**Python Ágazati**

**Python érzékeny kis és nagy betűkre, tehát a print() működik de a Print() nem, úgyhogy valami beépített függvény nagy kezdőbetűvel van írva az valószínűleg csak word faszsága**

**Változó**: egy szó amihez hozzákötsz egy értéket

pl:

a = 5

4 alap fajtája van: int (egész szám), float (tört szám), str (szöveg) bool (True/False)

lehet a fajták között váltogatni az int(x) float(x) és str(x) függvényekkel

pl:

a = float("45") -> a = 45.0

Ha a változó fajtája nem egyezik pl ha szöveget akarsz összeadni számmal akkor kifagy a program és TypeErrort fogsz kapni, ilyenkor kell a szöveget intre váltani

**print()**

Kiírja az értéket amit beírsz neki a képernyőre

**input()**

Bekér egy szöveget

pl:

x = input(">")

ha ide beírod hogy 5 akkor visszakapod hogy "5", mint szöveg

x = int(input(">"))

ha ide beírod hogy 5 akkor visszakapod hogy 5, mint szám mert az int() automatikusan átváltja szövegből számba

**Modulo (maradék)**

z = x % y

Így a z értéke az a maradék lesz amit akkor kapnál ha elosztanád x-et y-vel, pl ha x=10 és y=3 akkor z =1

**Egész számos osztás:**

z = x // y

Így a z értéke az az x és az y hányadosa maradék nélkül pl ha x= 10 és y=3 akkor z=3

**Hatvány**

X\*\*Y

**Strip:**

Akkor kell ha egy txtből olvasol be szöveget és a végén ott marad a “\n”, ami a sortörést jelenti, a strip() az annyit csinál hogy bekér egy szöveget és visszaadja azt ‘\n’ nélkül

Pl

X = “asd\n”

X = strip(x)

Ilyenkor x=“asd”

**x.Replace(y, z)**

Az x-ben kicserél minden y-t z-re

Tehát

X = “palacsinta”

X = x.replace(“a”,”.”)

Ilyenkor x = “p.l.csint.”

**F string:**

Olyan szöveg amibe vannak beleszúrva értékek pl:

X = 5

Y = 2

Szoveg = f”az első szám {x}, a második {y}”

Ilyenkor a szoveg=”Az első szám 5, a második 2”

**If** (elágazás):

Megnézi hogy az érték vagy művelet amit kapott az igaz-e

műveletek: ==, >, >=, <, <=, != (nem egyenlő)

ilyenkor figyelni kell h 2 =t rakj mert ha 1et raksz akkor azt hiszi hogy változót akarsz definiálni és kifagy

A művelet után kell : és a következő sort beljebb kell kezdeni

Pl:

X = 5

If x == 5:

Print(“az x értéke 5”)

Ilyenkor kiprinteli, mert az x == 5 művelet igaz

**Else:**

Ha a fölötte lévő if érték nem igaz akkor ez fog helyette lefutni

X = 10

If x == 5:

Print(“az x értéke 5”)

Else:

Print(“Kys”)

Ilyenkor azt fogja kiprintelni hogy Kys

**Elif:**

Ha több if elágazást akarsz létrehozni és azt akarod hogy az else akkor fusson le ha mindegyik hamis akkor ezt kell használnod

Pl:

If x == 1:

Print(“a”)

Elif x == 2:

Print(“b”)

Elif x == 3:

Print(“c”)

Else:

Print(“d”)

Ez azért kell mert másképp python csak az utolsó ifhez kapcsolná hozzá az elset, tehát ha x= 1 akkor kiírná azt is hogy a meg azt is hogy d

**And/or:**

Akkor használjuk ha az if-be komplexebb műveletet írunk

Pl

If x == 1 and y == 2:

Print(“:)”)

Itt csak akkor fut le ha az x=1 és y=2

If x == 1 or y == 2:

Print(“:)”)

Itt csak akkor fut le ha az x=1 vagy y=2, tehát ha x=5 és y=2 akkor is lefut

**While** (while ciklus)

Addig fut amíg a művelet igaz

Pl:

szoveg = “”

While szoveg != “kilepes”:

Szoveg = input(“>”)

Ez most addig fog futni amíg be nem írod a > után hogy kilepes

Jegyzet:

Mode = “0”

While mode == “0”:

Input(“Válasszon egy módot: ”)

While mode == "1”:

Input(“Choose a mode: ”)

Ha az első inputhoz beírod hogy 1, akkor át fog váltani a másikra, mert mode=1,de ha a choose a modehoz beírod hogy 0, akkor nem fog visszamenni az elsőre mert python nem futtatja le visszamenőlegesen

Ha ezt helyesen szeretnéd megcsinálni akkor:

mode = "0"  
while mode == "0" or mode == "1":  
 if mode == "0":  
 mode = input("Válasszon egy módot: ")  
 if mode == "1":  
 mode = input("Choose a mode: ")

Ez mostmár működne, mert ha a choose a modenak beírod hogy 0, akkor újraindul a while loop és megint magyar lesz

(Ennyi valószínűleg elég lesz az 1. feladathoz)

**Lista**

Egy komplex változó típus ami értékek sorozatát tartalmazza

Pl:

X = 10

Lista = [1,”kys”,3.14,x]

**Lista függvények**

**Lista.Append(x)**

Hozzáad valamit a listához

Pl:

Lista = []

Lista.append(1)

Lista.append(4)

Lista.append(3)

Ilyenkor lista = [1,4,3]

**Lista.Remove(x)**

Kivesz valamit a listából

Pl:

lista = [1,4,3]

Lista.remove(3)

Ilyenkor list = [1, 4]

**Lista[i] (indexelés)**

Kiadja a listából az i. értéket, úgy hogy 0-tól kezdi a számolást

Pl:

Lista = [1,4,3]

X = lista[1]

Ilyenkor x= 4, mert a lista 0. értéke 1, a 1. 4, a 2. pedig 3

Lista = [1,4,3]

Lista[2] = 0

**Lista.index(x)**

Visszaadja x indexét a listában,

Lista = [1,4,3]

X = lista.index(4)

Ilyenkor x = 1

**Lista.Pop(i)**

Kivesz valahonnan valamit a listából

Pl:

Lista = [1,4,3]

Lista.pop(2)

Lista = [1,4]

**Lista.Clear()**

Kitöröl mindent a listából

**Lista = [\*szoveg]**

Itt a szoveg minden egyes karakterét beleteszi egy listába külön értékként

**Szoveg.Strip(x):**

A szoveget elválassza ott ahol x van és listát csinál belőle.

Az x itt alapjáraton space

Pl:

X = “K ys”

X = x.split()

Ilyenkor az x=['K’, ‘ys’]

X = “kayas”

X = x.split(‘a’)

Ilyenkor x=['k’,’y’,’s’]

**“”.join(lista)**

Ilyenkor a listából stringet csinál, úgy hogy azzal köti őket össze ami az idézőjelben van

Pl:

Lista = [1, 4, 3]

X = “&”.join(lista)

Ilyenkor x=”1&4&3”

**Len(lista/szoveg)**

Visszaadja a lista/szoveg értékét:

(a szöveg esetén a spaceket is beleszámolja)

Pl

Lista = [1,4,3]

Szoveg = “HEAVY METAL DARKLORD CRASHER”

X = len(lista)

Y = len(Szoveg)

Itt az x=3 és y=28

**For** x in lista (for ciklus):

Végigmegy a listán és lefuttatja annyiszor a kódot annyiszor amennyi érték van a listában, úgy hogy az x mindig a mostani érték lesz

Itt az x olyan mint egy változó tehát ha mondjuk numnak vagy a-nak nevezed el akkor ugyan úgy működik

Pl:

Lista = [1,4,3]

For x in lista:

Print(x)

Így először ki fogja írni hogy 1, aztán 4, aztán 3

**Continue**

Mindent amit ezalatt van azt átlép és átmegy a következő értékre

Pl:

Lista = [1,4,3]

For x in lista:

If x == 4:

continue

Print(x)

Itt most csak az 1-et és a 3-at fogja kiprintelni, mert ha az x=4 akkor egyből továbblép és nem éri el a printet

**Break**

Átlépi az egész for/while ciklus többi részét

Pl:

Lista = [1,4,3]

For x in lista:

If x == 4:

break

Print(x)

Ilyenkor csak az 1-et fogja kiprintelni mert a 4 már nem éri el a printet, a 3 pedig le sem fut

**Range(x,y)**

Ez csinál egy intervallumot x és y között, úgy hogy x beleszámít de y nem

Ha csak x-et adsz meg akkor 0 és x között fogja számítani

Pl:

For i in range(1,5):

Print(i)

Ez kiírja a számokat 1-4-ig

Ez csak fornál működik tehát ha beírod hogy

X = range(4)

Akkor x nem =[0,1,2,3] hanem range(4)

Ha listát akarsz belőle csinálni akkor

Lista = []

For i in range(4):

Lista.append(i)

Ilyenkor lista=[0,1,2,3]

**X (not) in lista/szoveg/range**

Megnézi hogy x benne e van listában/szövegben

Pl

X = int(input(“Adjon meg egy számot:”))

Lista = [1,4,3]

If x in lista:

Print(“x benne van a listában”)

Ilyenkor ha az x 1, 4, vagy 3 akkor printel

**For i,x in Enumerate(lista)**

Úgy megy végig egy listán hogy az i az x indexe

Pl:

Lista = [1, 4, 3]

For i,x in enumerate(lista):

Print(f”a lista {i}. Tagja {x}”)

Ez most ki fogja írni hogy:

A lista 0. tagja 1

A lista 1. tagja 4

A lista 2. tagja 3

**Függvények (Def):**

Olyan programsorok amiket többször is le akarsz futtatni

Pl

Def fuggveny():

Print(“Csináld ezt”)

Print(“Csináld azt”)

For i in range(10):

Fuggveny()

Ez most 10x kiírja hogy csináld ezt meg h csináld azt,

A függvénybe tudsz bekérni egy változót is

Def fuggveny(x):

Print(“a”\*x)

Kys(50)

Ilyenkor kiírja 50szer hogy a

**Return**

Ha vissza szeretnél kapni valami értéket a függvényből akkor a return kulcsszót kell használni

Pl:

Def negyzet(x):

Return x\*\*2

A = negyzet(5)

Ilyenkor a=25

+A return levágja a függvény alját, ugyan úgy mint a break

Note: A globális változók és a függvényben lévő változók külön vannak tehát

Pl

A = 10

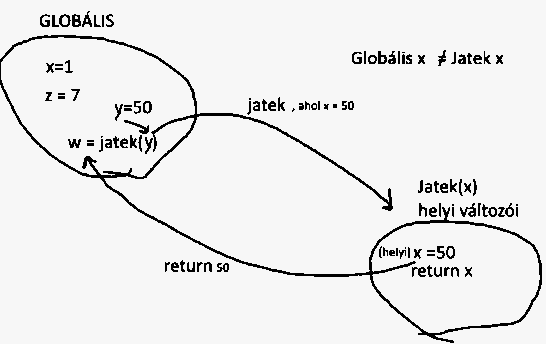
Func change\_a():

A = 5

Change\_a()

Ilyenkor az a értéke 10 marad, mert a change\_a()ban lévő a és a globális a két külön változó

Ábra:



**Import random**

Ha random számot akarsz generálni akkor a kód legtetején be kell hívnod a random modult ezzel a kóddal

**Random.randint(x,y)**

Generál egy random számot x és y között úgy hogy y már nincs benne

Pl:

X = random.randint(0,11)

Ilyenkor az x lehet akármi 0 és 10 között

(Ennyi valószínűleg elég lesz a 2 feladathoz)

**Szótár:**

Egy olyan lista amiben az indexek helyett kulcsok vannak

Pl:

Szamok = {

“x” : 5,

“y” : 10,

“z” : 15

}

Itt ha beírnád hogy szamok[“y”] akkor 10-et kapnál vissza

Itt append helyett [] van

Pl:

Szamok = {

“x” = 5,

“y” = 10,

“z” = 15

}

Szamok[“w”] = 20

Ilyenkor szamok = {

“x” = 5,

“y” = 10,

“z” = 15

“w” = 20

}

Ez azért más mintha azt írnád hogy

X = 5

Y = 10

Z = 15

Mert ha külön változókkal csinálod akkor jobban fut a program és átláthatóbb, de a szótárba tudsz létrehozni új kulcsokat miközben fut a program ez azért kell mert pl egy játék esetén nem kell külön beírogatni hogy enemyhp1 = 0, enemyhp2 = 0, hanem csak

Enemyhps = {}

For i in range(0,100):

Enemyhps[f“Enemy{i}”] = 100

Enemyhps[“Enemy51”] -= 70

Ilyenkor van egy szótárad amiben benne van Enemy0-Enemy99 hpja, és az enemy51-nek sebeztünk 70-et tehát csak 30 hpja maradt

**Szótár.erase(x)**

Ilyenkor kiveszi az x kulcsot a szótárból tehát

szamok = {

“x” = 5,

“y” = 10,

“z” = 15}

Szamok.erase(“z”)

Ilyenkor szamok = {

“x” = 5,

“y” = 10

}

**Szótár.keys() / Szótár.values()**

A szótár.keys() visszaadja a szótár kulcsait, a szótár.values(), pedig a szótár értékeit egy listában

szamok = {

“x” = 5,

“y” = 10,

“z” = 15

}

Kulcs = szamok.keys()

Ertek = szamok.values()

Ilyenkor a kulcs [“x”,”y”,"z”] a ertek meg [5,10,15]

**Txt beolvasás**

With open(“txtneve.txt”,encoding=”utf-8”) as f:

X = f.readlines()

Ez most beolvassa a txtneve.txt-t f-ként és lementi a sorait egy listába amiben minden sor egy str aminek a végére került egy ‘\n’ a sortörések miatt (strippel le tudod szedni).