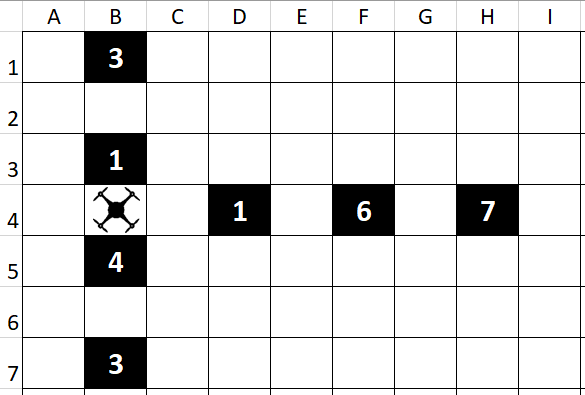
1. feladat: Drón reptetés (40 pont)

Van egy drónunk , amely épületek magasságát képes megállapítani egy lézeres érzékelővel. A drón érzékelője mindig azt az értéket adja vissza, amely a legmagasabb épület magassága az adott irányba nézve. **Azonban a mérőeszközének hatótávolsága 5 egység, utána már nem lát.** Az alábbi felülnézeti pályán a fekete négyzetekben lévő számok azt jelentik, hogy az ott található épület hány emelet magas.

A drón helyzetét úgy tudjuk megadni, hogy megadjuk melyik cellába kerüljön (pl. B4), és melyik irányba nézzen. A drón négy irányba nézhet, felfele (F), jobbra (J), lefele (L), és balra (B). A drón nem lehet olyan helyen, ahol van épület.

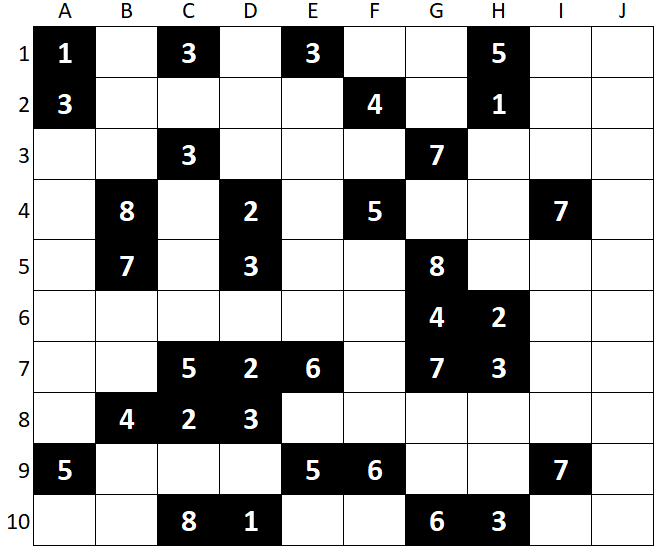
Lássunk egy példát. Ha a drón a B4J parancsot kapja, akkor a B4-es cellába repül és jobbra néz. Ekkor a legnagyobb ház, amire rálát a hatótávolságán belül, 6 emelet magas, így a 6-os értéket fogja visszaadni. (A 7 emeletes házat már nem érzékeli, mert az 6 egység távolságra van).

Ha a B4B parancsot kapja, akkor ugyanebben a cellában marad, de balra néz, ahol nem lát épületet, így a nulla értéket adja vissza. Az E4J,F2L parancsokra érvényes, hogy mindegyik 6-os értéket ad vissza. (De sok más parancsot ki lehetne adni, amelyre igaz, hogy 6-os értéket ad vissza…)

Most nézzük az alábbi pályát!

Adj meg olyan parancsokat, amelyek kiadásakor igazak lesznek az alábbi állítások.

Ha több olyan parancs is van, ami megfelelő lehet, add meg mindegyiket, vesszővel elválasztva. Ha esetleg nincs megfelelő parancs, írd azt, hogy Nincs.



|  |  |  |
| --- | --- | --- |
| **Kérdés** | **Állítás** | **Parancs(ok)** |
| A. | A drón jobbra néz, és 5-ös értéket ad vissza |  |
| B. | A visszaadott érték 1 |  |
| C. | A visszaadott érték 2 |  |
| D. | Az érték 7-es, de ha az érzékelő nagyobb távolságra mérne, akkor 8-as lenne |  |

2. feladat: Négyszögek (32 pont)

Egy négyszöghöz azonos méretű kisebb négyszögeket teszünk. Használunk egy eljárást:

def tégla(a,b):  
 for i in range(2):   
 turtle.forward(a)   
 turtle.left(120)   
 turtle.forward(b)   
 turtle.left(60)

Melyik programhoz melyik ábra tartozik?

A. def egy(a,b):  
 for i in range(2):   
 tégla(a,b)   
 turtle.forward(3\*a)   
 tégla(a,b)   
 turtle.forward(a)  
 turtle.right(60)   
 turtle.forward(3\*b)   
 turtle.right(120)

B. def kettő(a,b):  
 for i in range(2):   
 tégla(b,a)   
 turtle.forward(2\*a)   
 tégla(b,a)   
 turtle.forward(2\*a)  
 turtle.right(60)   
 turtle.forward(3\*b)   
 turtle.right(120)

C. def három(a,b):  
 for i in range(2):   
 tégla(b,a)   
 turtle.forward(4\*a)   
 turtle.right(60)  
 tégla(b,a)   
 turtle.forward(3\*b)   
 turtle.right(120)

D. def négy(a,b):  
 for i in range(2):   
 turtle.forward(a)   
 tégla(a,b)   
 turtle.forward(a)   
 tégla(a,b)   
 turtle.forward(2\*a)  
 turtle.right(60)   
 turtle.forward(3\*b)   
 turtle.right(120)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 1 | 2 | 3 | 4 |

Mit rajzol a négy eljárás a következő hívásokra, ha a tégla eljárásban a balra helyett jobbra fordulunk, a négy további eljárásban pedig felcseréljük a két fordulást (előbb a jobbra 120, utána a jobbra 60)?

E. egy(30,60)

F. kettő(30,60)

G. három(30,60)

H. négy(30,60)

3. feladat: Függvény (18 pont)

A valami függvény egy szót kap paraméternek.

def valami(x):  
 if len(x)<2:   
 return 0  
 else:  
 if x[0]==x[-1]:   
 return valami(x[1:len(x)-1])  
 else:  
 return 1+melyik(valami(x[1:]),valami(x[:len(x)-1]))

A melyik függvény két számot kap paraméternek.

def melyik(a,b):  
 if a<=b:  
 return a   
 else:  
 return b  
Mit adnak eredményül a következő függvényhívások?

A. valami(”anna”)

B. valami(”panna”)

C valami(”miriam”)

D. valami(”aranka”)

E. valami(”éva”)

F. Milyen paraméterre lesz 0 a függvény értéke?

G. Fogalmazd meg általánosan a függvény feladatát!

4. feladat: Rovásírás (20 pont)

A rovásírás egyes betűi egyenes vonalakból állnak. Az alábbi eljárásokból hiányzik egy-két utasítás. Ezek helyét egy üres dobozzal jelöltük. Add meg a hiányzó utasításokat úgy, hogy a mellékelt betűket rajzolja ki az eljárás.

 def mbetű(h):  
 turtle.forward(h)   
 for i in range(2):   
 x   
 turtle.forward(h/2)   
 turtle.right(120)  
 for i in range(2):   
 turtle.forward(h/2)   
 turtle.left(120)

 def pbetű(h):   
 x   
 turtle.left(120)   
 turtle.forward(2\*h/3)   
 turtle.backward(2\*h/3)   
 turtle.right(120)   
 turtle.backward(h/3)   
 turtle.left(120)   
 turtle.forward(2\*h/3)   
 turtle.backward(2\*h/3)   
 turtle.right(120)   
 turtle.backward(h/3)  
 0   
 turtle.forward(2\*h/3)   
 turtle.backward(2\*h/3)   
 turtle.right(120)   
 turtle.backward(h/3)

 def rbetű(h):   
 turtle.forward(h)   
 turtle.backward(3\*h/5)   
 turtle.right(66)   
 turtle.forward(h/2)   
 turtle.left(66)   
 turtle.forward(2\*h/5)   
 x   
 turtle.forward(3\*h/5)   
 turtle.right(66)   
 turtle.backward(h/2)  
 turtle.left(66)   
 turtle.backward(2\*h/5)