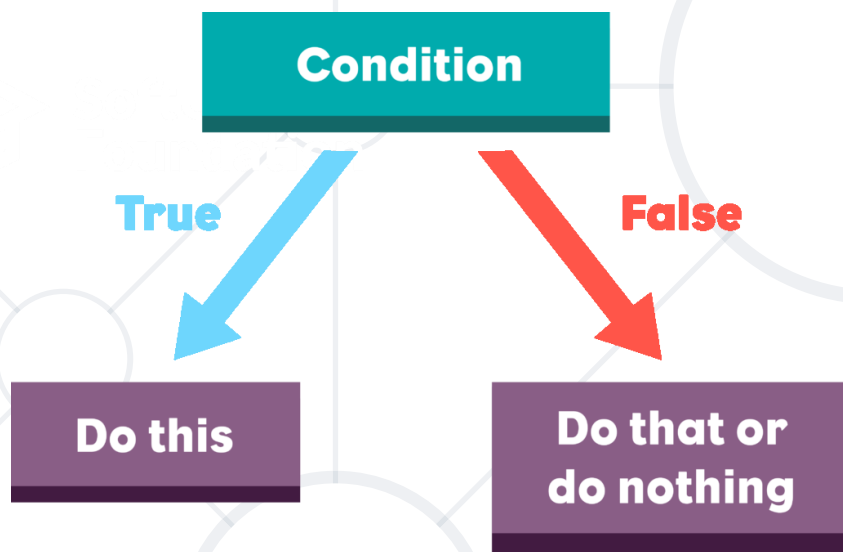


Условни конструкции

Логически изрази и проверки. Условна конструкция If-else



СофтУни

Преподавателски екип



SoftUni



Software University

<https://softuni.bg>

1. Преговор
2. Логически изрази и проверки
 - Оператори за сравнение
3. Условни конструкции
4. Закръгляне и форматиране
5. Дебъгване
6. Серия от проверки
7. Живот на променлива





Преговор

1. Какво ще се отпечата на конзолата, ако изпълним следната команда:

```
print("a" + "b")
```

ab

Error

ba

195

2. Какъв е типът на променливата:

```
number = "1000"
```

str

int

float

3. Как се нарича долепването на два текста (низа)?



Събиране

Конкатенация

Кулминация

Съединяване

4. Какво ще се отпечата на конзолата, ако изпълним следната команда:

```
print(10 % 3)
```

10

1

0

3

5. Каква стойност държи променливата **result**:

```
a = 5  
b = 2  
result = a / b
```

2

7

2.5

1

6. Какъв би бил резултатът, ако се опитаме да изпълним следната команда:

```
print(1 + 1 + "4" + 2 + 1)
```

Error

9

243

2421



Логически изрази и проверки

Оператори за сравнение

Оператори за сравнение



Оператор	Означение	Работи за
Равенство	==	числа, дати, други сравними типове
Различно	!=	
По-голямо	>	
По-голямо или равно	>=	
По-малко	<	
По-малко или равно	<=	

- В програмирането можем да сравняваме стойности
 - Резултатът от логическите изрази е **True** или **False**

```
a = 5
b = 10
print(a < b)           # True
print(a > 0)           # True
print(a > 100)         # False
print(a < a)           # False
print(a <= 5)          # True
print(b == 2 * a)      # True
```

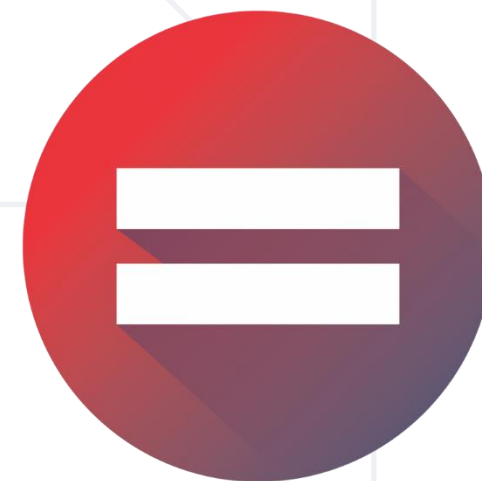


- Сравняване на текст чрез оператор за равенство (**==**)

```
a = 'Example'  
b = a  
print(a == b)      # True
```

```
a = input()  
b = input()  
print(a == b)      # True
```

Въвеждане на
еднаква стойност



- Има само следните две стойности **True** (вярно) или **False** (грешно)

```
is_valid = True
```

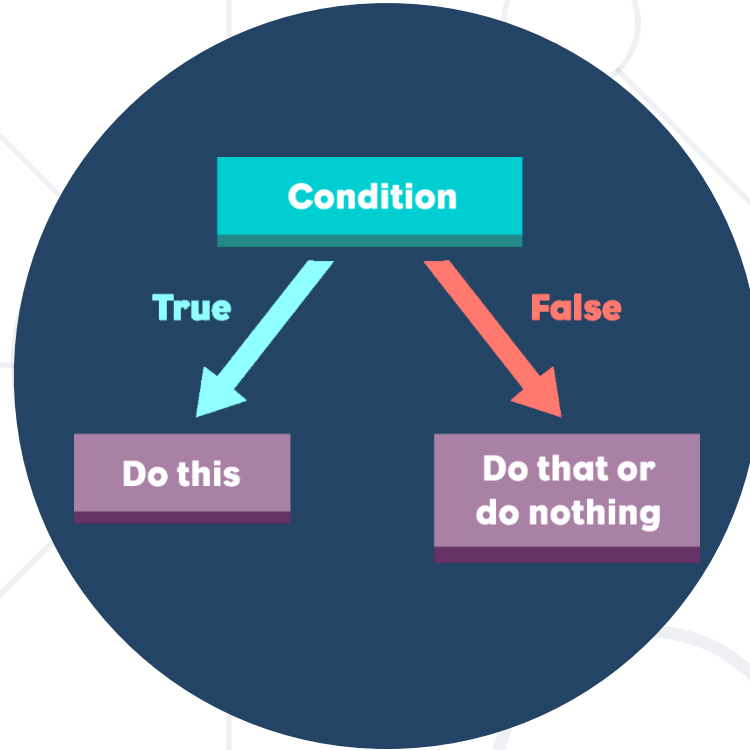
- Може да се създаде и с условие, което се свежда до True или False

```
is_positive = a > 0
```

Булева променлива - Пример

```
a = 5  
is_positive = a > 0  
print(is_positive) # True
```

```
a = -5  
is_positive = a > 0  
print(is_positive) # False
```




Условни конструкции

Прости проверки

Прости проверки

- Често проверяваме условия и извършваме действия според резултата



Условие
(булев израз)

```
if ...:  
    # код за изпълнение
```

Код за изпълнение при
вярност на условието

- Резултатът е **True** или **False**

- Напишете **програма**, която:
 - **Чете** оценка (**число**), въведена от потребителя
 - **Проверява** дали е отлична
 - **Отпечатва на конзолата** "Excellent!", ако оценката е по-голяма или равна на 5.50
- Пример:



Read input

grade \geq 5.50

false

No output

true

Print output



Прости проверки – if-else

- При **невярност** (**false**) на условието, можем да изпълним други действия – чрез **else** конструкция



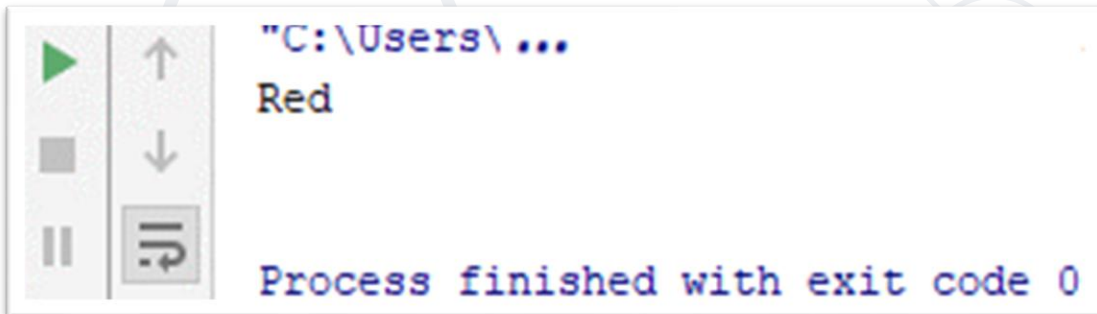
```
if ...:
    # код за изпълнение
else:
    # код за изпълнение
```

Код за изпълнение при
невярност на условието

- **Табулациите** въвеждат **блок от код** (група команди)
 - Изпълнява се редът, който **отговаря** на условието

```
color = 'red'  
if color == 'red':  
    print('Red')  
else:  
    print('Yellow')  
print('bye')
```

Извежда се "Red"



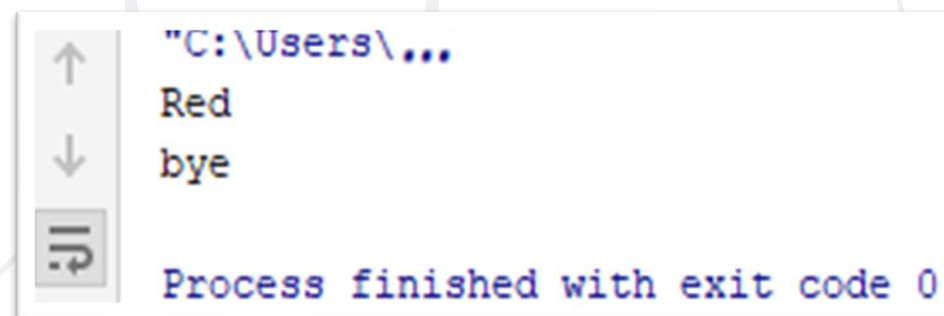
```
"C:\Users\...  
Red  
  
Process finished with exit code 0
```

- Без табулации ще се изпълнява и **последният** ред

```
color = 'red'  
if color == 'red':  
    print('Red')  
else:  
    print('Yellow')  
print('bye')
```

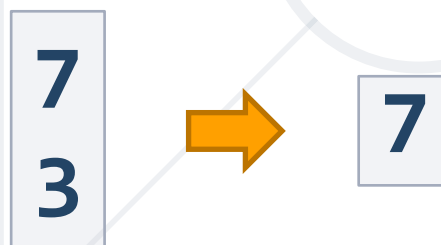
Изпълняват се
редовете отговарящи
на условието

Изпълнява се винаги – не е
част от if/else конструкцията



```
"C:\Users\...  
Red  
bye  
  
Process finished with exit code 0
```

- Напишете програма, която:
 - Чете две **числа**
 - Отпечатва на конзолата **по-голямото** от тях
- Пример:



Read input

num1 > num2

Print output

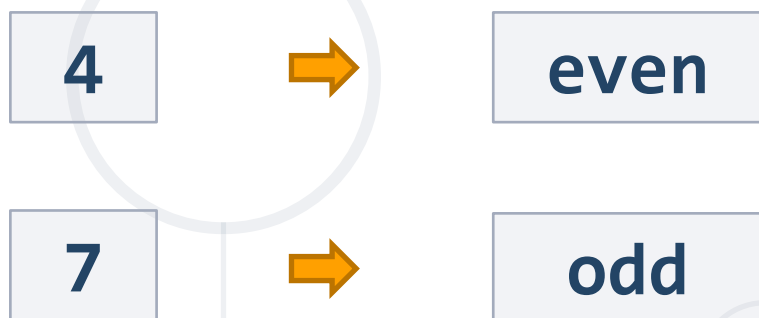
false

true

Print output


Четно или нечетно число – условие

- Напишете програма, която:
 - Проверява дали едно число е **четно** или **нечетно**
 - Ако е четно отпечатва на конзолата **"even"**
 - Ако е нечетно отпечатва на конзолата **"odd"**
- Пример:



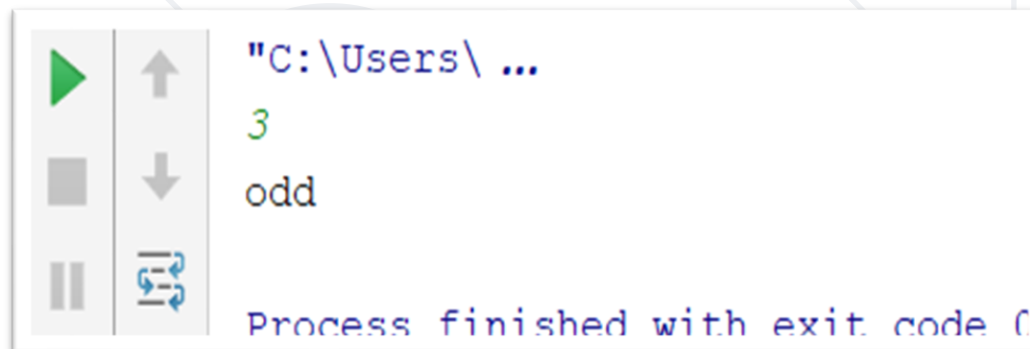
Четно или нечетно – решение

```
num = int(input())  
if num % 2 == 0:  
    print('even')  
else:  
    print('odd')
```



Terminal window showing execution for even number 4:

```
"C:\Users\ ...  
4  
even  
Process finished with exit code 0
```



Terminal window showing execution for odd number 3:

```
"C:\Users\ ...  
3  
odd  
Process finished with exit code 0
```



Закръгляне и Форматиране

- В програмирането можем да закръгляме дробни числа

- Закръгляне до следващо (по-голямо) цяло число:

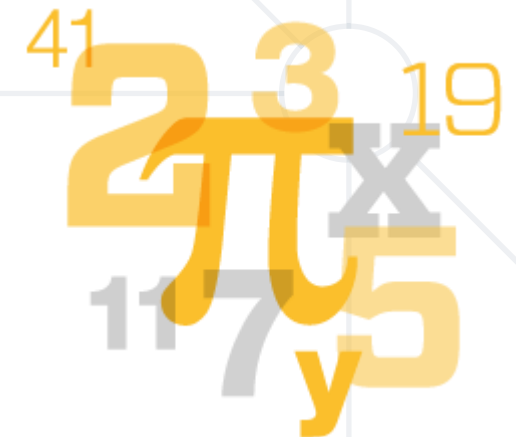
```
up = math.ceil(23.45)      # 24
```

- Закръгляне до предишно (по-малко) цяло число:

```
down = math.floor(45.67)   # 45
```

- Намиране на абсолютна стойност

```
example1 = abs(-50)        # 50  
example2 = abs(50)         # 50
```



- Закръгляне до 2 знака след десетичния знак:

```
rounded = round(45.67852, 2) # 45.68
```

- Форматиране до 2 знака след десетичния знак:

```
print(f"{123.456:.2f}") # 123.46
```

Брой символи след
десетичния знак

- Разлика между форматиране и закръгляне:

```
print(round(45.60000, 4)) # 45.6
```

```
print(f"{45.60000:.4f}") # 45.6000
```



Дебъгване

Прости операции с дебъгер

Дебъгване

- Процес на проследяване на изпълнението на програмата
- Това ни позволява да откриваме грешки (бъгове)



Breakpoint

```
1 a = int(input()) a: 5
2 b = a b: 5
3 c = 2 * a c: 10
4
5 if a < 10:
6     c = 10 * a
7 else:
8     b = 2 * a
9
10
```

if a < 10

Variables

Special Variables

- a = (int) 5
- b = (int) 5
- c = (int) 10

Дебъгване във PyCharm

- Натискане на **[Shift + F9]** ще стартира програмата в debug режим
- Можем да преминем към следващата стъпка с **[F8]**
- Можем да създаваме **[Ctrl + F8]** стопери – breakpoints
 - До тях можем директно да стигнем използвайки **[F9]**



Breakpoint

Breakpoint

```
SimpleConditions.py x
1  a = int(input())  a: 5
2  b = a  b: 5
3  c = 2 * a  c: 10
4
5  if a < 10:
6      c = 10 * a
7  else:
8      b = 2 * a
9
10
```

if a < 10

Variables

Special Variables

- a = (int) 5
- b = (int) 5
- c = (int) 10



Серии от проверки

По-сложни условни конструкции

Серии от проверки

- Конструкцията `if/else` - `if/else...` е серия от проверки




```
if ....:
    # КОД
elif ....:
    # КОД
elif ....:
    # КОД
else:
    # КОД
```

TRUE OR FALSE?

- При истинност на едно условие, **не се продължава** към проверяване на следващите условия

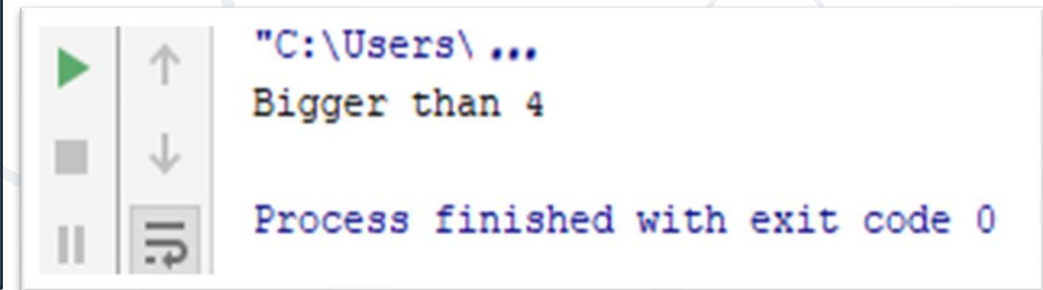
Серия от проверки – пример

- Програмата проверява първото условие, установява, че е вярно и приключва



```
a = 7
if a > 4:
    print('Bigger than 4')
elif a > 5:
    print('Bigger than 5')
else:
    print('Equal to 7')
```

Извежда се
само "Bigger
than 4"



```
"C:\Users\ ...
Bigger than 4

Process finished with exit code 0
```



Живот на променлива

Диапазон на използване

- Пример: Променливата `salary` ще съществува **само** ако е инициализирана някъде в програмата

```
current_day = "Monday"
if current_day == "Monday":
    salary = 1000
print(salary)  # 1000
```

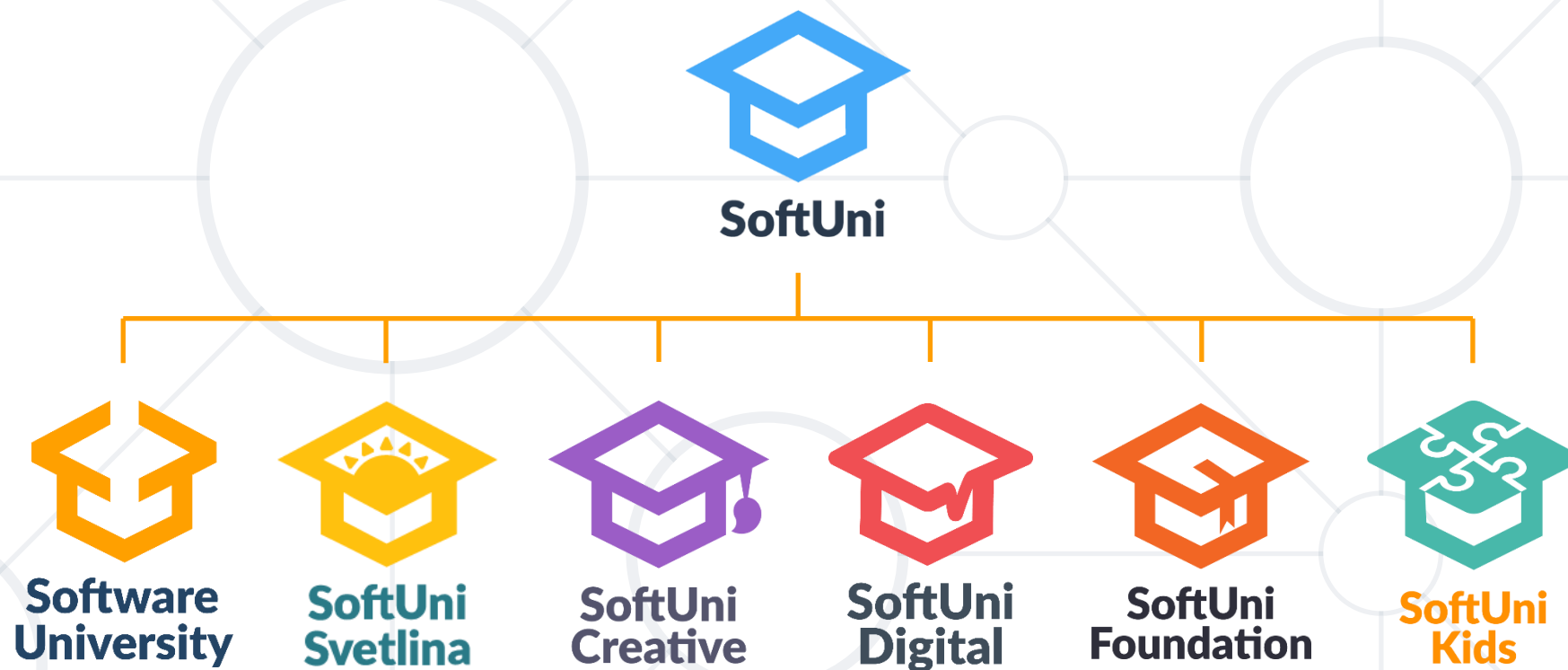
- Пример: Променливата **salary** **няма да** съществува, ако не бъде инициализирана някъде в програмата

```
current_day = "Tuesday"  
if current_day == "Monday":  
    salary = 1000  
print(salary)  # Error
```

- Оператори за сравнение
- Конструкции за проверка на условие – **if** и **if-else**
- Закръгляне и форматиране
- Серии от проверки
- Дебъгване
- Живот на променливата



Въпроси?



- Този курс (презентации, примери, демонстрационен код, упражнения, домашни, видео и други активи) представлява **защитено авторско съдържание**
- Нерегламентирано копиране, разпространение или използване е незаконно
- © СофтУни – <https://softuni.org>
- © Софтуерен университет – <https://softuni.bg>



- Софтуерен университет – качествено образование, професия и работа за софтуерни инженери
 - softuni.bg
- Фондация "Софтуерен университет"
 - softuni.foundation
- Софтуерен университет @ Facebook
 - facebook.com/SoftwareUniversity
- Дискуссионни форуми на СофтУни
 - forum.softuni.bg



Software University

