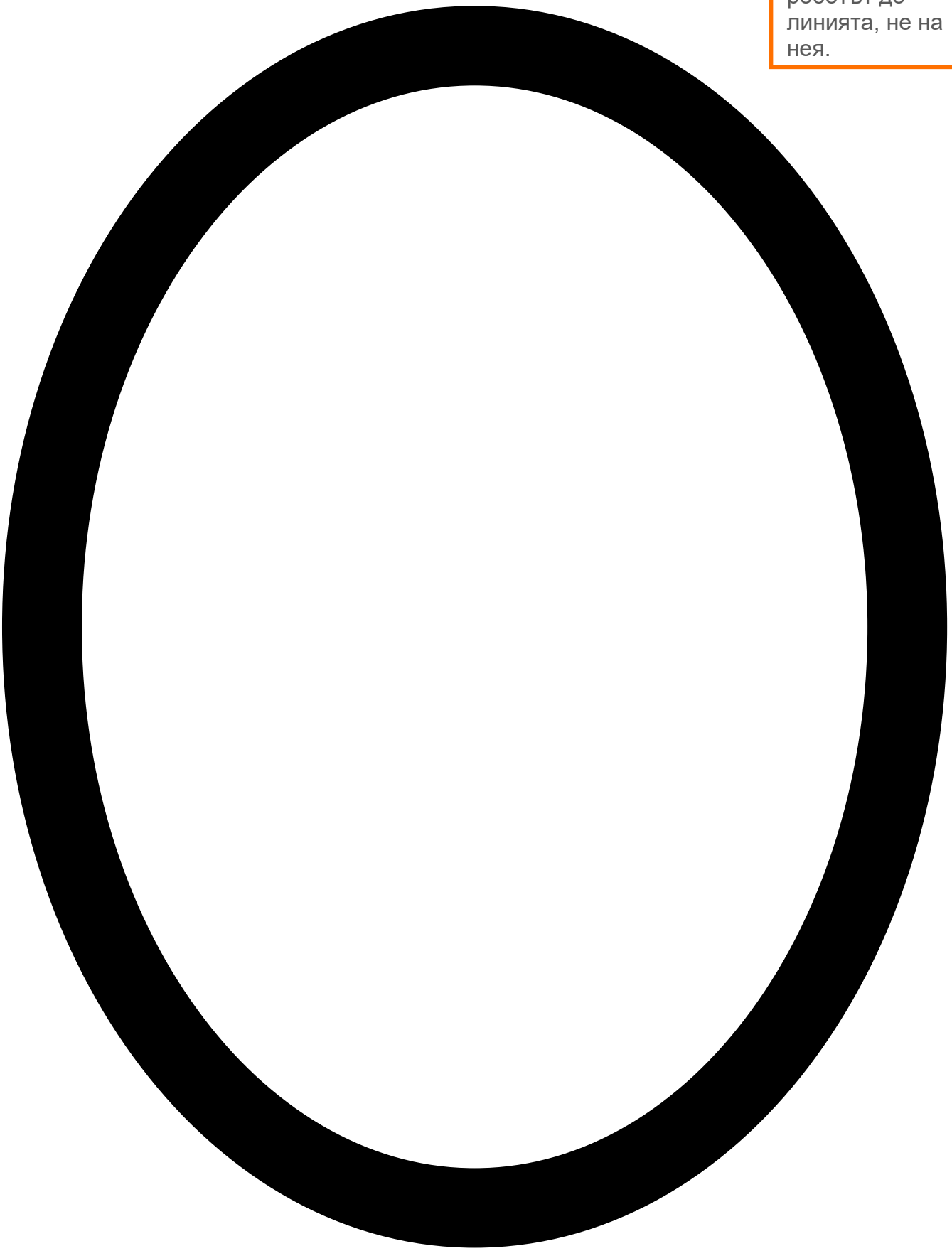




Name_____

Урок 8: Упражнение 8.2

Поменете!
Винаги поставяйте роботът до линията, не на нея.



Name_____

Урок 9: Глава 9.1 – Светлинна аларма

В този урок, вие ще трябва да напишете програма, която да включва алармата на Едисон, когато светлината в стаята се включат.

Използване на светлинните сензори на Едисон в програма

Вашият робот има два светлинни сензора, един отляво и един отдясно, които могат да долавят видима светлина. Може да използваме тези сензори, за да програмираме Едисон да реагира на светлина.

В една програма можем да направим тези сензори да прочетат количеството светлина, което е открито, и да го върнат като цифрова стойност.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 while Ed.ReadLeftLightLevel() < 100:
14     pass
15 while True:
16     Ed.PlayBeep()
17
```

Вижте следната програма:

Тази програма използва функцията **Ed.ReadLeftLightLevel()** в ред 13. Тази функция казва на програмата да разчете нивото на светлина от левият сензор на Едисон и да върне стойност между 0 и 1023.

Понеже фунцкията **Ed.ReadLeftLightLevel()** връща стойност, можем да приложим математически изчисления към стойността. Първият цикъл в програмата използва математика, за да определи какво да прави. Първият цикъл казва да премине (с други думи да не прави нищо), докато върнатата стойност от функцията **Ed.ReadLeftLightLevel()** е по-малка **(<)** от 100.

Когато върнатата стойност е по-голяма **(>)** от 100, програмата напуска първият цикъл и отива на следващият, което задейства аларма продължително.

Name_____

Ваш ред е:

Напишете програмата и я приложете към Едисон. Поставете роботът в тъмна обстановка преди да натиснете старт бутона.

След като сте изключили осветлението или сте блокирали левият сензор на Едисон, натиснете старт бутона. Когато светлините се включат, или каквото блокира сензора бъде премахнато, роботът ще задейства алармата.

1. Помислете за реална ситуация, в която този тип аларма би била полезна. Опишете ситуацията и как алармата може да се използва.

_____
_____
_____
_____

2. Какви промени трябва да бъдат извършени в програмата, за да се направи тъмна аларма?

_____
_____
_____
_____

Name_____

Урок 9: Глава 9.2 – Автоматични светлини

В тази част от урокът, трябва да напишете програма, която да задейства и двата светодиода на Едисон когато е тъмно.

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 Ed.Drive(Ed.FORWARD, Ed.SPEED_5, Ed.DISTANCE_UNLIMITED)
13
14 while True:
15     if Ed.ReadLeftLightLevel() < 100:
16         activateBothLights(Ed.ON)
17     else:
18         activateBothLights(Ed.OFF)
19
20
21 def activateBothLights(stateOfLed):
22     Ed.LeftLed(stateOfLed)
23     Ed.RightLed(stateOfLed)
24
```

Вижте следната програма:
В тази програма, ние използваме символът по-малко **(<),** за да определим пътят по който програмата ще продължи.
Ако върнатата стойност от функцията **Ed.ReadLeftLightLevel(),** е по-малка от 100, то тогава функцията activateBothLights() ще бъде извикана заедно с входния параметър на **Ed.ON**. В противен случай програмата продължава към “else” частта от оператора **„if**“, който също извиква функцията **activateBothLights(),** но с входния параметър на **Ed.OFF**.

Ваш ред е:

Напишете програмата и я приложете към роботът. Ще трябва да съставите няколко „тунела“ през които Едисон да се движи, което ще блокира светлината, за да се видят предните светлини на Едисон при включване.
Стартирайте програмата, като Едисон се движи, излизайки и влизайки в тунели, за да видите как работи.

Name_____

След което, експериментирайте със стойността (100) в оператора „**if**“, в ред 15, за да видите какво ще се случи, когато стартирате програмата с по-голям или по-малък номер.

1. Какво се случи, когато увеличихте стойността на оператора “**if**”?

_____

2. Какво се случи, когато намалихте стойността на оператора „**if**“?

_____

Опитайте и вие!

Ще има ли промяна ако използваме функцията **ReadRightLightLevel(),** вместо **ReadLeftLightLevel()?** Модифицирайте програмата си с тази промяна и я изпробвайте.

Name_____

Урок 9: Глава 9.3 – Следване на светлина

В тази част от урока, трябва да напишете програма, така че Едисон да следва светлина от фенер.
Вижте следната програма:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12
13 while True:
14     if Ed.ReadRightLightLevel() - Ed.ReadLeftLightLevel() < 0:
15         Ed.Drive(Ed.FORWARD_LEFT, Ed.SPEED_5, Ed.DISTANCE_UNLIMITED)
16     else:
17         Ed.Drive(Ed.FORWARD_RIGHT, Ed.SPEED_5, Ed.DISTANCE_UNLIMITED)
18
```

Тази програма сравнява нивата на светлина между десният и левият светлинен сензор, за да определи потока на програмата.
Присъствието на фенер на една от двете страни на робтът, ще го предизвика да разчита по - гоелми нива на светлина на едната страна. Логиката на програмата казва, че когато се извади стойността на десният източник на светлина от левият и резултатът е по-малък от нула, робтът се движи наляво към по-високият източник на светлина „**else**“ (или), се движи надясно.

Ваш ред е:

Задача 1: Проследете програмата.
В 5.2, вие научихте за проследяване на програмата. Когато има много пресмятания и много различни възможни стойности в програма, може да бъде полезно да я проследите. Това ви позволява да разберете различните стойности, който се появяват и да предвидите асоацираното поведение.

1. Попълнете следните проследявания за тази програма. Очакваното поведние би трябвало да бъде или ‘drive forward right’ или ‘drive forward left’.

Name_____

	Right_Light	Left_Light	Очаквано поведние
Фенер от дясно	200	100	
Фенер от ляво	100	200	
Без фенер	100	100	

Задача 2: Напишете и стартирайте програмата
Напишете програмата и я приложете към роботът. След като натиснете старт бутонът, насочете фенерът към Едисон. Роботът ще се движи към светлината. Използвайте фенерът, за да контролирате движението на Едисон.

2. Какво ще се случи, ако промените символът за по-малко (**<**) със симовлът за по-голямо (**>**)?

_____
_____

Какво мислите?
Програмата за проследяване по-горе, демонстрира поведението на роботът, че е доста подобно на това на молец привлечен към улична светлина в топла нощ. Интелигентен ли е молецът, който е привлечен от улична светлина? Ами робот със същото поведение?
Защо насекомо, което е привлечено от светлината е живо, а роботът не?

Name_____

Урок 10: Глава 10.1 – Робот вампир

В този урок, ще приложите всичко, което сте научили досега, за да създадете програма, която има много интересни поведения за изпълнение на вашият робот докато е трансформиран в робот вампир.
За да превърнем Едисон в робот вампир, ще използваме обектно-ориентиран код.

Обекти и класове в Python

Python е обектно-ориентиран програмен език. Обектно-ориентиран код е силен начин за намаляване на сложността в програмата. Затова много програмисти използват обектно-ориентиран код в големи, сложни програми.
В обектно-ориентиран програмен език, вместо да използваме самостоятелни променливи и функции, програмите също могат да използват обекти.
Обектът е колекция от функции и променливи, които са свързани помежду си. Обектите се дефинират от класове.
Един клас (също наричан класова дефиниция) е като план, който описва как да направите обект.
В една програма, можете да съставите много различни обекти от един клас, след което да манипулирате тези обекти индивидуално във вашата програма за различни цели. Всички съставени обекти от класът, ще имат един и същ план когато са създадени, но можете да правите различни неща на различни обекти в програмата си.

Пример: клас форма

Най-честият пример за клас е форма.
Всички форми като квадратна, триъгълна и шестоъгълна, имат общи атрибути. Например, всички форми имат някакъв брой страни и някакъв брой върхове. Има също общи неща, които правите с формите, като намиране на площта или обиколка на формата.
Може да използваме тези общи елементи да създадем клас за форми в Python. Като създадем клас, ние правим план само с познатите общи елементи. След това ще можем да създадем много различни обекти от този клас, като квадрат или триъгълник.
Синтаксиса за дефиниране на клас в Python е „class“, след това името, което искате да дадете на този клас, последвано от две точки (:). Пример:

```
class Shape:
```

Всичко, което добавите като отстъпен код в редовете, следващи това, е в дефиницията на класа.

```
class Shape:
     def __init__(self,x,y):
         self.x = x
         self.y = y

     def area(self):
         return self.x * self.y

     def perimeter(self):
          return 2 * self.x + 2 * self.y
```

Вижте следният пример на клас за форма в Python:
Този клас казва, че планът за всеки обект е съставен от „class Shape“, което има два входни параметъра (x и y). Класът също дефинира три функции: ‘init’, ‘area’ и ‘perimeter’.

Функциите __init__ и ‘self’

Всеки клас винаги трябва да има функция __init__.
Функцията __init__ е старт на кода. Терминът “init” значи „initialisation“ (инициализиране)
Когато програмата се нуждае от това да създаде обект, тя отива в дефинирането на класът и стартира функцията __init__. Тази функция съдържа код, който задава обекта, което обикновено означава задаване на началните стойности на променливите на обекта.
Функциите в един клас винаги трябва да имат „self“ за техен първи параметър. Това казва на създаденият обект да използва своите функции и променливи.

Пример: обект правоъгълник

След като сме създали дефиниция на класа, можем да я използваме, за да създадем обекти в нашата програма.
Да кажем, че искаме да създадем обект правоъгълник в „class Shape“.
В Python, когато искаме да създадем обект от клас, трябва да използваме следният синтаксис object_name = class_name(parameters).

```
rectangle = Shape(100,45)
```

Вижте примера за създаване на обект правоъгълник:
В една програма, това вика функцията __init__, което стартира създаването на обекта правоъгълник използвайки планът от дефиницията на „Shape class“. Обектът правоъгълник и зададен да има 100 за входна стойност “x” и 45 за входна стойност „y“. (Няма нужда да добавяте „self“- това е само справка за програмата да знае къде да търси.)
Щом програмата създаде обектът, той може да осъществи достът до функциите на обекта, както е определен от класа.
За да извикате някой от тези функции в Python, използвайте следният синтаксис.
object_name.function_name(parameters).
Например:

```
rectangle.area()
```

В този пример, функцията rectangle.area(), ще върне стойност на входния параметър „x“, на правоъгълния обект, умножен по входния параметър „y“.
Един начин за използването на тази функция, е да съхраните тази стойност в променлива във вашата програма. Например:

```
AreaOfRectangle = rectangle.area()
```

Задаване на клас В EdPy

В EdPy, може да използваме клас, за да създадем различни обекти в програма за Едисон. Ето прост клас за роботът вампир с функция __init__, както и още две функции: една за поведение през деня за роботът, и една за поведение през нощта. Класът вампир също съдържа променлива за възрастта на робота:

```
class Vampire:

    def __init__(self,age):
        self.age = age

    def doDayTimeBehaviour(self):
        Ed.LeftLed(Ed.OFF)
        Ed.RightLed(Ed.OFF)
        Ed.PlayTone(Ed.NOTE_A_7, self.age)
        while Ed.ReadMusicEnd() == Ed.MUSIC_NOT_FINISHED:
            pass
        Ed.TimeWait(1, Ed.TIME_SECONDS) # for debugging purposes and as a placeholder

    def doNightTimeBehaviour(self):
        Ed.RightLed(Ed.ON)
        Ed.LeftLed(Ed.ON) # for debugging purposes and as a placeholder
```

Това определение на класа ви позволява да създадете различни предмети за вампири. Може да използвате тези предмети за вампири в програма за Едисон.

```
1
2   #-------------Setup----------------
3
4   import Ed
5
6   Ed.EdisonVersion = Ed.V2
7
8   Ed.DistanceUnits = Ed.CM
9   Ed.Tempo = Ed.TEMPO_MEDIUM
10
11  #--------Your code below-----------
12
13  myEdisonVampire = Vampire(21)
14
15  while True:
16      # if it is the daytime
17      if Ed.ReadLeftLightLevel() > 100:
18          myEdisonVampire.doDayTimeBehaviour()
19      #it is night time
20      else:
21          myEdisonVampire.doNightTimeBehaviour()
22
23  class Vampire:
24
25      def __init__(self,age):
26          self.age = age
27
28      def doDayTimeBehaviour(self):
29          Ed.LeftLed(Ed.OFF)
30          Ed.RightLed(Ed.OFF)
31          Ed.PlayTone(Ed.NOTE_A_7, self.age)
32          while Ed.ReadMusicEnd() == Ed.MUSIC_NOT_FINISHED:
33              pass
34          Ed.TimeWait(1, Ed.TIME_SECONDS) # for debugging purposes and as a placeholder
35
36      def doNightTimeBehaviour(self):
37          Ed.RightLed(Ed.ON)
38          Ed.LeftLed(Ed.ON) # for debugging purposes and as a placeholder
39
```

Напишете програмата и се уверете, че разбирате какво се случва в нея:

Name_____

Ваш ред е:

Задача 1: Определете вампирските поведения

Сега е време да определите някой интересни поведения за вашият робот вампир. Бъдете креативни! Може да накарате вашият робот да реагира на натискания на бутони, ръкопляскания и обекти. Обмислете използването на обработка на събития във вашата програма. Можете да направите няколко вампирски обекта на каквато възраст искате. Може също да модифицирате базата на програмата, така че вашите обекти имат няколко входни параметри. Помислете какво искате да добавите във вашата програма, след това създайте блок-схема и после псевдокод базиран на блок-схемите.

Стъпка 1: Блок-схеми

Начертайте блок-схеми за поведения през деня и през нощта, или на ръка или на програма като Google Slides или подобна програма. Добавете вашите блок-схеми тук. Използвайте допълнителна хартия, ако ви е необходима.

Стъпка 2: Псевдокод

Преведете вашите блок-схеми в псевдокод. Напишете вашият псевдокод тук. Използвайте допълнителна хартия, ако вие необходима.

Name_____

Задача 2: Напишете вашата програма.
Преведете вашият псевдокод в код в Python, използвайки приложението EdPy. Приложете и тествайте програмата на Едисон.

Задача 3: Решаване на проблеми
В програмирането е много често, нещата да не работят както сте планували от първият път. Не се тревожете! Помнете: Томас Едисон се е провалил 10,000 пъти преди успешно да изобрети крушката. Така че, не спирайте да опитвате!
Учим се от проблемите, с които се сблъскваме, така че е добър навик да си дадете време и да помислите над това какво се е объркало. Опишете проблемите, които имахте, какви идеи измислихте, за да ги решите, и какви решения намерихте в опитите си. Използвайте допълнителна хартия, ако ви е нужна.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Задача 4: Опишете някой от програмните структури във вашата програма.
Изберете три различни програмни структури, които използвахте във вашата програма. Опишете всяка една по-долу.

1. Име на програмната структура:
   _____

   Какво прави тя?

_____
_____
_____

Name_____

_____

2. Име на програмната структура:
   _____

   Какво прави тя?

_____
_____
_____
_____

3. Име на програмната структура:
   _____

   Какво прави тя?

_____
_____
_____
_____

Задача 5: Презентирайте програмата си.