Python for Data Science

Лекція 3. Управляючі конструкції

Присвоєння

- **Присвоєння** — це процес зв’язування імен та значень
- Оператором присвоєння є символ **=**

```
1  name = "John"
2  number = 1
3  dictionary = {}
4
```

Порядок присвоєння

- При присвоєнні спочатку виконується права частина виразу

  Тобто вираз `a = 2 + 2` присвоїть `a` значення `4`

- Також можливе переприсвоєння, таке як інкрементація або декрементація (збільшення та зменшення значення)

  ```
  a += 1   # 5
  ```

  або

  ```
  a -= 1   # 3
  ```

Присвоєння декількох імен

- Можна робити декілька присвоєнь одночасно, якщо кількість імен зліва дорівнює кількості значень справа:

```
a, b = 1, 2
```

- Таким чином можна поміняти дві змінні місцями:

```
a, b = b, a
```

- Такий запис може мати і вкладену структуру:

```
(a, b), c = (1, 2), 3
```

Моржовий (walrus) оператор присвоєння

- Починаючи з версії Python 3.8 можлива наступна конструкція присвоєння:

```
print(x := 24)
```

```
>> 24
```

Використовується для динамічного присвоєння значень змінним у функціях і виразах:

```
value = input('Please enter something: ')

while value != '':
    print('Nice!')
    value = input('Please enter something: ')
```

```
while (value := input('Please enter something: ') != ''):
    print('Nice!')
```

Оператори порівняння

- == — дорівнює
- != — не дорівнює
- > — більше
- < — менше
- >= — більше або дорівнює
- <= — менше або дорівнює

Логічні оператори

- and — логічне “і”
- or — логічне “або”
- not — логічне “не” (заперечення)

Умовні вирази: if

- Умовні вирази починаються з ключового слова if та мають такий запис:

```
if a > b:
    print(a)
```

- Наведений вище код спрацює лише у тому випадку, коли вираз a > b повертає True

Умовні вирази: elif

- Умов може бути більше однієї, тоді використовується ключове слово `elif`:

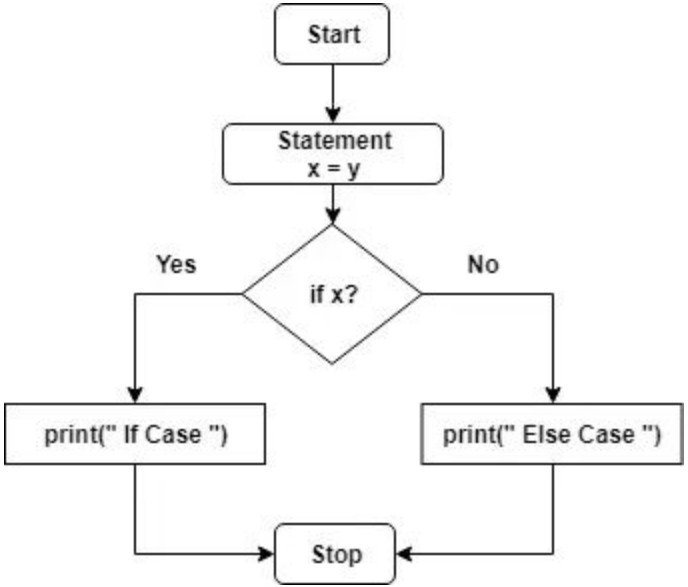
```
if a > b:
    print(a)
elif a < b:
    print(b)
```

Умовні вирази: else

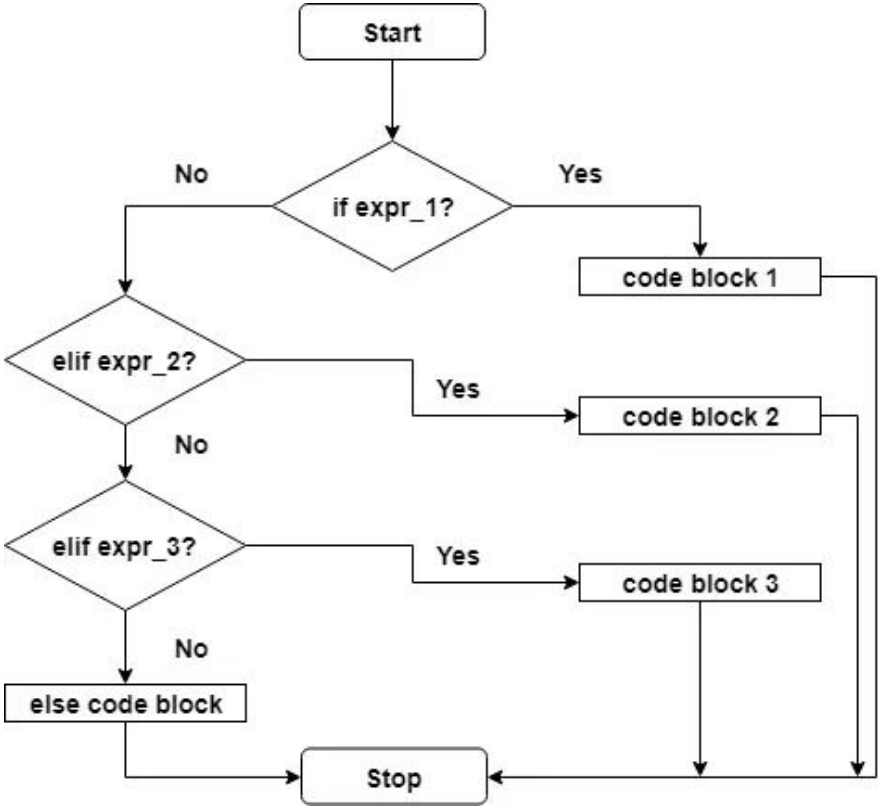
- Також можна додати блок `else`, який буде виконано, якщо жодна з умова не була `True`:

```
if a > b:
    print(a)
elif a < b:
    print(b)
else:
    print("==")
```

Візуалізація if else



Візуалізація if elif else



Вирахування умовних виразів

- Результат умовного виразу завжди повертає тип даних `bool`. Тобто або `True`, або `False`

- False повертають пусті контейнери, число `0`, та `None`

Тернарний оператор

- Скорочена форма написання if-else:

```
a = 54

b = 100 if a < 100 else 50
```

Цикл For

- Цикли надають можливість повторювати у коді певні дії
- `for` — найбільш вживаний цикл у Python
- `for` працює з рядками, послідовностями, генераторами та масивами

Синтаксис циклу For

- Оголошення циклу `for` має такий вигляд:

```
for i in sequence:
    …
```

- `i` — це змінна, за допомогою якою ми звертаємось до кожного наступного об’єкта у колекції `sequence`

- Кожен такий крок циклу називається ітерацією

```
for i in range(5):
    print(i)
```

List Comprehension

Це інструмент Python, який дозволяє утворити новий список на базі вже існуючого за допомогою спрощеного запису. Наприклад ми маємо список:

```
my_list = [1, 2, 8, 3, 4, 7, 12]
```

Код, що дозволить швидко отримати новий список, який містить тільки парні числа з `my list`:

```
new_list = [i for i in my_list if i % 2 == 0]
```

Цикл While

- Універсальний цикл Python — це `while`
- Синтаксис виглядає так:

```
while condition:
    …
```

- Цикл while працюватиме до тих пір, доки вказана умова дорівнюватиме `True`. Тобто можно зробити нескінченний цикл:

```
while True:
    …
```

Контроль циклів

Ці ключові слова дозволяють контролювати виконання циклів:

- `break` — повністю перериває виконання циклу
- `continue` — перериває поточну ітерацію та переходить до наступної
- Конструкція `else` (за аналогією з `if else`) — виконується, якщо цикл не було перервано

Задача FizzBuzz

- Виведіть на екран числа від 1 до 100.
- Замість чисел, які кратні 3, виведіть слово Fizz
- Замість чисел, які кратні 5, виведіть слово Buzz
- Замість чисел, які кратні 15, виведіть слово FizzBuzz
- Рішення має бути якомога простим та ефективним

Задачі

- Створіть список з рядків. За допомогою List Comprehension отримайте з нього новий список рядків, які не довші за 5 символів та починаються з голосних літер.

- Створіть цикл, який роздруковує у консолі всі високосні роки з початку минулого століття.