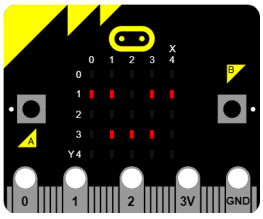
Δέκα απλά προγράμματα με το micro:bit

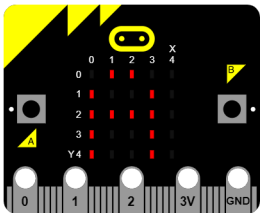
1 Buzzer

Σε ένα παιχνίδι με δύο παίκτες ο ένας παίκτης πατάει το κουμπί A και ο άλλος το κουμπί B, ποιό κουμπί πατήθηκε πρώτο;

Η λειτουργία του προγράμματος είναι να δείχνει την εικόνα:



Όταν κάποιος παίκτης πατήσει το δικό του κουμπί εμφανίζεται για 5 δευτερόλεπτα το A ή το B αντίστοιχα.



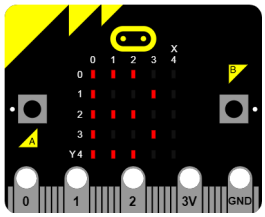


Figure 1: Image.ASLEEP

Το παρακάτω πρόγραμμα υλοποιεί αυτή τη λειτουργία:

```
#buzzer
#show the fastest of two players
from microbit import *

display.scroll('Buzzer', wait = True)
```

```
6
7  while True:
8     display.show(Image.ASLEEP)
9     if button_a.is_pressed():
10       display.show('A')
11       sleep(5000)
12    if button_b.is_pressed():
13       display.show('B')
14       sleep(5000)
```

Ο λογική του προγράμματος είναι να δείχνει την εικόνα `Image.ASLEEP`, να ελέγχει αν πατήθηκε το Α και στη συνέχεια να ελέγχει αν πατήθηκε το Β. Όταν δεν πατιέται κανένα κουμπί εμφανίζεται η εικόνα `Image.ASLEEP` με την εντολή:

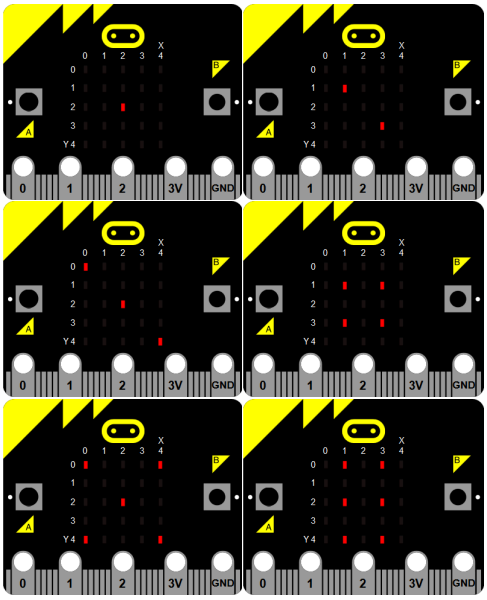
```
8  display.show(Image.ASLEEP)
```

για κάθε ένα από τα κουμπιά το πρόγραμμα ελέγχει αν πατήθηκε και αν πατήθηκε το εμφανίζει με την εντολή `display.show` στη συνέχεια το micro:bit δεν κάνει τίποτα για 5 δευτερόλεπτα που είναι 5000 χιλιοστά του δευτερολέπτου `sleep(5000)` ώστε να προλάβουμε να δούμε το Α ή το Β.

2 Ένα ζάρι

Πατώντας το κουμπί Α στην οθόνη του micro:bit εμφανίζεται το πάνω μέρος ενός ζαριού από το 1 έως το 6 όπως αυτά φαίνονται στις παρακάτω εικόνες.

```
1  #die
2  #throws a die
3  from microbit import *
4  import random
5
6  display.scroll('A Die', wait = True)
7
8  def getDiceImage(num):
9     num2images = {
10       1: '00000:00000:00900:00000:00000',
11       2: '00000:09000:00000:00090:00000',
12       3: '90000:00000:00900:00000:00009',
13       4: '00000:09090:00000:09090:00000',
14       5: '90009:00000:00900:00000:90009',
15       6: '09090:00000:09090:00000:09090'}
16    if num < 1 or num > 6:
17       display.show(Image.SAD)
18       return(None)
19    return(Image(num2images[num]))
20
21 while True:
22    if button_a.was_pressed():
```



```
23        display.clear()
24        sleep(500)
25        display.show(getDiceImage(random.randint(1,6)))
```

Η συνάρτηση `getDiceImage` παίρνει σαν όρισμα έναν αριθμό `num` και επιστρέφει την εικόνα του. Όταν πατιέται το κουμπί A τότε η οθόνη του micro:bit δεν δείχνει τίποτα για μισό δευτερόλεπτο και μετά δείχνει μια από τις έξι εικόνες με τυχαίο τρόπο.

3 Ζάρια

Το micro:bit ρίχνει δύο ζάρια και τα εμφανίζει το ένα μετά το άλλο.

```
1  #dice
2  #Throws two dice
3  from microbit import *
4  import random
5
6  display.scroll('Dice', wait = True)
7
8  def getDiceImage(num):
9     num2images = {
10       0: '00000:00000:00000:00000:00000',
11       1: '00000:00000:00900:00000:00000',
12       2: '00000:09000:00000:00090:00000',
13       3: '90000:00000:00900:00000:00009',
```

```
      4: '00000:09090:00000:09090:00000',
      5: '90009:00000:00900:00000:90009',
      6: '09090:00000:09090:00000:09090'}
   if num < 0 or num > 6:
      display.show(Image.SAD)
      return(None)
   return(Image(num2images[num]))

dice1 = 0
dice2 = 0
while True:
   if button_a.was_pressed():
      display.clear()
      sleep(500)
      dice1 = random.randrange(1,6)
      dice2 = random.randrange(1,6)
      ar = [getDiceImage(dice1),getDiceImage(dice2)]
      display.show(ar, wait = False, loop = True, clear = True, delay = 600)
```

Η εντολή `display.show` μπορεί να δείχνει και μια λίστα από εικόνες. Τη λίστα αυτή τη σχηματίζουμε με την εντολή `ar = ...` και στη συγκεκριμένη περίπτωση η λίστα περιέχει μόνο δύο εικόνες (τα δύο ζάρια). Στην εντολή `display.show` ορίζουμε να κάνει επανάληψη μεταξύ των δύο εικόνων `loop = True` και να δείχνει την κάθε μία για 0.6 δευτερόλεπτα `delay = 600`.

4 Αντανακλαστικά

Το πρόγραμμα δείχνει ποιο κουμπί πρέπει να πατήσει οπαίκτης και όταν αυτός το πατήσει εμφανίζει τον χρόνο που πέρασε από τη στιγμή που εμφανίστηκε στην οθόνη το A ή το B μέχρι να πατηθεί το κουμπί μετρώντας στην ουσία τα αντανακλαστικά.

```
#reflex
#Displays A or B and
#the time in which the player presses
#the corresponding time
from microbit import *
import random

display.scroll('Reflex', wait = True)

buttons = ['A','B']
while True:
   sleep(random.randint(2000,3000))
   button = random.choice(buttons)
   display.show(button)
   strt = running_time()
   if button == 'A':
      while not button_a.is_pressed():
```

```
        pass
    reflex = running_time() - strt
    display.scroll(str(reflex),wait = True)
 elif button == 'B':
    while not button_b.is_pressed():
        pass
    reflex = running_time() - strt
    display.scroll(str(reflex),wait = True)
```

Ο χρόνος εκκίνησης αποθηκεύεται με την εντολή `strt = running_time()` στην μεταβλητή `strt` μόλις ο χρήστης πατήσει το κουμπί το πρόγραμμα βγαίνει από την συνεχόμενη επανάληψη:

```
while not button_a.is_pressed():
   pass
```

και ο χρόνος που διανύθηκε υπολογίζεται με την εντολή `reflex=running_time()-strt`. Στη συνέχεια ο χρόνος αυτός εμφανίζεται με την εντολή `display.scroll(reflex)`.

5 Διάλεξε ένα γράμμα

Το πρόγραμμα διαλέγει ένα γράμμα από την αλφαβήτα αλλά το κάνει σαν να τη λέει από μέσα του με έναν ρυθμό που μπορούμε να τον προγραμματίσουμε. Η λογική προκύπτει από την αλφαβήτα που λέμε πριν παίξουμε *όνομα ζώο πράγμα φυτό*.

```
#pickaLetter
#button A starts the alphabet
#button B stops the alphabet and
#shows the chosen letter.
from microbit import *

display.scroll('Pick a letter')

letter = ['A','B','C','D','E','F','G','H','I','J',
         'K','L','M','N','O','P','Q','R','S','T',
         'U','V','W','X','Y','Z']
while True:
   if button_a.is_pressed():
      display.show('A')
      sleep(1000)
      strt = running_time()
      display.show(Image.ALL_CLOCKS, wait=False, loop = True)
   if button_b.is_pressed():
      display.show(letter[((running_time()-strt)//200)%26])
```

Το `running_time()-strt` της τελευταίας γραμμής είναι ο χρόνος που διανύθηκε. Αν κάνουμε το ακέραιο μέρος της διαίρεσής του με το 200 έχουμε το πλήθος των γραμμάτων που είπε το πρόγραμμα από μέσα του. Ωστόσο αν το πρόγραμμα ξαναρχίσει από την αρχή το πλήθος αυτό θα είναι μεγαλύτερο από 26 που είναι τα γράμματα της αγγλικής αλφαβήτου για αυτό

κάνουμε και το υπόλοιπο της διαίρεσης με το 26 και έτσι το γράμμα που αναζητούμε είναι στη θέση `((running_time()-strt)//200)%26` του πίνακα `letters`.