Python programming language

Week 2. Python variables, data type, comments

Alisher Ismailov
Lecturer

Department of Information technology
Andijan Branch of Tashkent Financial Institute
Email: alisherismailov534@gmail.com

Python dasturlash tili

2-Mavzu. Python o`zgaruvchilari, berilganlar turlari, izoh qoldirish

Ma’ruzachi: Alisher Ismoilov

Axborot texnologiyalari kafedrasi

Toshkent moliya instituti Andijon filiali

Elektron pochta: alisherismailov534@gmail.com

1-Mavzu. Python o`zgaruvchilari, berilganlar turlari, izoh qoldirish

Reja:

1. Python dasturlash tilida Izoh qoldirish
2. O‘zgaruvchi (variable)
3. Berilganlar turi (data types)
4. Python dasturlash tilida raqam berilganlar turlari
5. Berilganlar turini o‘zgartirish (Type Conversion)
6. Boolean berilganlar turi

Python dasturlash tili izoh qoldirish

Izoh dasturlashda quyidagi maqsadlarda foydalaniladi:

1. Izoh Python kodini tushuntirish uchun ishlatilishi mumkin.
2. Dasturiy kodni yanada osonroq o‘qilishi uchun izohlardan foydalanish mumkin.
3. Izohlar kodni sinab ko‘rishda bajarilishini oldini olish uchun ishlatilishi mumkin.

Python dasturlash tili izoh qoldirish

- Izohlar # belgisi bilan boshlanadi va Python dasturlash tili ularni e’tiborsiz qoldiradi.

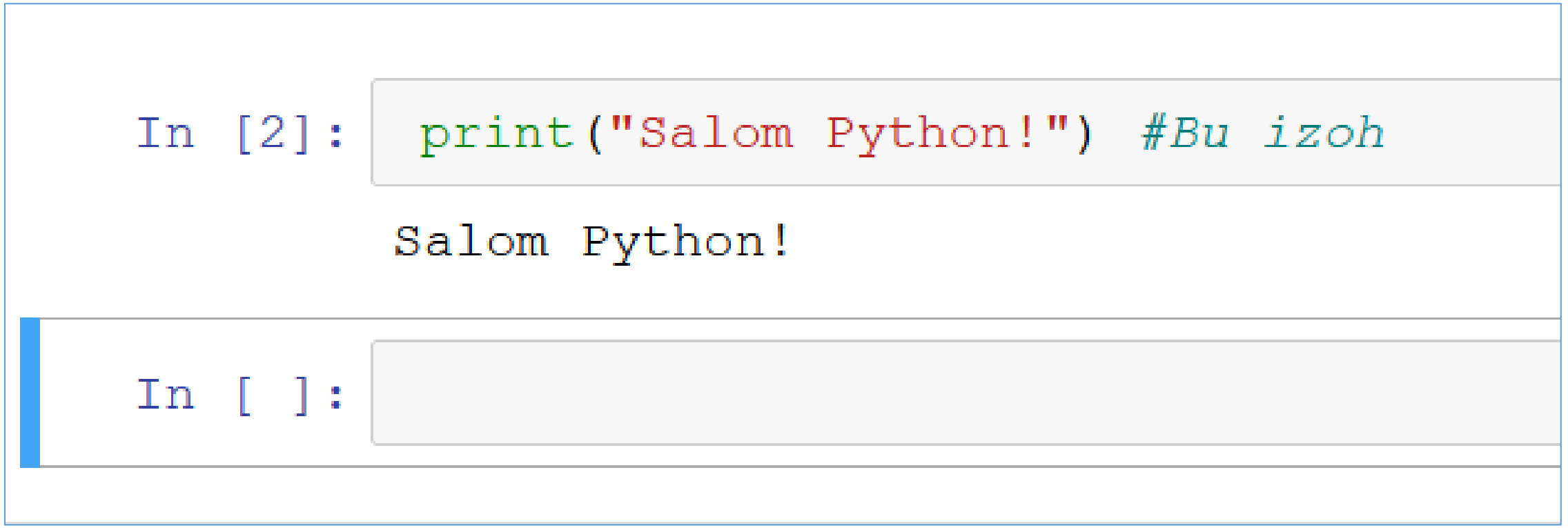
```
In [1]: #Bu izoh qoldirish
        print("Salom Python")

        Salom Python

In [ ]:
```

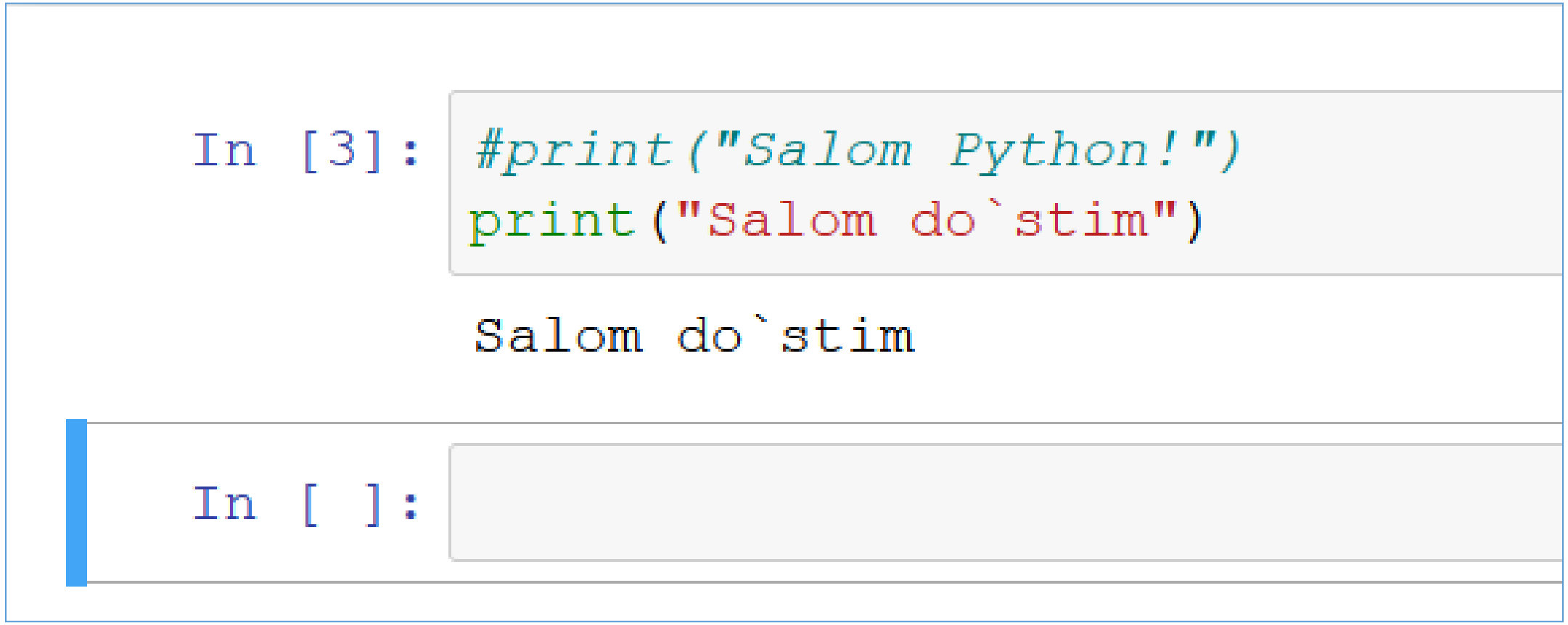
Python dasturlash tili izoh qoldirish

- Izohlar matn oxirida joylashtirilishi ham mumkin va Python dasturlash tilida qatorning qolgan qismi e’tiborsiz qoldiradi:



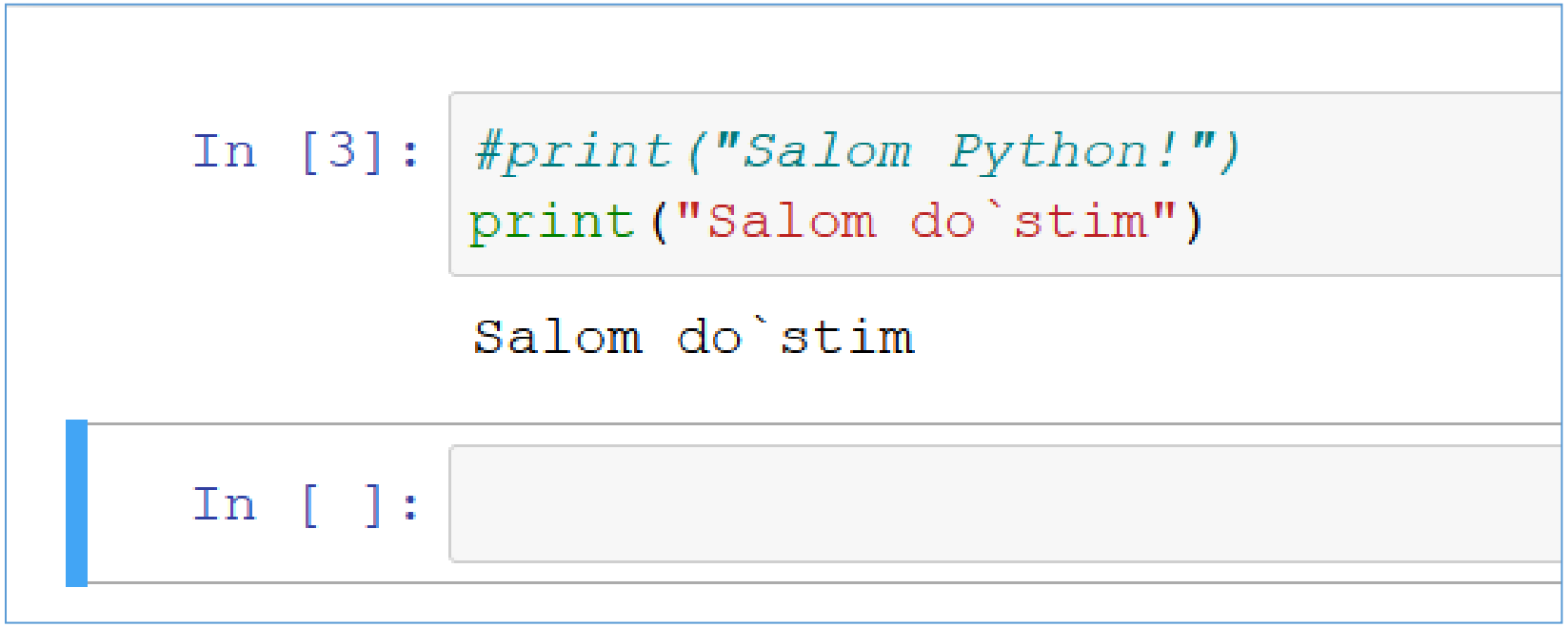
Python dasturlash tili izoh qoldirish

- Izoh kodni tushuntiruvchi matn bo‘lishi shart emas, u Python dasturlash tilida kodni bajarishini oldini olish uchun ham ishlatilishi mumkin:



Python dasturlash tili izoh qoldirish

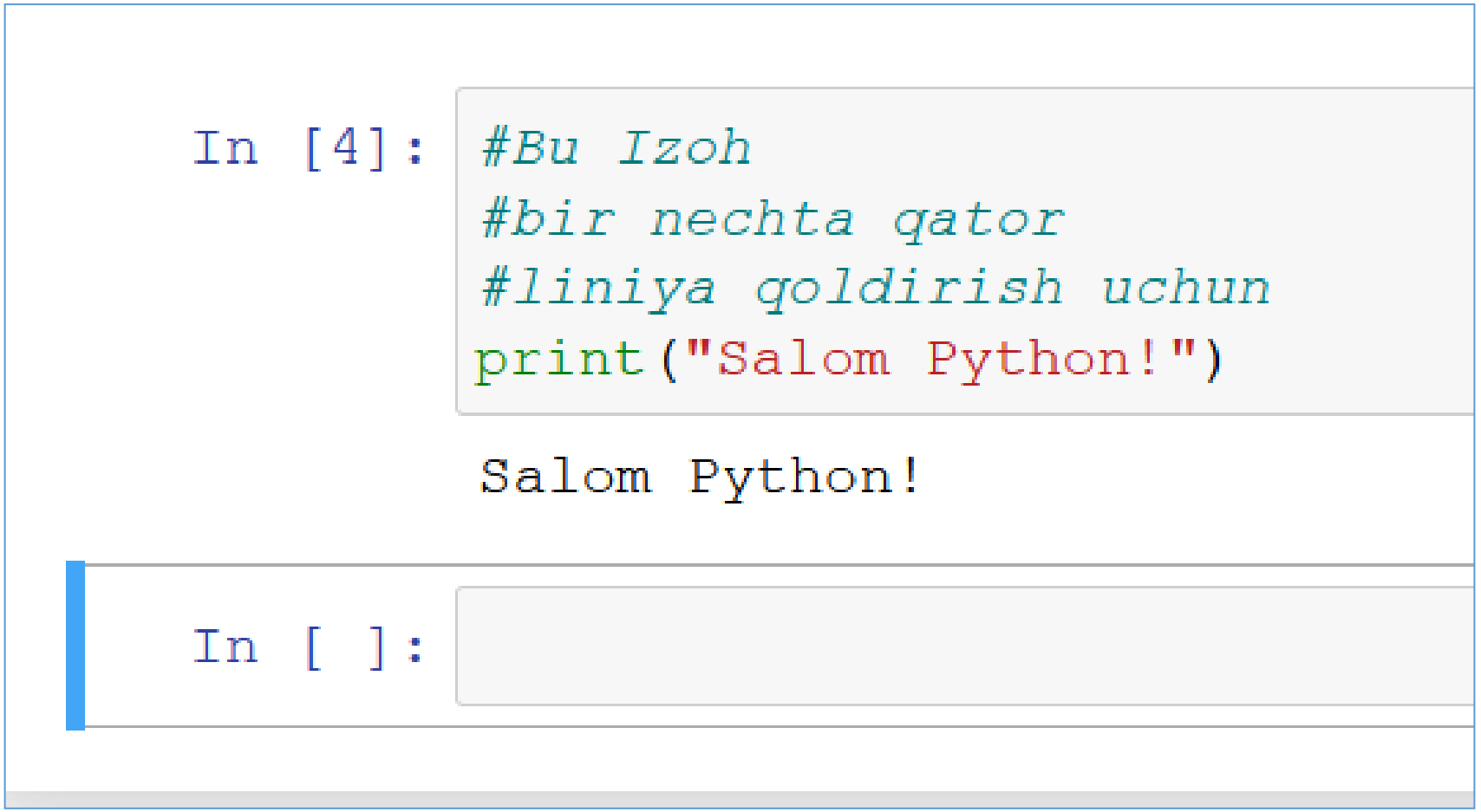
- Izoh kodni tushuntiruvchi matn bo‘lishi shart emas, u Python dasturlash tilida kodni bajarishini oldini olish uchun ham ishlatilishi mumkin:



- Suratda ko‘rib turganimizdek **print(“Salom Python”)** qatori e’tiborsiz qoldirildi sababi bu qatorning boshiga # belgisi orqali uni izoh qilib belgiladik.

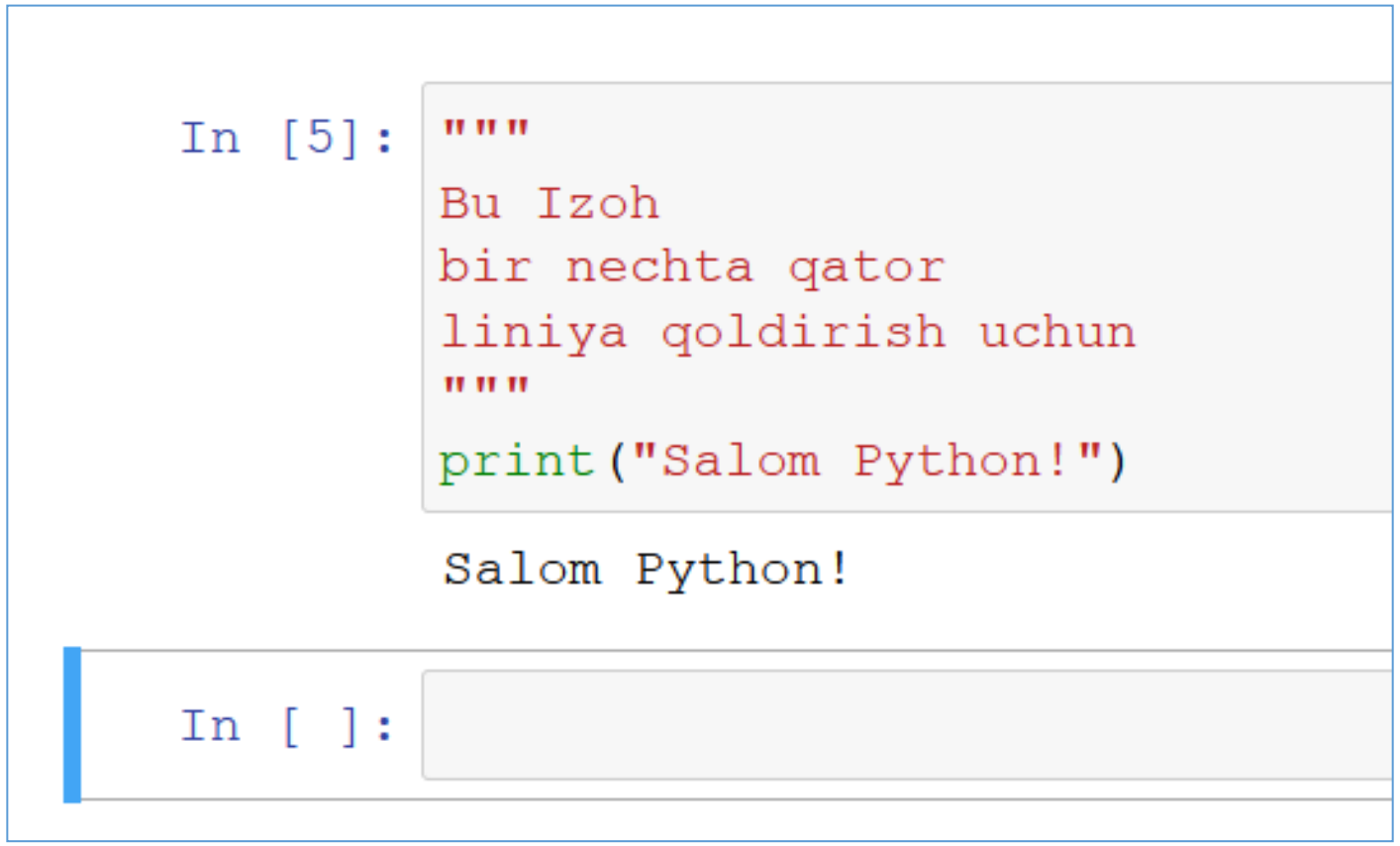
Python dasturlash tili izoh qoldirish

- Ko‘p qatorli izoh qoldirishning birinchi usuli bu har bir qatorga # belgisini qo‘shishingiz mumkin



Python dasturlash tili izoh qoldirish

• Ko‘p qatorli izoh qoldirishni ikkinchi usuli uchta qo‘shtirnoq **“””izoh matni joylanadi“””** belgisidan foydalanish.



Python dasturlash tilida o`zgaruvchilar

Python dasturlash tilida **o‘zgaruvchi** qiymatlarni saqlash uchun ajratilgan xotira joyidir. Boshqacha qilib aytganda, python dasturidagi o‘zgaruvchi ma’lumotlarni qayta ishlash uchun kompyuterga beradi. Python dasturlash tilidagi har bir qiymat berilganlar turiga ega.

Python dasturlash tilida turli xil berilganlar turlari mavjud, bular raqamlar, ro‘yxat, kortej, stringlar, lug‘at va boshqalar hisoblanadi. O‘zgaruvchilarga nom berishda ba’zi qoidalar mavjud.

Python dasturlash tilida o`zgaruvchilar

Python dasturlash tilida o‘zgaruvchilarni nomlash qoidalari:

- O‘zgaruvchi nomi harf (a-zA-Z) yoki pastki chiziq (_) bilan boshlanishi kerak.
- O‘zgaruvchi nomida pastki chiziqdan (_) boshqa maxsus belgilarga ruxsat berilmaydi.
- O‘zgaruvchilar katta-kichik harflarda belgilansa bir-biridan farqli hisoblanadi.
- O‘zgaruvchi nomiga raqamlar ham kiritish mumkin, lekin o‘zgaruvchi nomi raqam bilan boshlanmaydi.
- O‘zgaruvchi nomi Python dasturlash tilida ishlatiladigan kalit so‘zi bo‘lmasligi kerak.

Python dasturlash tilida o`zgaruvchilar

- Misol: 3 ta o‘zgaruvchi yaratamiz va ularni har birini ekranga chop etamiz print() funksiyasi orqali.

```
In [2]: _dasturlashTili = "Python"
        soha2022 = "kompyuter lingivistika"
        yil_ = 2022

        print(_dasturlashTili)
        print(soha2022)
        print(yil_)
```

```
Python
kompyuter lingivistika
2022
```

Python dasturlash tilida o`zgaruvchilar

- Python dasturlash tili bir qatorda bir nechta o‘zgaruvchilarga qiymatlarni belgilash imkonini ham beradi:

```
In [5]: x, y, z = "Olma", "Gilos", "Shaftoli"
        print(x)
        print(y)
        print(z)

        Olma
        Gilos
        Shaftoli
```

Python dasturlash tilida o`zgaruvchilar

- Va bir qatorda bir nechta o‘zgaruvchilarga bir xil qiymatni belgilashingiz ham mumkin:

```
In [6]: x = y = z = "Olma"
        print(x)
        print(y)
        print(z)
```

```
Olma
Olma
Olma
```

Python dasturlash tilida o`zgaruvchilar

- Matn va o‘zgaruvchini birlashtirish uchun Python dasturlash tilida + belgisidan foydalaniladi:

```
In [7]: x = "oson"
        print("Python dasturlash tili " + x)

        Python dasturlash tili oson
```

Python dasturlash tilida o`zgaruvchilar

- O‘zgaruvchini boshqa o‘zgaruvchiga birlashtirish uchun + belgisidan ham foydalanishingiz mumkin:

```
In [8]: x = "Python dasturlash tili "
        y = "zo`r"
        z =  x + y
        print(z)

        Python dasturlash tili zo`r
```

Python dasturlash tilida o`zgaruvchilar

- Raqamlar uchun + belgisi matematik operator sifatida ishlaydi:

```
In [9]: x = 5
        y = 10
        print(x + y)

        15
```

Python dasturlash tilida o`zgaruvchilar

• Agar siz matn va raqamni + operatori orqali birlashtirmoqchi bo‘lsangiz, Python dasturlash tili sizga xato xabarini ko‘rsatadi sababi raqam va matn bir-biriga qo‘shilishi mumkin emas:

```
_dasturlashTili = "Python"
soha2022 = "kompyuter lingivistika"
yil_ = 2022

print(_dasturlashTili+yil_)
```

```
---------------------------------------------------------
TypeError                                 Traceback (mo
~\AppData\Local\Temp/ipykernel_6668/2257627967.py in <m
      3 yil_ = 2022
      4
----> 5 print(_dasturlashTili+yil_)

TypeError: can only concatenate str (not "int") to str
```

Python dasturlash tilida o`zgaruvchilar

Pythonda funksiyadan tashqarida e’lon qilingan o‘zgaruvchi **global** o‘zgaruvchi sifatida belgilanadi. Bu global o‘zgaruvchiga funksiyaning ichida yoki tashqarisida kirish mumkinligini anglatadi.

Python dasturlash tilida global o`zgaruvchilar

• Quyidagi python kodida biz **global** o‘zgaruvchi sifatida ‘**x**’ ni yaratdik va global ‘**x**’ o‘zgaruvchisini chop etish uchun **funksiya()** yaratdik. Va biz **x** qiymatini chop etish uchun yaratilgan **funksiya()**ni chaqiramiz (yuqoridagi suratda ko‘rsatilganidek).

```
In [14]: x = "global"

         def funksiya():
             print("x o`zgaruvchisi funksiya ichida:", x)


         funksiya()
         print("x o`zgaruvchisi funksiya tashqarisida:", x)

         x o`zgaruvchisi funksiya ichida: global
         x o`zgaruvchisi funksiya tashqarisida: global
```

Python dasturlash tilida Mahalliy (lakal) o‘zgaruvchilar

- Funksiya ichida yoki lakal doirada e’lon qilingan o‘zgaruvchi **lakal** o‘zgaruvchi sifatida belgilanadi.

```
In [18]: def funksiya():
             x = "global"
             print("x o`zgaruvchisi funksiya ichida:", x)


         funksiya()
```

```
x o`zgaruvchisi funksiya ichida: global
```

Python dasturlash tilida Mahalliy (lakal) o‘zgaruvchilar

- Agar **lakal** o‘zgaruvchini funksiyani tashqarisida chaqirsak bizga xato xabar ko‘rsatadi

```
def funksiya():
    y = "lakal"


funksiya()
print(y)
```

```
---------------------------------------------------------------
--
NameError                                 Traceback (most re
t)
~\AppData\Local\Temp/ipykernel_1728/55514807.py in <module>
      4
      5 funksiya()
----> 6 print(y)
```

Python dasturlash tilida Mahalliy (lakal) o‘zgaruvchilar

- Agar **lakal** o‘zgaruvchini funksiyani tashqarisida chaqirsak bizga xato xabar ko‘rsatadi

```
def funksiya():
    y = "lakal"


funksiya()
print(y)
```

```
---------------------------------------------------------------
--
NameError                                 Traceback (most re
t)
~\AppData\Local\Temp/ipykernel_1728/55514807.py in <module>
      4
      5 funksiya()
----> 6 print(y)
```

- Yuqoridagi suratda yozilgan python kodini natijasi xato bo‘lishining sababi biz global miqyosda lakal ‘**y**’ o‘zgaruvchisiga kirishga harakat qilmoqdamiz, **lakal** o‘zgaruvchi esa faqat **funksiya()** yoki lakal doirada ishlaydi.

Python dasturlash tilida berilganlar turi

Berilganlar turi - bu ma’lumotlarning bir qismi bilan bog‘liq bo‘lgan **atribut** bo‘lib, u kompyuter tizimiga uning qiymatini qanday talqin qilishni belgilaydi. Berilganlar turlarini tushunish ma’lumotlarning afzal ko‘rilgan formatda to‘planishini va har bir xususiyatning qiymati kutilganidek bo‘lishini ta’minlaydi. Masalan o‘zgaruvchini **matn** yoki **raqam** qiymatga ega ekanligini bilish uchun berilganlar turlari ishlatiladi. O‘zgaruvchilar har xil turdagi berilganlarni o‘zida saqlashi mumkin.

Python dasturlash tilida quyidagi berilganlar turlari mavjud:

Berilganlar turi	Pythondagi kalit so‘zi
Matn berilganlar turi:	str
Raqamli berilganlar turlari:	int, float, complex
Ketma-ketlik berilganlar turlari:	list, tuple
Xaritalash(mapping) berilganlar turi:	dict
To‘plam berilganlar turlari:	set, frozenset
Boolean berilganlar turi:	bool
Binary berilganlar turi:	bytes, bytearray, memoryview

Python dasturlash tilida quyidagi berilganlar turlari

- Python dasturlash tilida o‘zgaruvchi yaratayotgan vaqtda uning berilganlar turini yozish shart emas. Odatda o‘zgaruvchini qiymatiga qarab uning berilganlar turini avtomatik tarzda aniqlab olinadi.
- *Misol: ism = ‘Nilufar’*
- Yuqorida keltirilgan o‘zgaruvchini berilganlar turi **String (str)** sababi **ism** o‘zgaruvchisining qiymati **“”** qo‘shtirnoq belgisi ichida joylashgan. O‘zgaruvchilarni berilganlar turini bilish uchun **type()** funksiyasidan foydalaniladi.

Python dasturlash tilida quyidagi berilganlar turlari

```
In [1]: ism = "Nilufar"
        print(type(ism))

        <class 'str'>
```

Python dasturlash tilida quyidagi berilganlar turlari

- Agar siz berilganlar turini belgilamoqchi
- bo‘lsangiz, quyidagi konstruktor
- funksiyalaridan foydalanishingiz mumkin:

x = str("Salom Python")	**Berilganlar turi:** str
x = int(20)	**Berilganlar turi:** int
x = float(20.5)	**Berilganlar turi:** float
x = complex(1j)	**Berilganlar turi:** complex
x = list(("olma", "behi", "gilos"))	**Berilganlar turi:** list
x = tuple(("olma", "behi", "gilos"))	**Berilganlar turi:** tuple
x = range(6)	**Berilganlar turi:** range
x = dict(ism="Ali", yoshi=30)	**Berilganlar turi:** dict
x = set(("olma", "behi", "gilos"))	**Berilganlar turi:** Set
x = frozenset(("olma", "behi", "gilos"))	**Berilganlar turi:** frozenset
x = bool(5)	**Berilganlar turi:** bool
x = bytes(5)	**Berilganlar turi:** bytes
x = bytearray(5)	**Berilganlar turi:** bytearray
x = memoryview(bytes(5))	**Berilganlar turi:** memoryview

Python dasturlash tilida quyidagi berilganlar turlari

```
In [1]: ism = "Nilufar"
        print(type(ism))

        <class 'str'>
```

Python dasturlash tilida raqamli berilganlar turi

Python dasturlash tilida raqamli berilganlar turlari

Python dasturlash tilida uchta raqamli berilganlar turi mavjud:

Int

Float

Complex

Python dasturlash tilida raqamli berilganlar turlari

- Raqamli berilganlar turdagi o‘zgaruvchilar ularga qiymat berganingizda yaratiladi:

```
In [3]: x = 7      # int
        y = 3.8    # float
        z = 1j     # complex
```

Python dasturlash tilida raqamli berilganlar turlari

- Python dasturlash tilida har qanday o‘zgaruvchini berilganlar turini tekshirish uchun type() funksiyasidan foydalanish mumkin:

```
In [4]: x = 7     # int
        y = 3.8   # float
        z = 1j    # complex
        print(type(x))
        print(type(y))
        print(type(z))

        <class 'int'>
        <class 'float'>
        <class 'complex'>
```

Python dasturlash tilida raqamli berilganlar turlari

- **int (butun son) berilganlar turi**
- int - bu musbat yoki manfiy butun son hisoblanadi. Misol uchun butun sonlar lingivistikada matndagi so‘zlarni nechtaligini hisoblaganda ishlatilishi mumkin.

```
In [5]: x = 1
        y = 35656222554887711
        z = -3255522

        print(type(x))
        print(type(y))
        print(type(z))

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

Python dasturlash tilida raqamli berilganlar turlari

- **Float**
- Float qoldiqli raqam turi hisoblanadi. Bunday raqamlar manfiy yoki musbat bo‘lishi mumkin.

```
In [6]: x = 1.10
        y = 1.0
        z = -35.59

        print(type(x))
        print(type(y))
        print(type(z))
```

```
<class 'float'>
<class 'float'>
<class 'float'>
```

Python dasturlash tilida raqamli berilganlar turlari

- Float berilganlar turi raqami ilmiy raqamlar ham bo‘lishi mumkin.

```
In [7]: x = 35e3
        y = 12E4
        z = -87.7e100

        print(type(x))
        print(type(y))
        print(type(z))

        <class 'float'>
        <class 'float'>
        <class 'float'>
```

Python dasturlash tilida raqamli berilganlar turlari

- **Complex**
- Complex sonlar xayoliy qism sifatida "j" bilan yoziladi:

```
In [8]: x = 3+5j
        y = 5j
        z = -5j

        print(type(x))
        print(type(y))
        print(type(z))

        <class 'complex'>
        <class 'complex'>
        <class 'complex'>
```

Python dasturlash tilida O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

Siz **int()**, **float()** va **complex()** funksiyalari yordamida o‘zgaruvchini bir berilganlar turidan boshqa berilganlar turiga o‘zgartirishingiz mumkin:

Python dasturlash tilida O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

In [9]:
```
x = 1    # int
y = 2.8  # float
z = 1j   # complex

#int dan floatga o`zgartirish:
a = float(x)

#float dan intaga o`zgartirish:
b = int(y)

#int dan complexga o`zgartirish:
c = complex(x)

print(a)
print(b)
print(c)

print(type(a))
print(type(b))
print(type(c))
```

```
1.0
2
(1+0j)
<class 'float'>
<class 'int'>
<class 'complex'>
```

Python dasturlash tilida O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

- **int()** berilganlar turiga o‘zgartirish misollari:

```
In [10]: x = int(1)    # x 1ga teng
         y = int(2.8)  # y 2ga teng
         z = int("3")  # z 3 raqam
```

Python dasturlash tilida O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

- **float()** berilganlar turiga o‘zgartirish:

```
In [11]: x = float(1)      # x 1.0ga teng
         y = float(2.8)    # y 2.8ga teng
         z = float("3")    # z 3.0ga teng
         w = float("4.2")  # w 4.2ga teng
```

Python dasturlash tilida O‘zgaruvchi berilganlar turini o‘zgartirish (Type Conversion)

- **str()** berilganlar turiga o‘zgartirish:

```
In [12]: x = str("s1") # x 's1'ga teng
         y = str(2)    # y '2'ga teng
         z = str(3.0)  # z '3.0'ga teng
```

- ***Eslatma**: complex raqamlarini boshqa raqam turiga o‘zgartirish mumkin emas.*

Python dasturlash tilida Boolean (mantiqiy) berilganlar turi

Boolean so‘zini o‘zbek tiliga **mantiqiy** deb tarjima qilsak bo‘ladi. Mantiqiy degani dasturlashda ikkita natijadan birini chiqaradi deb ta’riflanadi. Malasan, to‘g‘ri yoki noto‘g‘ri, o‘chiq yoki yoniq, bor yoki yo‘q. Dasturlashda **Boolean** aslida **to‘g‘ri(True)** yoki **no‘to‘gri(False)** natijalarini beradi. Bu to‘g‘ri yoki noto‘g‘ri natijalar ba’zi kitoblarda 1 (To‘g‘ri) va 0 (noto‘g‘ri) bilan ham ifodalanadi. Odatda Boolean berilganlar turi solishtirish uchun ishlatiladi

Python dasturlash tilida Boolean (mantiqiy) berilganlar turi

- Quyida **Boolean** berilganlar turiga oid masalalar berilgan.

```
In [1]: print(10 > 9)
        print(10 == 9)
        print(10 < 9)

        True
        False
        False
```

Python dasturlash tilida Boolean (mantiqiy) berilganlar turi

- Boolean berilganlar turi agar (**if**) holatida ham ishlatiladi. *(Holatlar bo‘yicha keyingi mavzularda chuqurroq o‘rganamiz.)*

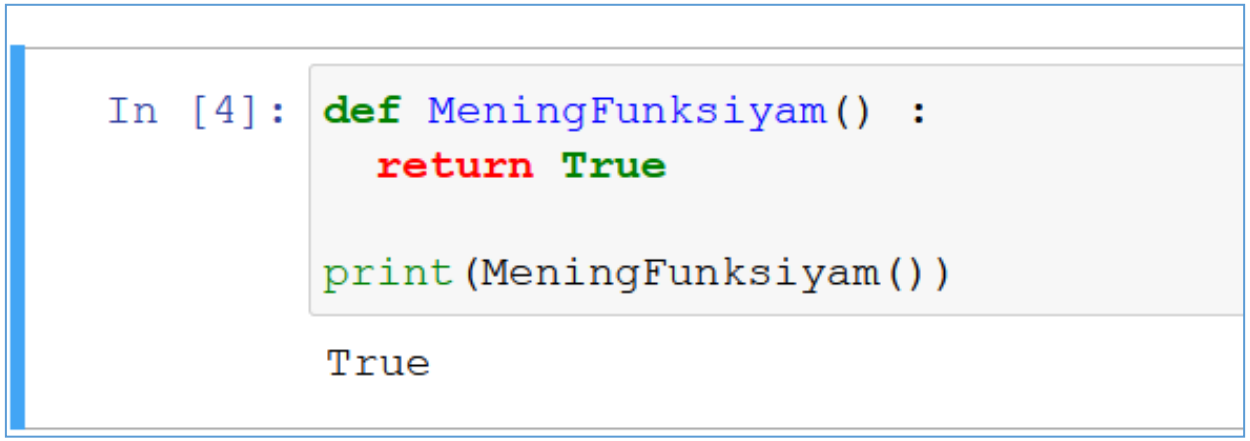
```
In [4]: satr = "Matn kattaligi"
        matn = "Python Dasturlash tili"

        if satr == matn:
            print("O`zgaruvchilar qiymati bir xil")
        else:
            print("O`zgaruvchilar qiymati har xil")

        O`zgaruvchilar qiymati har xil
```

Python dasturlash tilida Boolean (mantiqiy) berilganlar turi

• Boolean berilganlar turini funksiya yaratganda ham ishlatishingiz mumkin. Bunday funksiyalar javob tariqasida **True** yoki **Flase** javoblarini qaytarishi mumkin.



Foydalanilgan adabiyotlar

1. Mastering Object-Oriented Python: Build powerful applications with reusable code using OOP design patterns and Python 3.7, 2nd Edition, Steven F. Lott, Packt Publishing (June 14, 2019)

2. Learning Python, 5th Edition Fifth Edition, Mark Lutz , O'Reilly Media, June 12, 2013

3. Python Programming for Beginners: The Ultimate Guide for Beginners to Learn Python Programming: Crash Course on Python Programming for Beginners, AMZ Publishing, Independently published (July 13, 2021)

4. https://www.python.org/

5. https://www.w3schools.com/

6. https://www.codecademy.com/catalog/language/python

7. https://realpython.com/

8. https://www.anaconda.com/

E’tiboringiz uchun rahmat!