VŠB TECHNICKÁ UNIVERZITA OSTRAVA | HORNICKO GEOLOGICKÁ FAKULTA | KATEDRA GEOINFORMATIKY

Programování v GIS 1
4 - Procvičování struktur řízení chodu programu

Michal Kačmařík

A924, tel.: 5512

e-mail: michal.kacmarik@vsb.cz

https://www.hgf.vsb.cz/548/cs/

Příklad 1

Na vstupu je zadáno číslo. Vytiskněte na standardní výstup informaci o tom, zdali je kladné, záporné, či rovno nule.

x = 0.01
Jak bude vypadat výsledek tištěný na standardní výstup:

cislo je kladne.

Tvorbu kódu je možné si vždy vyzkoušet na:
https://www.w3schools.com/python/trypython.asp?filename=demo_default

Příklad 1

```
x = 0.01

if x > 0:
    print("cislo je kladne")
elif x < 0:
    print("cislo je zaporne")
else:
    print("cislo je rovno nule")
```

Příklad 2

Na standardní výstup vytiskni nejvyšší ze tří zadaných čísel.

A = 10
B = 8
C = 14

Jak bude vypadat výsledek tištěný na standardní výstup:

14

Příklad 2

```
A = 10
B = 8
C = 14

if A > B:
    if A > C: #vnorena podminka
        print(A)
    else:
        print(C)
elif B > C: #pokracovani prvotni podminky, proto se odsazeni vraci zpet na zacatek radku
    print(B)
else:
    print(C)
```

Příklad 3

Vytvořte cyklus, který bude na standardní výstup tisknout celá čísla od 1 do 10, a ke každému číslu bude na stejný řádek zároveň tisknout informaci o tom, zdali číslo je sudé, či liché.

Jak bude vypadat výsledek tištěný na standardní výstup:

1 liche

2 sude

3 liche

4 sude

5 liche

…

Příklad 3 – cyklus while

```
i = 1 #zavadim ridici promennou i
while i <= 10:
    #prikaz % vraci zbytek celociselneho deleni, pokud je zbytek celociselneho deleni
    dvema roven nule, cislo musi byt sude
    if i%2 == 0:
        print(i, "sude") #kombinace cisla a textu v jednom prikaze print
    else: #pokud neni cislo sude, musi byt liche
        print(i, "liche")
    i+= 1 #zvysuji ridici promennou o 1 pro dalsi iteraci cyklu
```

Příklad 3 – cyklus for

```
for i in range(1, 11):
    if i%2 == 0:
        print(i, "sude")
    else:
        print(i, "liche")
```

Příklad 4

Tiskni na standardní výstup násobek dvou po sobě jdoucích čísel uložených v seznamu.

seznam = [1,3,8,2,10,15,7]

Jak bude vypadat výsledek tištěný na standardní výstup:

3

24

16

…

Příklad 4

```
seznam = [1,3,8,2,10,15,7]


#pouziti cyklu for, cyklus se ukonci na predposlednim prvku seznamu (pomoci len(seznam)-1), aby bylo mozne tento vynasobit s poslednim prvkem seznam.
for i in range(0,len(seznam)-1):
    soucin = seznam[i]*seznam[i+1] #nasobi se vzdy prvek zpracovavany v dane iteraci s prvkem s nasledujicim indexem
    print(soucin)
```

Příklad 5

Na vstupu je dán seznam obsahující prvky různého datového typu. Vytvořte výstupní seznam a uložte do něj pouze prvky datového typu float a integer ze vstupního seznamu.

Navrhněte dvě možná řešení zapsání podmínky.

seznam = [1, 10.5, "medved", "vlk", 100.111, 12, "rys"]

Jak bude vypadat výsledek:

seznam_vystup = [1, 10.5, 10.111, 12]

Příklad 5

```
seznam = [1, 10.5, "medved", "vlk", 100.111, 12, "rys"]

seznam_vystup = [] #vytvori se vystupni promenna v podobe prazdneho seznamu


for prvek in seznam:

    if type(prvek) == int: #podminka testujici zda je zadana promenna dat. typu integer (int)

    seznam_vystup.append(prvek) #append prida zadanou hodnotu na konec vystupniho seznamu

    elif type(prvek) == float: #pokud podminka vyse neni splnena, otestuje se promenna zdali je DT float

    seznam_vystup.append(prvek)
```

Příklad 5

```
seznam = [1, 10.5, "medved", "vlk", 100.111, 12, "rys"]
seznam_vystup = [] #vytvori se vystupni promenna v podobe prazdneho seznamu

for prvek in seznam:
#podminka testujici zda je zadana promenna dat. typu integer NEBO (OR) float
    if type(prvek) == int or type(prvek) == float:
        seznam_vystup.append(prvek)

print(seznam_vystup)
```

Příklad 6

Zjistěte medián čísel zadaných v seznamu a vytiskněte jej na standardní výstup.

medián:

- jedna ze středních hodnot používaných ve statistice.
- je hodnotou, která dělí řadu vzestupně seřazených hodnot na dvě stejně početné poloviny.
- oproti průměru není medián (silně) ovlivňován odlehlými hodnotami vyskytujícími se ve vzorku → viz průměrná mzda versus medián mzdy

Příklad 6

seznam = [2,5,15,8,4,10,7]

Jak bude vypadat výsledek:

7

Poznámka: pokud má řada čísel na vstupu sudý počet hodnot, lze za medián považovat:
a) průměr dvou hodnot nejblíže středu

b) případně jednu ze dvou hodnot nejblíže středu

Příklad 6

```
seznam = [2,5,15,8,4,10,7]
#fce sorted setridi seznam prvku od nejmensiho po nejvetsi, pripadne dle uzivatelem zadaneho klice
seznam_sorted = sorted(seznam)

if len(seznam_sorted) % 2 == 1: #pokud je pocet prvku lichy, bere se prvek uprostred seznamu dle jeho indexu
    median = seznam_sorted[int(len(seznam_sorted)/2)]
else: #pokud pocet prvku neni lichy, musi byt sudy -> pouzije se else
    #v pripade sudeho poctu prvku se pocita prumer dvou prvku uprostred seznamu
    median = (seznam_sorted[int(len(seznam_sorted)/2)] + seznam_sorted[int((len(seznam_sorted)/2))-1]) / 2.0
print(median)
```

Příklad 7

S využitím cyklu vytvořte Fibonacciho posloupnost do čísla 100 a uložte ji do seznamu.

První dvě čísla jsou dána: 0 a 1

Každé další číslo posloupnosti je rovno součtu dvou předešlých čísel.

Jak bude vypadat výsledek:

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]

Příklad 7

```
i = 0
j = 1
fibonacci = [] #vytvarim prazdny seznam pro ulozeni vysledku
#do seznamu ukladam vstupni cisla
fibonacci.append(i)
fibonacci.append(j)

#v cyklu while pocitam a v podmince testuji soucet aktualni hodnoty i a j
while i+j < 100:
    fibonacci.append(i+j) #soucet i a j zapisi do seznamu
    docasne_i = i #hodnotu i si docasne ulozim do pomocne promenne
    #prohodim hodnoty i a j, s vyuzitim pomocne promenne pro i
    i = j
    j = docasne_i+j
print(fibonacci)
```

Děkuji za pozornost

Michal Kačmařík

michal.kacmarik@vsb.cz

www.vsb.cz