UČNA PRIPRAVA / OSNOVNI PODATKI

Šola: NAPOJ2

Avtor: Lidija Babič

Letnik: 1.

Datum: september 2017

Predmet: INFORMATIKA

Učna tema: Računalniški način razmišljanja

Učna enota: Razvoj kode za meritev reakcijskega čas

Učne oblike: frontalna, delo v skupinah/parih

Učne metode: pogovor, razlaga, demonstracija, samostojno delo

Predznanje:

. programiranja:

pozna logične operatorje

pozna zanki *while* in *for*

pozna sezname

. kodiranja v Py:

ve za knjižnici *gpiozero, time*

ve za knjižnico za delo z naključnimi števili (*random*)

zna napisati program za utripanje sveteče diode

zna izpisovati

zna pripeti nov podatek v seznam

. vezave:

zna na vezavni ploščici povezati svetečo diodo, ustrezen upor in RPi

zna povezati RPi in stikalo (objekt *Button* iz knjižnice *gpiozero*)

zna povezati RPi in piskač (objekt *Buzzer* iz knjižnice *gpiozero*)

Operativni učni cilji:

. dijak sledi ideji razvoja programa za merjenje reakcijskega časa od osnovne zamisli do zapisa kode

. dijak zna spremeniti kodo, če namesto vidnega signala (sveteča dioda) uporabi zvočni signal (piskač)

Učna sredstva:

. računalnik, projektor

. RPi z osnovno opremo / 1x na skupino

. sveteča dioda, 330-ohmski upor, stikalo, piskač, 5 m-m veznih žic / 1x na skupino

. učbenik (S čim vse se ukvarja informatika): <https://lusy.fri.uni-lj.si/ucbenik/book/1102/index3.html>

. delovni listi, konceptne kartice

Viri:

. Computational Thinking (<https://en.wikipedia.org/wiki/Computational_thinking>)

Prostor: računalniška učilnica (ne nujno, a priporočljivo)

Didaktične etape učnega procesa:

. pripravljanje / uvajanje

. obravnava nove učne snovi / usvajanje

. delo po skupinah ali parih

. povzetek

Medpredmetne povezave: fizika, angleščina

Novi pojmi:

. random.uniform(2.0, 5.0)

. Button.wait\_for\_press()

. time.time()

. list.append(cas)

Priloga: delovni listi / konceptne kartice

Izvedba :

Varianta A:

. dijaki sledijo razlagi in demonstraciji

. samostojno delo po skupinah

Varianta B (bolje):

. dijaki sledijo razlagi

. kodo vnašajo sočasno z učiteljem .. sproti preverjajo izvajanje kode

. samostojno delo po skupinah

Opomba:

. koda je zapisana "na krajši način"

namesto .. import gpiozero

in kasneje .. ledica = gpiozero.LED(27)

je zapisano .. from gpiozero import LED

in kasneje .. ledica = LED(27)

POTEK DVOURNE UČNE URE

5 min

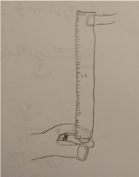
. motivacijski nagovor

A. KAJ JE REAKCIJSKI ČAS

Reakcijski čas je (povedano enostavno) čas od dražljaja do odziva. Dražljaj je lahko zvočni, vidni, ..

Pri pouku fizike ga največkrat določijo s pomočjo prostega pada ravnila. Ena oseba v nekem trenutku ravnilo spusti, druga pa ga poskuša uloviti.

Globje ko pade ravnilo, daljši je reakcijski čas osebe, ki je lovila ravnilo.



10 min

. preoblikovanje problema

. razdelitev večjega problema na manjše

B. NAMESTO RAVNILA ..

Radi bi izmerili ali določili reakcijski čas bolj “po računalniško”.

Potrebujemo

. vidni signal ..

namesto začetka padanja ravnila .. se bo prižgala sveteča dioda

. naključno izbran začetek dogodka ..

namesto, da ravnilo začne padati ob naključno izbranem trenutku

se bo sveteča dioda prižgala v naključno izbranem trenutku .. glej poglavje C

. konec dogodka ..

ravnilo smo ujeli .. svetečo diodo bomo ugasnili .. glej poglavje D

. meritev časa ali pa povezavo med časom in določeno drugo fizikalno količino

pri prostem padu ravnila poznamo zvezo med

časom padanja ravnila

in h = gt2/2

globino pada ravnila

torej iz globine pada ravnila (kar je enostavno izmeriti)

izračunamo čas pada ravnila

ta je kar enak reakcijskemu času

meritev časa s programom .. glej poglavje E

15 min

. reševanje prvega dela problema

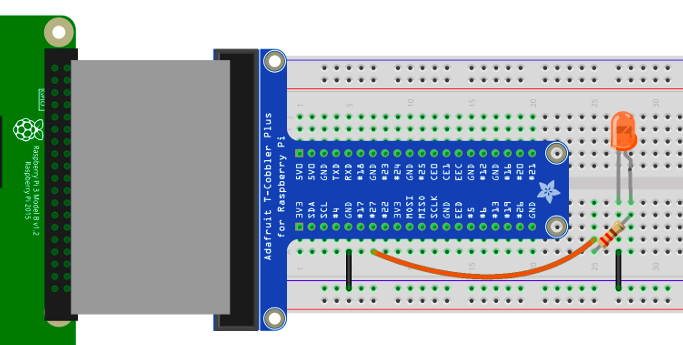
. dopolnjevanje/preoblikovanje kode (siva barva - staro, črna barva - novo)

C. KAKO PRIŽGATI SVETEČO DIODO OB NAKLJUČNEM TRENUTKU

Svetečo diodo znamo povezati na RPi z ustreznim uporom, znamo jo prižigati in ugašati.

Glej e-učbenik za ponovitev: <https://anzeljg.github.io/rpi/book/3201/index2.html>

Vezava sveteče diode na GPIO 27 s 330-ohmskim uporom za omejitev toka.



Koda (00 ledica.py):

from gpiozero import LED

from time import sleep

ledica = LED(27)

while True:

ledica.on()

sleep(1)

ledica.off()

sleep(1)

1. korak .. uvoz knjižnice, ki omogoča delo z naključnimi števili

Potrebujemo dodatno knjižnico random in iz te knjižnice fukcijo uniform.

Koda:

from gpiozero import LED

from time import sleep

from random import uniform

ledica = LED(27)

while True:

ledica.on()

sleep(1)

ledica.off()

sleep(1)

Funkcija uniform sprejme dva argumenta.

Primer:

>>> uniform (2.5, 10.0)

3.1800146073117523

Po izvedbi ukaza bomo dobili realno število (tip float) na intervalu od vključno 2,5 do nevključno 10,0 oziroma v matematičnem zapisu na intervalu [2.5, 10.0).

Vir: <https://docs.python.org/3/library/random.html#module-random> .. poglavje 9.6.7 z osnovnimi primeri

2. korak

Sveteča dioda bo najprej ugasnjena.

Koda:

from gpiozero import LED

from time import sleep

from random import uniform

ledica = LED(27)

ledica.off()

while True:

ledica.on()

sleep(1)

ledica.off()

sleep(1)

3. korak

Sveteča dioda se bo prižgala po naključno izbranem času. Vrednost naj bo na intervalu [2.0, 5.0).

Po eni sekundi bo ugasnila, potem pa se bo postopek ponovil.

Koda (01 rnd vklop.py):

from gpiozero import LED

from time import sleep

from random import uniform

ledica = LED(27)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

sleep(1)

ledica.off()

15 min

. reševanje drugega dela problema

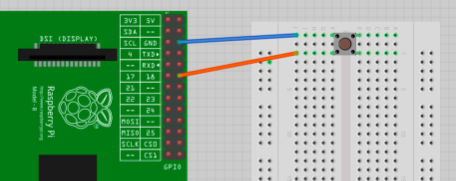
D. KAKO DRUGAČE UGASNITI SVETEČO DIODO

Ugasnili jo bomo s stikalom, ang. push-button.

Med svetečo diodo in stikalom ne bo “fizične” povezave. Elementa bosta povezana programsko (s kodo).

1. korak

Dopolnimo vezje, tako da eno vejo stikala vežemo na GND, drugo vejo pa na GPIO 21.



2. korak

Dopolnimo kodo .. iz knjižnice *gpiozer*o uvozimo objekt *Button*

Koda:

from gpiozero import LED, Button

from time import sleep

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

sleep(1)

ledica.off()

3. korak .. funkcija *wait\_for\_press()*

Ko se sveteča dioda prižge, naj program čaka na stisk gumba na stikalu.

Ko stisnemo gumb na stikalu, pa naj se sveteča dioda ugasne takoj, brez zamika ene sekunde.

Koda (02 stikalo.py):

from gpiozero import LED, Button

from time import sleep

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

stikalo.wait\_for\_press()

ledica.off()

15 min

. reševanje tretjega dela problema

E. MERJENJE ČASA

Uspelo nam je prižgati svetečo diodo ob poljubnem času in jo ugasniti ob stisku gumba na stikalu. Čaka nas še merjenje časa.

1. korak

Iz knjižnice time potrebujemo še funkcijo time().

Več tu: <https://docs.python.org/3/library/time.html>

Koda:

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

stikalo.wait\_for\_press()

ledica.off()

2. korak

Ko se prižge sveteča dioda, vklopimo še “štoparico” oziroma zabeležimo čas, ko se je to zgodilo. Zapišemo ga v spremenljivko start.

Ko pritisnemo gumb na stikalu, pa “štoparico” ustavimo oziroma ponovno zabeležimo čas, ko se je to zgodilo. Zapišemo ga v spremenljivko konec.

Koda:

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

3. korak

Razlika obeh časov je reakcijski čas. Zapišemo in zaokrožimo ga na dve decimalni mesti natančno.

Izpišemo ga na zaslon.

Koda (03 cas.py):

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

cas = round(konec - start, 2)

print(cas)

15 min

. reševanje zadnjega dela problema

. poskrbimo za več meritev in izračun povprečja

. ter lep izpis

F. POVPREČEN REAKCIJSKI ČAS

Ena sama meritev ni najboljši pokazatelj, vedno je bolje opraviti več meritev in izračunati povprečno vrednost.1. korak

Vsako od meritev bomo zapisali (ali pripeli) na seznam. Zato moramo najprej ustvariti prazen seznam. Poimenovali ga bomo meritve.

Koda:

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

meritve = []

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

cas = round(konec - start, 2)

meritve.append(cas)

print(cas)2. korak

Zdaj lahko naredimo “lepši” izpis meritev. Vpeljemo števec (spremenljivka stevec), ki ga povečujemo z vsako meritvijo.

Pri izpisu moramo paziti, da “seštevamo” enake tipe, zato str(stevec).

Koda:

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

stevec = 0

meritve = []

while True:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

cas = round(konec - start, 2)

meritve.append(cas)

stevec = stevec + 1

print(str(stevec) + “. meritev:”, cas, “s”)

3. korak

Uporabnika bomo vprašali, koliko meritev želi opraviti. Odgovor bo hranila spremenljivka koliko, za katero pa moramo poskrbeti, da je tipa int.

Potem je seveda smiselno, da izvajanje programa zaključimo, ko stevec doseže izbrano število meritev. Namesto neskončne zanke (while True), bomo zapisali končno zanko.

Koda:

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

stevec = 0

meritve = []

koliko = int(input (“Koliko meritev želiš narediti? ”))

while stevec != koliko:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

cas = round(konec - start, 2)

meritve.append(cas)

stevec = stevec + 1

print(str(stevec) + “. meritev:”, cas, “s”)

4. korak

Izračunajmo in izpišimo še povprečno vrednost reakcijskega časa, seveda primerno zaokroženega in zapisanega na dve decimalni mesti natančno.

Vpeljemo še spremenljivki vsota in povpCas.

Koda (04 reakCas.py):

from gpiozero import LED, Button

from time import sleep, time

from random import uniform

ledica = LED(27)

stikalo = Button(21)

ledica.off()

stevec = 0

meritve = []

vsota = 0

koliko = int(input (“Koliko meritev želiš narediti? ”))

while stevec != koliko:

sleep(uniform(2.0, 5.0))

ledica.on()

start = time()

stikalo.wait\_for\_press()

konec = time()

ledica.off()

cas = round(konec - start, 2)

meritve.append(cas)

stevec = stevec + 1

print(str(stevec) + “. meritev:”, cas, “s”)

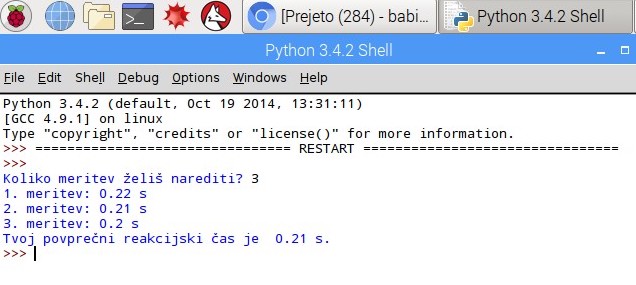
for i in range (len(meritve)):

vsota = vsota + meritve[i]

povpCas = round(vsota/len(meritve), 2)

print (“Tvoj povprečni reakcijski čas je”, povpCas, “s.”)

Primer izpisa:



10 min

. izziv .. namesto vidnega signala uporabi zvočni signal (spremeni vezje in kodo)

5 min

. pospravljanje