Lesson2. Python1

February 9, 2024

1 Python dasturlash tili

2 Reja

1. Dasturlash tili
2. Python dasturlash tili va uning imkoniyatlari
3. Sonlarni mashina xotirasida tasvirlash
4. Python tilida o'zgaruvchi va uning turlari
5. Kiritish va chiqarish funksiyalari
6. Numunaviy masalalar
7. Mustaqil ishlash uchun vazifalar
8. E'lon

3 Dasturlash tili nima

Dasturlash tili hisoblash mashinasiga (kompyuter) buyruqlarni ketma-ketiligini berishga muljallangan qattiy gramatik qoidalarga ega notabiiy tildir. Ushbu buyruqlar ketma-ketligini biz kod (code) deb nomlaymiz va ba'zida dastur kodi yoki matni deb ham nomlashimiz mumkin. Chunki kod oddiy mant fayllardan iborat bo'ladi odatda. Lekin bundan mustasno hollatlar ham mavjud.

4 Python dasturlash tili va uning imkoniyatlari

4.1 Python dasturlash tili

Python - mashhur dasturlash tili. U Guido van Rossum tomonidan yaratilgan va 1991 yilda chiqarilgan.

U quyidagilar uchun ishlatiladi:

1. veb-ishlab chiqish (server tomonida),
2. dasturiy ta'minotni ishlab chiqish,
3. matematika,
4. tizim skripti.

4.2 Python nimalar qila oladi?

1. Python veb-ilovalarni yaratish uchun serverda ishlatilishi mumkin.
2. Python ish oqimlarini yaratish uchun dasturiy ta'minot bilan birga ishlatilishi mumkin.

3. Python ma'lumotlar bazasi tizimlariga ulanishi mumkin. Shuningdek, u fayllarni o'qishi va o'zgartirishi mumkin.
4. Python katta ma'lumotlarni qayta ishlash va murakkab matematika amallarni bajarish uchun ishlatilishi mumkin.
5. Python tez prototiplash yoki ishlab chiqarishga tayyor dasturiy ta'minotni ishlab chiqish uchun ishlatilishi mumkin.

4.3 Nega aynan Python tili

1. Python turli platformalarda ishlaydi (Windows, Mac, Linux, Raspberry Pi va boshqalar).
2. Python ingliz tiliga o'xshash oddiy sintaksisga ega.
3. Pythonda ishlab chiquvchilarga boshqa dasturlash tillariga qaraganda kamroq qatorli dasturlar yozish imkonini beruvchi sintaksis mavjud.
4. Python tarjimon tizimida ishlaydi, ya'ni kod yozilishi bilanoq bajarilishi mumkin. Bu prototip yaratish juda tez bo'lishi mumkinligini anglatadi.
5. Pythonni protsessual, ob'ektga yo'naltirilgan yoki funktsional usulda yozish mumkin.

5 Mashina xotirasida qiymatlarni saqlash

1. Sonlarni ikkilik sanoq sistemasida saqlaymiz, misol, 23 soni 10111 bo'ladi.
2. Agar manifiy son bo'lsachi, masalan, -23
3. IEEE 754 standarti
4. Butun sonlarni saqlash qoidasi

5.1 Butun sonlarni saqlash qoidasi

Tassavur qilaylik soni 32 bitlik xotiraga saqlamoqchimiz:

1. ishora (musbat yoki manifiy uchun) 1 bit;
2. sonning o'zi uchun esa 31 bit

Ishora	son
1	31

5.1.1 Tinglovchilarga savollar

1. 2 bitli xotira qanday oraliqdagi butun sonlarni qabul qiladi?
2. 4 bitli bo'lsachi?
3. 6 bitli bo'lsachi?
4. 8 bitli bo'lsachi?

5.2 Haqiqiy sonlarni saqlash qoidasi

Tassavur qilaylik soni 32 bitlik xotiraga saqlamoqchimiz:

1. ishora (musbat yoki manifiy uchun) 1 bit;
2. exponenta uchun 8 bit;
3. hamda, mantissa uchun qolgan 23 bit.

Ishora	Exponenta	Mantissa
1	8	23

6 Namuna

Son	ko'paytiriluvchi son (2)	Ikkilik ko'rinish
$1+\frac{1}{3}$	$\frac{2}{3}*2$	1
$0+\frac{2}{3}$	$\frac{1}{3}*2$	10
$1+\frac{1}{3}$	$\frac{2}{3}*2$	101
$0+\frac{2}{3}$	$\frac{1}{3}*2$	1010
$1+\frac{1}{3}$	$\frac{2}{3}*2$	10101
...	...	...
...	...	...
$1+\frac{1}{3}$	$\frac{2}{3}*2$	10101010101010101010101

6.0.1 Tinglovchilarga savollar

Quyidagi sonlarni 32 bitli xotirada saqlash uchun ikkilik kodini hosil qiling

1. -2.23
2. 400.3
3. -6.54
4. 80.3

7 Python tilida o'zgaruvchi va uning turlari

7.1 O'zgaruvchilar

Agar xotiraga biror qiymatni joylashtirmoqchi bo'lsak, u holda bizga kerakli bo'lgan qismga murojaat qiluvchi nom kerak bo'ladi va bu nomni biz o'zgaruvchi deb ataymiz. Bu o'zgaruvchi mashinaning operartiv xoritasi (RAM: Random Access Memory)dan joy egallaydi va biz shu o'zgaruvchiga yangi qiymat yuklashimiz bilan eski qiymatni o'chiradi va yangisini ikkilik (ikkilika o'tkazish jarayoni haqida biz umuman bosh qotirmaymiz) ko'rinishga o'tkazib yozadi va bizda boshqa eski qiymatni olishga imkon bo'lmaydi. Quyida bir qator namunalar berilgan.

7.2 O'zgaruvchi e'loni

[ ]:
```
a = 4
```

7.3 Izohlar

[ ]:
```
# Bu izoh qatori: bu qatorni mashini buyruq sifatida qabul qilmaydi.
# Ushbu kod ishlashi natijasida hech narsa ro'y bermaydi
a = 4
```

7.4 O'zgaruvchi turlari

Yuqorida butun turdagi o'zgaruvchilar bilan ishladik, bundan tashqari bir qator turlar mavjud:

1. int - butun
2. float - haqiqiy
3. str - matn

Matni saqlovchi o'zgaruvchilarni e'lon qilishda, o'zgaruvchiga zarur matn ' yoki " ichida beriladi. Misollar:

7.5 O'zgaruvchilarga misollar

[ ]:
```
# Yoshi 14
age = 14
# Familiyasi. Matni ikkita " (qo'shtirnoq) ichiga yozdik
fam = "Abdusamatov"
# Ismi. Matni ikkita ' (tirnoq) ichiga yozdik
# Diqqat ikkisining ham umuman bir-birdan farqi yo'q.
ism = 'Doniyor'
# Og'irligi kgda
# sonning butun va haqiqiy qismini ajratish uchun . (nuqta)dan foydalanamiz
vazn = 78.5
# chop etish
print(fam, ism, age, vazn)
```

```
Abdusamatov Doniyor 14 78.5
```

7.6 O'zgaruvchi turini aniqlash

Yuqoridagi kodga e'tibor bersak, biz umuman o'zgaruvchi turini aytmadik, lekin Python o'zi bu narsani berilgan qiymatga qarab aniqlab oladi. Shuning uchun ham ushbu dinamik turga ega dasturlash tilidir. Bundan tashqari, bitta o'zgaruvchiga dastur ishlash davomida bir nechta har xil qimay bersa ham oxirgisni olib, qolganlarini unutgan holda ishlayveradi. O'zgaruvchi turini bilmoqchi bo'lsak, biz `type` funksiyasidan foydalanamiz. Misollar.

[ ]:
```
# a o'zgaruvchisi e'loni va unga 4 qiymat berildi
a = 4
# a o'zgaruvchisining turi
type(a) # natija int
```

[ ]:
```
int
```

[ ]:
```
# Birinchi a turi int
a = 4
# Endi uning turi float
a = 3.14
type(a)
```

[ ]: float

[ ]:
```
# Birinchi a turi float
a = 3.14
# Endi uning turi str
a = 'salom'
type(a)
```

[ ]: str

7.7 Farqlar

Yuqorida 3 ta o'zgaruvchi turini e'lon qilishni va ular bilan ishlashni o'rgandik. Lekin ko'pchilik boshlovchilar adashadigan bir holat bor. Masalan quyidagi kodga va uning natijasiga e'tibor beraylik.

[ ]:
```
# age o'zgaruvchisiga 25 raqami yozildi
age = 25
# age matni chop qilinyapti, bu age o'zgaruvchisi emas!!!
# Natija: age
print('age')
```

```
age
```

Yuqoridagi kodda biz age nomli o'zgaruvchi e'lon qildik va unga 25 qiymatni o'zlashtirdik. Lekin 5-qataorda biz `age` degan matn turidagi qiymatni chop qildik, `age` o'zgaruvchisi emas. Quyida esga ikki holni ham qayta ko'ramiz.

[ ]:
```
# age ga 25 raqami yozildi
age = 25
# age o'zgaruvchising qiymati chop qilinyapti
print(age)
# age matni chop qilinyapti
print('age')
```

```
25
age
```

7.8 O'zgaruvchi nomlarni aniqlashga cheklovlar va tavsiyalar

Biz xotiraning bir qismini band qilish va shu qismga o'zimizga kerak bo'ladigan qiymatni yozib qo'yishimiz uchun `o'zgaruvchi` tushunchasi bilan tanishdik. Lekin biz ma'lum bir qattiy qoidalarga ega bo'lgan dasturlash tili bilan ishlar ekanmiz, bizda o'zgaruvchilarni e'lon qilishda albatta cheklovlarga egamiz. Quyida ushbu qoidalarning ba'zilari:

1. o'zgaruvchi har doim katta-kichik (masalan, katta harf deb `A`, kichik deb esa `a`) harflar va tag chiziq `_` bilan boshlanadi;
2. o'zgaruvchining qolgan belgilari katta-kichik harflar, raqamlar (0, 1, 2, ..., 9) va tag chiziq `_` belgisi bo'lishi mumkin.

3. har doim katta va kichik harflardan ikki xil o'zgaruvchi paydo bo'ladi. Masalan, `age`, `Age`, `AGE` lar hammasi alohida o'zgaruvchilar, hattoki ma'nosi bir bo'lsa ham.

Quyida birinchi mumkin bo'lgan o'zgaruvchilarga misollarni ko'ramiz, keyin esa uning teskarisiga.

[ ]:
```
# Mumkin bo'lgan o'zgaruvchilar
# bolaning yoshi
boy_age = 25
# o'rtacha yosh
mean_age = 22
# Uning familiyasi
his_familiy = 'Abdurahmonov'
```

[ ]:
```
# Katta-kichik harflaridan farq
# qiluvchi o'zgaruvchilar
# age o'zgaruvchisi
age = 15
# Bu o'zgaruvchi bilan age o'zgaruvchisi
# ikkita alohida o'zgaruvchi
Age = 25
```

[ ]:
```
# Taqiqlangan o'zgaruvchilar
12boy_age = 25
mean age = 22
his familiy = 'Abdurahmonov'
```

```
  Cell In[11], line 2
    12boy_age = 25
     ^
SyntaxError: invalid decimal literal
```

7.9 Tavsiyalar.

Dastur oson o'qilishi va tushunarli bo'lishi uchun biz har doim o'zgaruvchilarga ularning saqlashi mumkin bo'lgan qiymatiga qarab nom berishimiz zarur. Masalan yuqorida, `age` o'zgaruvchisiga doimiy inson (yoki boshqa narsaning) yoshini saqlashimiz maqsadga muvofiq bo'ladi. Agar buning uchun bir nechta so'z zarur bo'lsa, unda tag `_` chiziq bilan yoki har bir so'zning bosh harifini katta harflar orqali ifodalash bilan erishishimiz mumkin. Masalan, `mean_age` - o'rtacha yosh. Yana bir boshqa tavsiya esa, albatta ingliz tilidagi so'zlardan foydalanishdir. Oxirgisi esa agar o'zgaruvchi nomi haddan tashqari uzun bo'lib, qayta yozishga halaqit qilsa unda uni qisqartirib yozish mumkin. Misol uchun, `familiy_name` o'rniga `fam_name` yoki `number_of_objects` o'rniga esa `n_objs`

7.10 Funksiya qaytargan qiymatni olish va ularni chop qilish

Biz shu vaqtgacha faqat ikkita funksiya bilan tanishdik. Shunda ham, faqat ular bilan qisman ishlashni o'rgandik. Shuning uchun ushbu bo'limda, funksiyaning qiymat qaytarishini va ularning

natijalarini boshqa o'zgaruvchi saqlab, keyinchalik ulardan foydalanishni o'rganamiz. Python dasturlash tilida har bir funksiya qiymat qaytaradi. Lekin, odatada, dasturlash tillarida biz funksiyalar qiymat qaytarishi yoki qaytarmasligiga qarab ikki turga ajratamiz. Pythonda esa bu holat faqat `None` o'zgarmas qiymati bilan amalga oshiriladi. Hozir keling bu qismlarga chuqur kirishmasdan, uning o'rniga, sodda misollar bilan mavzuni tushunaylik. Bu kabi holatlarni esa kelgusida ko'rib chiqaylik. Keling `type` funksiyasi qaytargan qiymatni, biror o'zgaruvchiga yozaylikda, keyin uni `print` funksiyasi yordamida chop qilaylik quyidagi kodada berilgan kabi.

[ ]:
```
# age o'zgaruvchisiga 25 yozish
age = 25
# age o'zgaruvchisi turini olish
age_type = type(age)
# va natijani chop qilish
print(age_type)
```

```
<class 'int'>
```

8 Kiritish va chiqarish funksiyalari

8.1 `print` funkiyasi

Ushbu bo'limda biz `print` funksiyasi bilan batafsil tanishishga harakat qilamiz. Chunki biz ushbu funkisyadan ushbu kitob davomida juda ko'p o'rinlarda foydalanimiz. Ushbu funksiya 5 qiymat (argument deb ham nomlaymiz) qabul qiladi, ular: 1. `*values` - ushbu argumentga biz chop qilmoqchi bo'lgan qiymatlarimizni beramiz, masalan:

[ ]:
```
age = 14
fam = "Abdusamatov"
ism = 'Doniyor'
vazn = 78.5
# chop etish
print(fam, ism, age, vazn)
```

```
Abdusamatov Doniyor 14 78.5
```

2. sep - bu satr (matn) turida qiymat qabul qilib, `*values` argumentidagi qiymatlarni o'rtasiga ajratuvchi sifatida yoziladi. Shuning uchun ham uning nomi `sep` ya'ni `separator` ning qisqartmasi. Quyidagi misolni diqqat bilan o'rganing.

[ ]:
```
age = 14
fam = "Abdusamatov"
ism = 'Doniyor'
vazn = 78.5
# turli xil chop etishlar
# ajratuvchi sifatida ustun '|' belgisidan foydalanimiz
print(fam, ism, age, vazn, sep='|')
# ajratuvchi sifatida ustun '|' belgisidan foydalanimiz
# va orasiga bo'sh joy ham qo'shamiz
print(fam, ism, age, vazn, sep=' | ')
```

```
Abdusamatov|Doniyor|14|78.5
Abdusamatov | Doniyor | 14 | 78.5
```

3. `end` - bu argument `print` funksiyasi ishlagandan keyin qanday qiymat qo'yishni aniqlaydi, odatda keyingi qatorga tushish qiymat sifatida turadi. Yuqoridagi kodga e'tibor bersak, unnda ikki chop qilishimiz (`print` fuknisyasini birinchi va ikkinchi marta chaqirgan qism) natijalari alohida ikki qatorga yozilgan. Keling bir tekshirib ko'ramiz.

[ ]:
```
# Yoshi 14
age = 14
fam = "Abdusamatov"
ism = 'Doniyor'
vazn = 78.5
# ushbu koddan keyin albatta keygi chop qilish natijalari yangi qatordan␣
 ↪yoziladi
print(fam, ism, age, vazn, sep='|')
# Bunda keyingi chop qilish bilan buning o'rtasida faqat bo'sh joy bo'ladi
print(fam, ism, age, vazn, sep=' | ', end=' ')
# orada bo'sh joy bor
print(fam, ism, age, vazn, sep=' | ', end=' ')
```

```
Abdusamatov|Doniyor|14|78.5
Abdusamatov | Doniyor | 14 | 78.5 Abdusamatov | Doniyor | 14 | 78.5
```

8.2 `input` kiritish funksiyasi

Qiymatlarni klaviaturadan kiritish uchun biz `input` funskiyasidan foydalanamiz. Uning bitta parameteri bo'lib, u faqat oynaga nima kirataypgan ekanligimizni bildirib turadi va u parameter turi `str` bo'ladi, ya'ni matn.

[ ]:
```
age = input('Yoshingizni kiriting')
print(age, type(age))
# age ning turi str shuning uchun uni int turiga
# o'tkazish zarur
age = int(age)
print(age, type(age))
```

```
43 <class 'str'>
43 <class 'int'>
```

9 Namunaviy masalalar

10 Mustaqil ish uchn vazifalar

11 E'lon