6 –TOM 2 – SON / 2023 - YIL / 15 - FEVRAL

PYTHON TILIDA FUNKSIYALAR BILAN ISHLASH

Eshonqulova Naima
To'laboyeva Nazokat
Yangiqo'rg'on tuman 2-son kasb-hunar maktabi

Python tilida ko‘plab tipik masalalar uchun funksiyalar (ularni usullar deb atash mumkin) joriy qilingan. Biz hozirgacha bunday funksiya-usullarning ayrimlari bilan tanishib chiqdik. Tabiiyki, turli mazmundagi masalalar uchun dastur ishlab chiqishda bu funksiyalar yetarli bo‘lmasligi mumkin. Python tilida dasturchilar uchun ehtiyojga ko‘ra yangi funksiyalarni yaratish va foydalanish imkoniyatlar mavjud.

Funksiyalar kichik bir masalani hal qilishga qaratilgan ma’lum bir amallar ketma-ketligini turli qiymatlar uchun qayta-qayta hisoblashga to‘g‘ri kelgan hollarda tashkil qilinadi. Bu holat dasturchining keyingi ishlarini osonlashtiradi. Funksiyalar formal o‘zgaruvchilar (argumentlar) uchun tashkil qilinadi.

Shunday masalalar mavjudki, ularni hal qilish jarayonida bir nechta kichik masalalarga ajratish mumkin bo‘ladi. Agar hosil bo‘lgan masalalar bitta sinfga tegishli bo‘lsa yana ham yaxshi. Bu holda bir sinfga tegishli bo‘lgan har bir masalaga alohida dastur yozish o‘rniga, ulardan bittasi uchun argumentlar deb ataluvchi formal o‘zgaruvchilar tanlanadi va masalaning ana shu o‘zgaruvchilar uchun yechish buyruqlari ketma-ketligi ishlab chiqiladi. Funksiyalarni bunday masalalarga nisbatan qo‘llash ham mumkin. Bu holda umumiy masalaning yechim lari kichik masalalar yechimlarining kombinatsiyalari shaklida ifodalanadi.

Bitta dastur tarkibidagi barcha funksiyalar ishini boshqaradigan dastur asosiy deb ataladi.

Agar dasturda funksiyalardan foydalanish rejalashtirilgan bo‘lsa, har bir funksiya uchun asosiy dasturdan o‘tadigan ma’lumotlar (parametrlar) ro‘yxati hamda qiymati funksiyadan asosiy dasturga qaytishi lozim bo‘lgan o‘zgaruvchi ko‘rsatb qo‘yiladi.

Funksiyaga undagi argumentlar o‘rniga mos parametrlar yordamida (ro‘yxatdagi 1-argument o‘rniga 1-parametr, 2-argument o‘rniga 2-parametr va hokazo) murojaat qilish mumkin.

Funksiyaga murojaat qilinganda, asosiy dasturning bajarish jarayoni to‘xtaydi va kompyuter funksiyani bajara boshlaydi. Funksiyadagi barcha buyruq-lar parametrlar uchun to‘la bajarilgandan so‘ng, kompyuter yana asosiy dasturning kelgan qismidan boshlab navbatda turgan buyruqlarni bajarishda davom etadi.

Funksiyalar bilan ishlash imkoniyatiga ega bo‘lish uchun ularga birinchi marta murojaat qilishdan avval aniqlangan bo‘lishi lozim. Bunda funksiya nomi va argumentlar ro‘yxati funksiyaning umumiy belgisi hisoblanadi va unga ana shu belgiga ko‘ra murojaat qilinadi.

Funksiyani aniqlash uning umumiy belgilari va jismini ko‘rsatishdan iborat bo‘lib, quyidagicha tuzilmaga ega:

def funksiya_nomi (ro‘yxat);

funksiya_jismi

Bu yerda *funksiya_nomi* o‘zgaruvchi-identifikatorni anglatadi. Bu nom boshqa o‘zgaruvchilar kabi takrorlanmas bo‘lishi lozim; *ro‘yxat* argumentlarni o‘z ichiga olishi yoki bo‘sh bo‘lishi ham mumkin; *funksiya_jismi* turli amallar va ko‘rsatmalar ketma-ketligidan iborat bo‘lib, odatda yuqorida ta’kidlanganidek, alohida olingan kichik bir masalani hal qilishga qaratiladi. Jismning so‘nggi bajariladigan buyrug‘i *return* bo‘lib, u boshqaruvni funksiyaga murojaat qilish nuqtasiga uzatish (qaytarish) amalini bajaradi. Bu buyruq umumiy ko‘rinishda

return ifoda

ko‘rinishida yoziladi. Bu operatordan keyin ko‘rsatilgan ifoda funksiyadan asosiy dasturga uzatiladigan qiymatni belgilab beradi.

Return operatori funksiyadan hech qanday qiymatni qaytarish ko‘zda tutilmagan hollarda yozilmaydi. Bitta funksiya jismida bir nechta *return* buyruqlaridan foydalanish mumkin.

Quyida funksiyalarni e’lon qilishga namunalar keltirilmoqda:

```
def pr ( )
print(‘salom’)
def f(x)
y=x+5
return y
def f(x, y, z)
return x+y+z
```

Keltirilgan *a*-namuna ekranga “salom” matnini chiqarish bilan o‘z ishini tugatadi. Hech qanday ma’lumot qaytarmaydi; *b* - namuna esa o‘ziga parameter sifatida uzatiladigan *x* – qiymati uchun *y=x+5* ifodaning qiymatini hisoblaydi va shu qiymatni asosiy dasturga qaytaradi; *c*-namuna esa parameter sifatida uzatilgan *x, y* va *z* – larning qiymatlari uchun asosiy dasturga *x+y+z* ifodaning qiymatini qaytaradi.

Funksiyaga murojaat qilganda parametrlar ro‘yxatga ko‘ra mos o‘rinda turgan argumentlar bilan almashtiriladi va bunda tiplarining o‘zaro mosligi qat’iy nazorat qilinadi. Bunday moslik bo‘lmaganda PYTHON tilida o‘zgaruvchilar tiplarini to‘g‘ridan-to‘g‘ri almashtirish ham ko‘zda tutilgan. PYTHON haqida fikr yuritganda, “tiplarning qat’iy mosligiga alohida e’tibor beriladi.

Funksiyaga murojaat qilish (sodda qilib aytganda, uni chaqirish) oddiy qavslar yordamida amalga oshiriladi. Qavslar ichida esa argumentlar ro‘yxati ko‘rsatiladi:

funksiya_nomi (parametrlar ro‘yxati).

Argumentlar (funksiya argumentlari) va parametrlar o‘rtasidagi moslik rasmiy va argumentlarning ro‘yxatdagi o‘rniga ko‘ra aniqlanadi.

Parametrlar murojaat qiluvchi dastur tomonidan uzatiladi va funksiya jismidagi ko‘rsatmalarni bajarishda ana shu ularning qiymatlaridan foydalaniladi.

Yuqoridagi fikrlarni amaliyotga tatbiq etishga urinib ko‘raylik.

1-masala. Haqiqiy *a, b, c* va *d* sonlari berilgan bo‘lsin. Ularning eng kichigini toping.

Yechish g‘oyasi. Dastlab, *a* va *b* sonlarining eng kichigini *p* bilan, *c* va *d* larning eng kichigini esa *q* bilan belgilaymiz. Ishning yakunida *p* va *q* larning eng kichigi topiladi. Ko‘rinib turibdiki, ikki sonning eng kichigini topish masalasidan 3 marta foydalanilmoqda. Shuning uchun parametr sifatida *n* va *m* sonlarni tanlab olinadi va ularning kichigini topish uchun funksiya tashkil qilinadi. Bu mulohazalarni e’tiborga olib, kodni quyidagicha yoziladi:

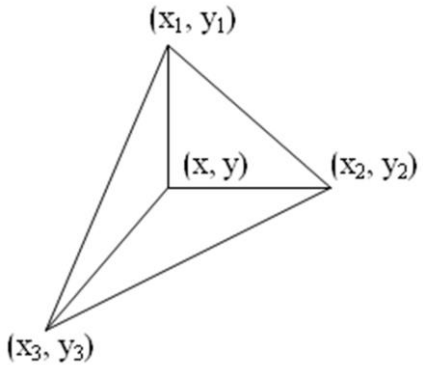
```
def min(n, m) :
if (n<m) :
k=n
else :
k=m;
return k
a=input("Birinchi sonni kiriting ")
b=input("Ikkinchi sonni kiriting ")
c=input("Uchinchi sonni kiriting ")
d=input("To‘rtinchi sonni kiriting ")
p=min(a, b)
q=min(c, d)
print("Berilgan sonlarning eng kichigi =", min(p, q))
```

```
Birinchi sonni kiriting 5
Ikkinchi sonni kiriting 3
Uchinchi sonni kiriting 7
To‘rtinchi sonni kiriting 4
Berilgan sonlarning eng kichigi = 3
>>>
```

Eslatma. *Return* buyrug‘idan bir necha marta foydalanish mumkin bo‘lgani uchun funksiyani

```
def min(n, m)
if n<m :
return n
else :
return m
```

ko‘rinishida ham yozish mumkin.



Python tilida bitta funksiya tarkibida boshqa funksiyalarga ham murojaat qilish mumkin.

2-masala. Uchburchak uchlarining

koordinatalari *(x1,y1), (x2,y2), (x3,y3)* berilgan bo‘lsin. *(x, y)* koordinatali nuqta shu uchburchakka tegishli bo‘la oladimi?

Yechish g‘oyasi. Berilgan uchburchak yuzasi *S* bo‘lsin. Uchburchak uchlarini *(x, y)* koor dinatali nuqta bilan tutashtirib, 3 ta uchburchak hosil qilinadi (7.1-rasm). Ularning yuzalari mos ravishda *S1, S2* va *S3* bo‘lsin. Agar *(x, y)* nuqta berilgan uchburchak ichida yotsa, *S=S1+S2+S3* bo‘ladi. Aks holda, nuqta uchburchak ichida yotmaydi.

Ko‘rinib turibdiki, bu yerda uchlarining koordinatalari ma’lum bo‘lgan to‘rtta uchburchak yuzini hisoblashga to‘g‘ri kelmoqda. Bu koordinatalar uchun formal o‘zgaruvchilarni *(a1, b1), (a2, b2)* va *(a3, b3)* tarzida tanlash mumkin. Uchburchakning tomonlari va yarim perimetrini belgilash uchun mos ravishda *A, B, S* va *P* o‘zgaruvchilar olinadi. Ular masala shartida ko‘rsatilmagani uchun lokal o‘zgaruvchilar hisoblanadi.

Tanlangan formal o‘zgaruvchilarni hisobga olib, uchburchak yuzini topish buyruqlaridan iborat funksiya hosil qilinadi. Funksiyada uchlarining koordinatalari ma’lum bo‘lgan kesma uzunligi uch marta hisoblanishi lozim bo‘lgani uchun kesma uzunligini hisoblash maqsadida alohida funksiya tashkil qilish mumkin. Shundan keyin

(x1, x2, x3, y1, y2, y3), (x, x1, x2, y, y1, y2),
(x, x2, x3, y, y2, y3) , (x, x1, x3, y, y1, y3)

argumentlar uchun yuzani hisoblash funksiyasiga murojaat qilib, uchlari ana shu nuqtalarda yotgan uchburchaklarning *S1, S2, S3* va *S* yuzalari hisoblanadi. Yuzani hisoblash funksiyasi har gal bajarilganda kesma uzunligini hisoblash funksiyasiga uch marta murojaat qiladi. Uchburchaklarning yuzalari topilganidan so‘ng, *S=S1+S2+S3* munosabatning rost yoki yolg‘on bo‘lishiga qarab xulosa chiqariladi. Mazkur jarayon uchun kod quyidagicha yoziladi:

```
import math
def kesma(a1, b1, a2, b2):
return math.sqrt((a2-a1)*(a2-a1)+(b2-b1)*(b2-b1))
def yuza(a1,b1,a2,b2,a3,b3):
n1=kesma(a1,b1,a2,b2)
n2=kesma(a1,b1,a3,b3)
n3=kesma(a2,b2,a3,b3)
p=(n1+n2+n3)/2.0
return round(math.sqrt(p*(p-n1)*(p-n2)*(p-n3)),1)
print('Uchburchak koordinatalarini x1,y1, x2,y2, x3,y3 shaklida kiriting')
x1,y1, x2,y2, x3,y3=map(float,input().split(','))
x,y=map(float,input('koordinatalarini kiriting ').split(','))
s1=yuza(x, y, x1, y1, x2, y2);
s2=yuza(x, y, x1, y1, x3, y3);
s3=yuza(x, y, x2, y2, x3, y3);
```

```
s=yuza(x1, y1, x2, y2, x3, y3);
print(s, s1,s2,s3)
if (s==s1+s2+s3) :
print('tegishli')
else :
print('tegishli emas')
```

```
>>>
Uchburchak koordinatalarini x1,y1, x2,y2, x3,y3 shaklida kiriting
-1,-1, 4,-1, 0,4
koordinatalarini kiriting 0,0
12.5 2.5 2.0 8.0
tegishli
>>>
```

FOYDALANILGAN ADABIYOTLAR:

1. Тохирджон о'Гли А. У., Низомиддинович И. С., Тоджиддин о'Гли Н. С. ИНТЕГРАЛЬНЫЙ ТЕНГЛАМАЛАРНИ MATHCAD ДАСТУРИДА ТАКРИБИЙ ЙЕЧИШ УСУЛЛАРИ //ИННОВАЦИИ В СОВРЕМЕННОЙ СИСТЕМЕ ОБРАЗОВАНИЯ. – 2022. – Т. 2. – №. 18. – С. 880-883.

2. Саматов Б. Т., Хорилов М. А., Иномиддинов С. Н. Дифференциальный LG-игра с “Линией жизни”. – 2019.

3. Umaralieva N. T., qizi Uralova S. I. EVASION PROBLEM FOR THE SECOND ORDER DIFFERENTIAL GAME //Scientific Bulletin of Namangan State University. – 2019. – Т. 1. – №. 4. – С. 8-12.