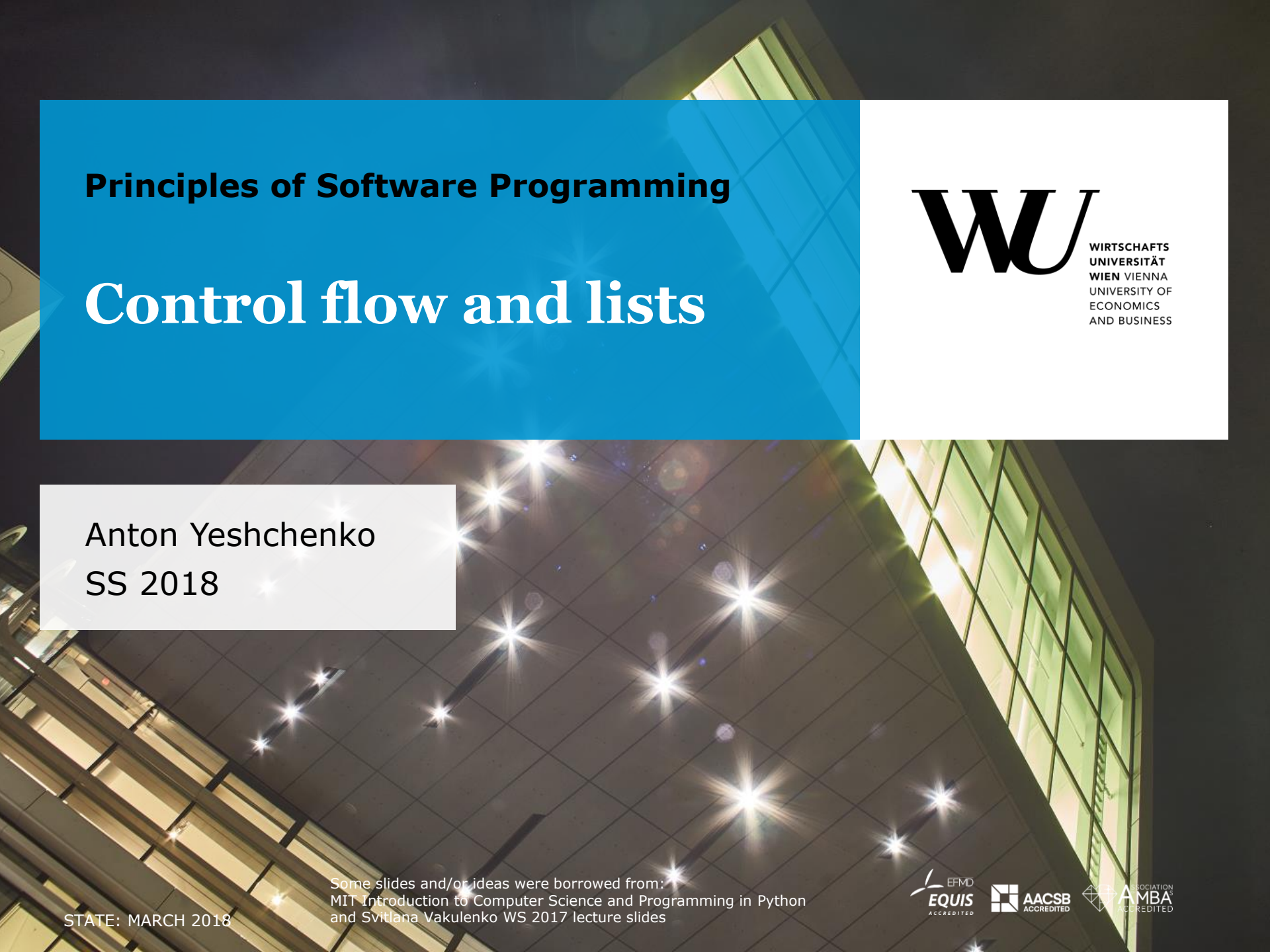


Principles of Software Programming

Control flow and lists

Anton Yeshchenko

SS 2018

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Some slides and/or ideas were borrowed from:
MIT Introduction to Computer Science and Programming in Python
and Svitlana Vakulenko WS 2017 lecture slides

STATE: MARCH 2018

EFMD EQUIS ACCREDITED · AACSB ACCREDITED · ASSOCIATION AMBA ACCREDITED

Quiz o!

- KAHOOT.it

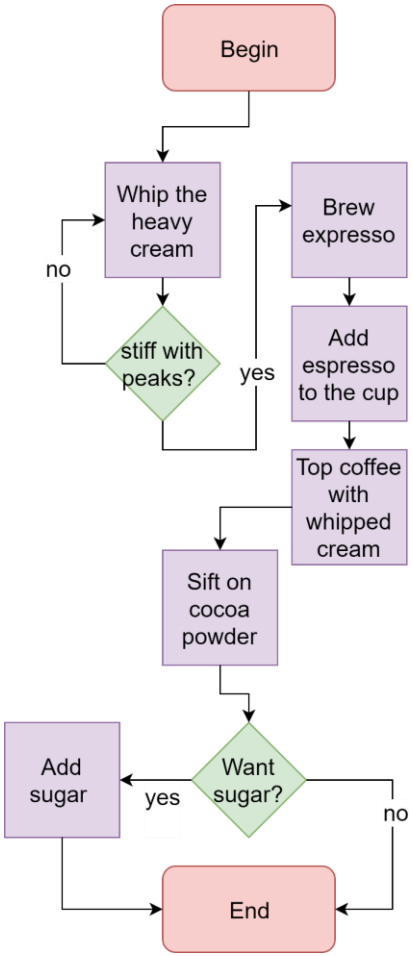
Recap

- Syntax, semantic
- Data types / Conversion
- Static / Dynamic typing
- Operators
- Variables
- Functions
- Packages

TODAY!

- **Control flow:**
  - if-else branches
  - loops
- **Lists:**
  - Arrays (lists)
  - create and fill Arrays
  - multidimensional Arrays

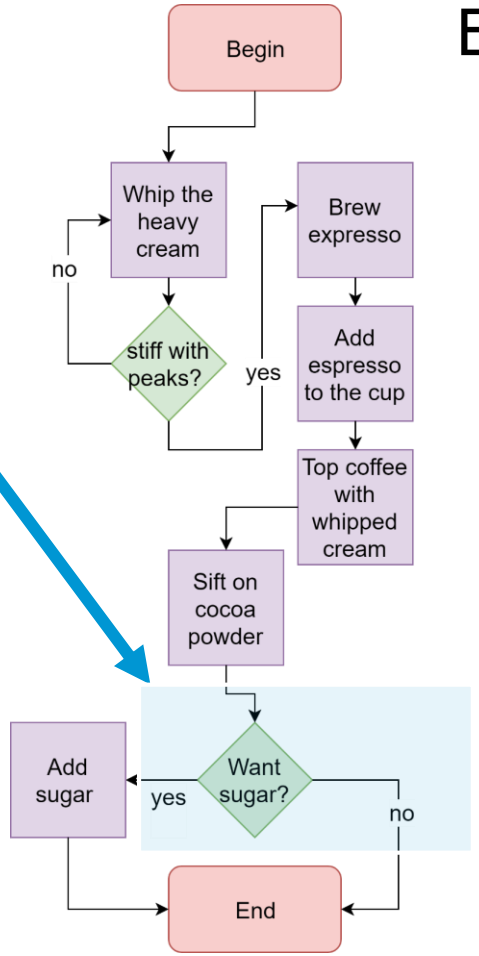
Control flow



EINSPANNER

Control flow

How do you represent choices in the program?



EINSPANNER

Choices in real live

- Deciding to go or not because of the price (float)
- Is it your friend? (string)
- Will there be ice cream? (bool)

Conditions!

```
choice = input("Do you want sugar? (yes/no)")

if choice == 'yes':
    print("get your sugar!")
elif choice == 'no':
    print("healthy choice! nice")
else:
    print("you are talking nonsense")
```

Condition operators!

Operator	What it means
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

EVERY COMPRARISON OPERATOR HAS RESULT!

Boolean operators!

Operator	What it means	What it looks like
and	True if both are true	`x and y`
or	True if at least one is true	`x or y`
not	True only if false	`not x`

TABLE of
How it works!

Conditions!

Comparison: >, <, ==, !=, <>, >=, <=
Arithmetic: +, -, *, /, %, **, //
Assignment: =, +=, -=,*=,/=, %=, **=, //=
Logical: and, or, not

EVERY OPERATOR HAS RESULT???

Combination of conditions

- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

Combination of conditions

- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost **>** 500:

Combination of conditions

- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost **>** 500:
  - buy_iphone_X()

Combination of conditions

- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost **>** 500:
  - buy_iphone_X()
- elif iphone_X_cost **>=** 500 **and** i_am_rich **==** True:

Combination of conditions

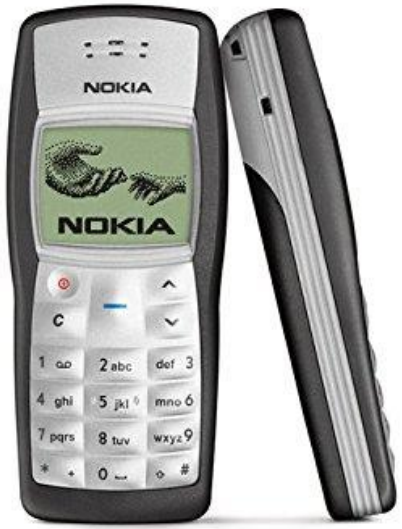
- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost **>** 500:
  - buy_iphone_X()
- elif iphone_X_cost **>=** 500 **and** i_am_rich **==** True:
  - buy_iphone_X()

Combination of conditions

- if Weather == “good” **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost **>** 500:
  - buy_iphone_X()
- elif iphone_X_cost **>=** 500 **and** i_am_rich **==** True:
  - buy_iphone_X()
- else:
  - buy_nokia_1100()



HTTPS://IMAGES-NA.SSL-IMAGES-AMAZON.COM/IMAGES/I/51CJ8F1AW6L._SY355_.JPG

Combination of conditions

- if Weather == "good" **and** i_have_free_time == True:
  - go_for_a_run()

- if **not** iphone_X_cost > 500:
  - buy_iphone_X()
- elif iphone_X_cost >= 500 **and** i_am_rich == True:
  - buy_iphone_X()
- else:
  - buy_nokia_1100()

- if I_want_to_talk == True **and** (girlfriend_insecure == False **or** I_am_stupid == True):
  - mention_that_she_ate_a_lot_after_romantic_dinner()

Combination of conditions

- if I_want_to_talk == True **and** (girlfriend_insecure == False **or** I_am_stupid == True):
  - mention_that_she_ate_a_lot_after_romantic_dinner()

Exercise 1

- ATM pin code checker

- Write a function
- Asks user to input the number
- **Check**
- If the number matches 1234 write “Login successful!”
- For any other number write “Login unsuccessful!”

- Hints:
  - use input(“Enter the pin”) function to ask user for the number
  - use if – else construct

https://pixabay.com/en/atm-pin-number-field-withdraw-cash-1524869/

THE PROBLEM ABOUT BEING A PROGRAMMER

My mom said:
"Honey, please go to the market and buy 1 bottle of milk. If they have eggs, bring 6"

I came back with 6 bottles of milk.

She said: "Why the hell did you buy 6 bottles of milk?"

I said: "BECAUSE THEY HAD EGGS!!!!"

I like a nice kürbiskernbrötchen!

HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!

```
def live_full_life():
    get_uP()
```

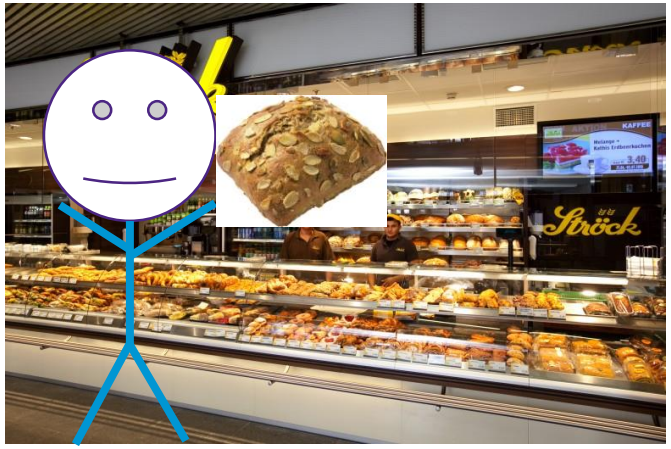

HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!

```
def live_full_life():
    get_uP()
    salivate()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

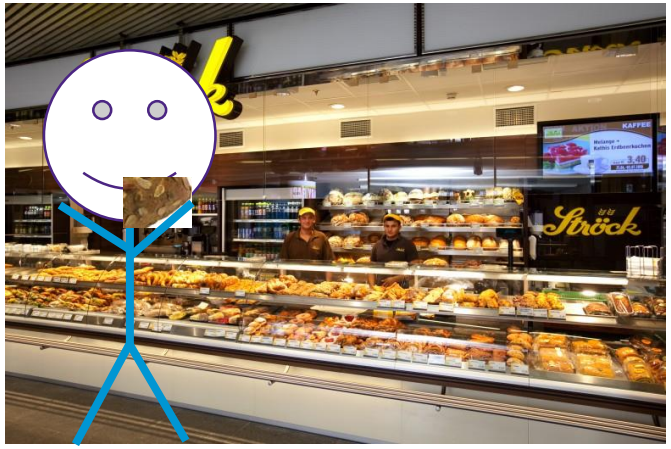
I like a nice kürbiskernbrötchen!




```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!




```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!

```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
live_full_life() #day 3
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
live_full_life() #day 3
live_full_life() #day 4
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
live_full_life() #day 3
live_full_life() #day 4
live_full_life() #day 5
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
live_full_life() #day 3
live_full_life() #day 4
live_full_life() #day 5
......
live_full_life() #day 234
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
live_full_life() #day 1
live_full_life() #day 2
live_full_life() #day 3
live_full_life() #day 4
live_full_life() #day 5
......
live_full_life() #day 234
live_full_life() #day 235
live_full_life() #day 236
live_full_life() #day 237
....
```

HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernbrötchen!


```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
while alive:
    live_full_life()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

I like a nice kürbiskernl

Now I can relax and just die one day. No need counting days!

```
def live_full_life():
    get_uP()
    salivate()
    go_to_backery()
    eat()
    go_to_sleep()
```

```
while alive:
    live_full_life()
```


HTTP://WWW.LOVETHISPIC.COM/UPLOADED_IMAGES/139011-COZY-CABIN-BED.JPG

Loops! While

```
while <condition>:
    <expression>
    <expression>
    ...
```

- `<condition>` evaluates to a Boolean
- if `<condition>` is `True`, do all the steps inside the while code block
- check `<condition>` again
- repeat until `<condition>` is `False`

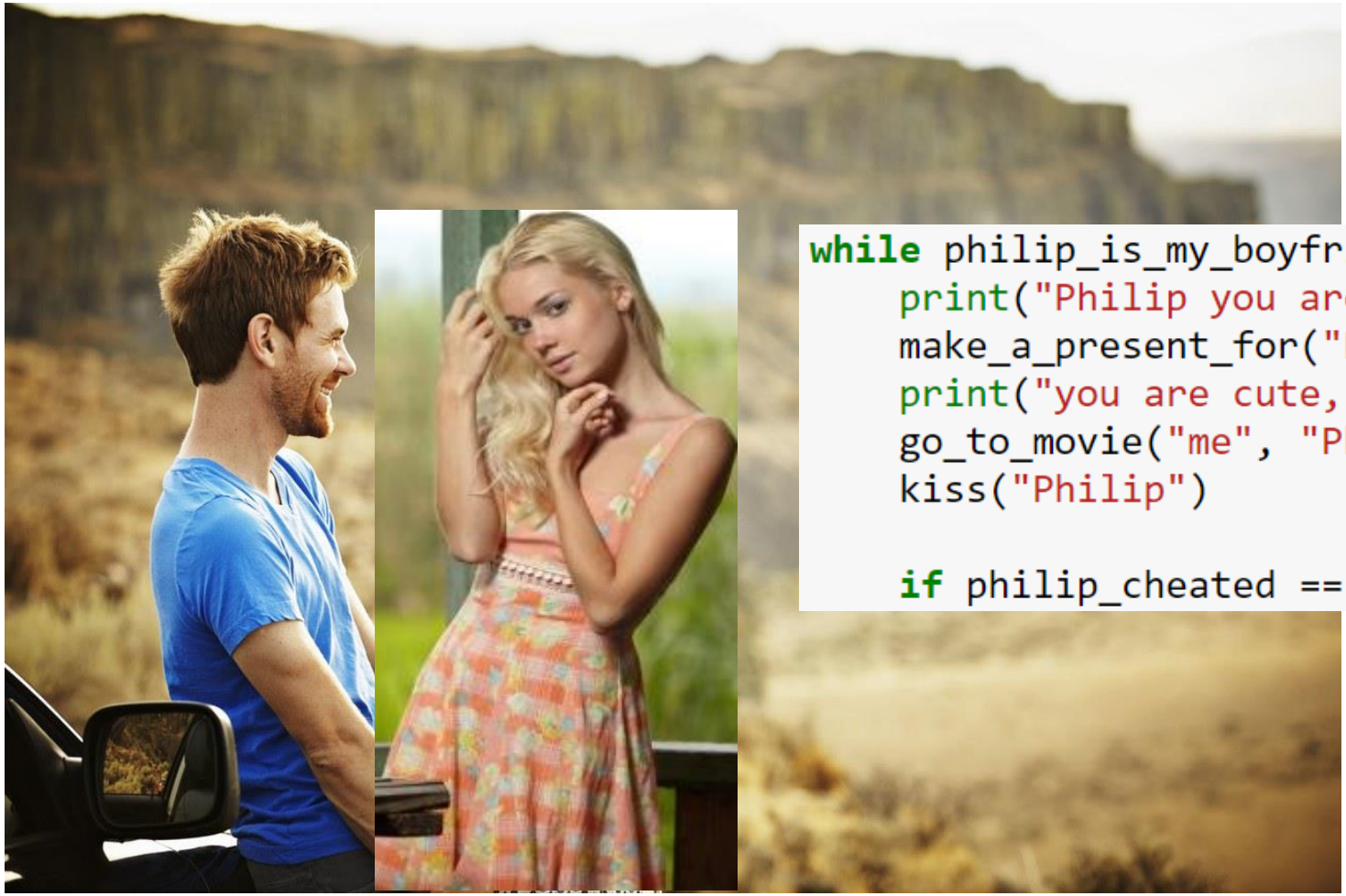
One more story of life!

Soooo nice couple

```
while philip_is_my_boyfriend == True:
    print("Philip you are a good guy!")
    make_a_present_for("Philip")
    print("you are cute, Philip!")
    go_to_movie("me", "Philip")
    kiss("Philip")
```


HTTPS://MEDIA.GLAMOUR.COM/PHOTOS/5696D70E01ED531C6F00B9AF/MASTER/W_1280,C_LIMIT/SEX-LOVE-LIFE-2015-07-TIMES-SAY-I-LOVE-YOU-MAIN.JPG

Not so nice boyfriend!

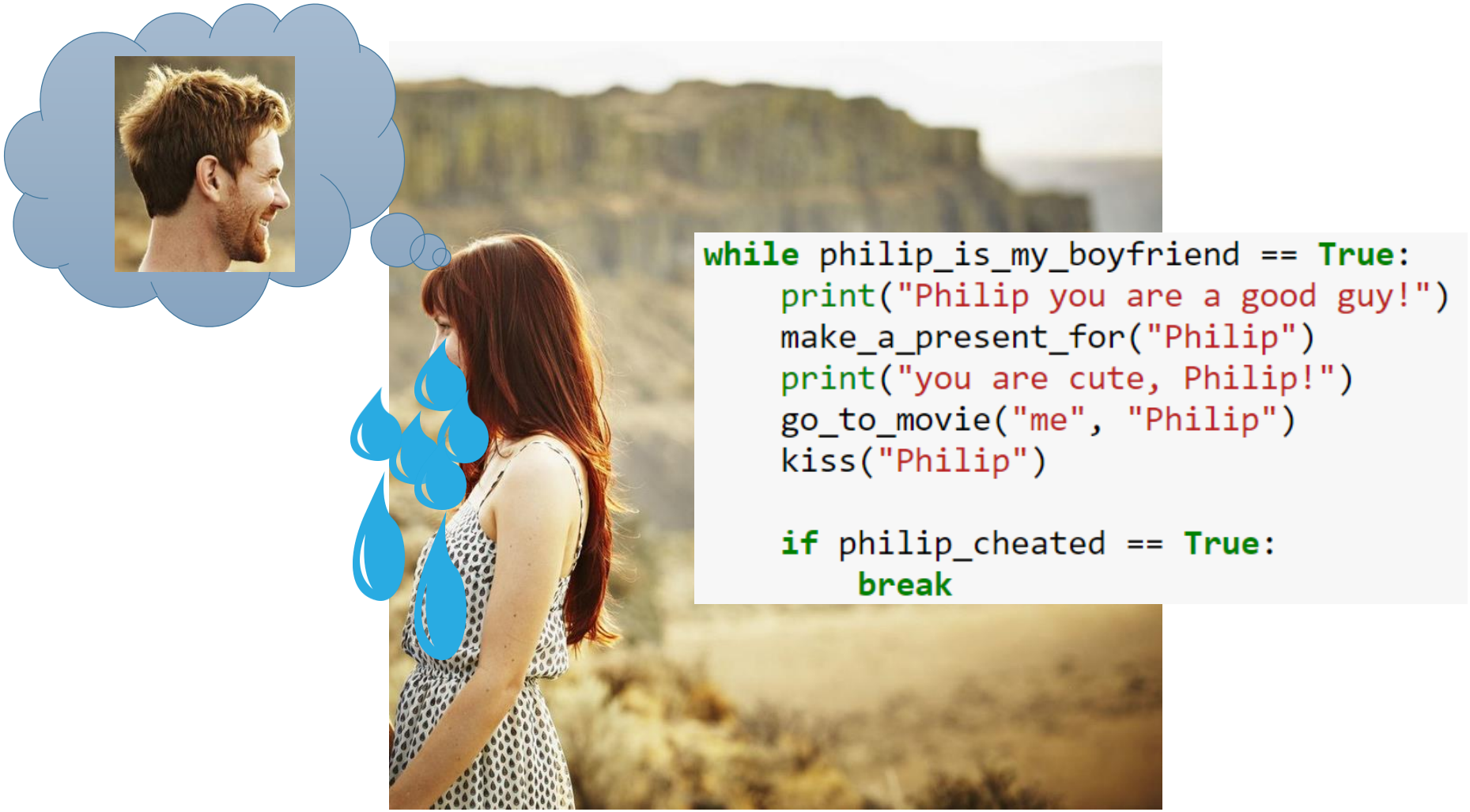


```
while philip_is_my_boyfriend == True:
    print("Philip you are a good guy!")
    make_a_present_for("Philip")
    print("you are cute, Philip!")
    go_to_movie("me", "Philip")
    kiss("Philip")

    if philip_cheated == True:
```

HTTPS://MEDIA.GLAMOUR.COM/PHOTOS/5696D70E01ED531C6F00B9AF/MASTER/W_1280,C_LIMIT/SEX-LOVE-LIFE-2015-07-TIMES-SAY-I-LOVE-YOU-MAIN.JPG

Dump him!



```
while philip_is_my_boyfriend == True:
    print("Philip you are a good guy!")
    make_a_present_for("Philip")
    print("you are cute, Philip!")
    go_to_movie("me", "Philip")
    kiss("Philip")

    if philip_cheated == True:
        break
```


HTTPS://MEDIA.GLAMOUR.COM/PHOTOS/5696D70E01ED531C6F00B9AF/MASTER/W_1280,C_LIMIT/SEX-LOVE-LIFE-2015-07-TIMES-SAY-I-LOVE-YOU-MAIN.JPG

Exercise 2

- ATM pin code checker

- Write a function
- Asks user to input the number
- **Check**
- If the number matches 1234 write “Login successful!”
- For any other number write “Login unsuccessful!”
- **If the user inputs wrong number, ask for input again until he does it right!**

- Use a loop!

https://pixabay.com/en/atm-pin-number-field-withdraw-cash-1524869/

Quiz 1!!!!!!

What if we have more than one thing.

Data Structure: List

http://www.bbc.co.uk/scotland/education/wwww/buildings/standard/shopping/?item=list

Data Structure: List


```
shopping_list = ['Milk', 'Apples',
'Eggs', 'Toilet rolls', 'Bananas',
'Bread']
```

http://www.bbc.co.uk/scotland/education/wwww/buildings/standard/shopping/?item=list

List Slicing


```
shopping_list = ['Milk', 'Apples',
'Eggs', 'Toilet rolls', 'Bananas',
'Bread']

shopping_list[1]
shopping_list[-1]
shopping_list[0:-1]
```

List Functions


```
shopping_list = ['Milk', 'Apples',
'Eggs', 'Toilet rolls', 'Bananas',
'Bread']

len(shopping_list)

'Milk' in shopping_list
```

List Insert


```
shopping_list = ['Milk', 'Apples',
'Eggs', 'Toilet rolls', 'Bananas',
'Bread']

len(shopping_list)

'Milk' in shopping_list
```

shopping_list.append('cucumber')

List Remove/Clear

shopping_list.remove(2)

Iterating the list!

(/ɪtəˈreɪʃ(ə)n/ the repetition of a process or utterance, also called successive approximation)

WU WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Why would we do such a thing?

1. With **while** loop
2. With **for** and in **range**

While and For loops

- iterate through numbers in a sequence

```
# more complicated with while loop
n = 0
while n < 5:
    print(n)
    n = n+1


# shortcut with for loop
for n in range(5):
    print(n)
```

For loops

```
for <variable> in range(<some_num>):
    <expression>
    <expression>
    ...
```

- each time through the loop, `<variable>` takes a value
- first time, `<variable>` starts at the smallest value
- next time, `<variable>` gets the prev value + 1
- etc.

range(start,stop,step)

- default values are `start = 0` and `step = 1` and optional
- loop until value is `stop - 1`

```
mysum = 0
for i in range(7, 10):
    mysum += i
print(mysum)
```

```
mysum = 0
for i in range(5, 11, 2):
    mysum += i
print(mysum)
```

For vs While

`for` loops

- **know** number of iterations
- can **end early** via `break`
- uses a **counter**
- **can rewrite** a `for` loop using a `while` loop

`while` loops

- **unbounded** number of iterations
- can **end early** via `break`
- can use a **counter but must initialize** before loop and increment it inside loop
- **may not be able to rewrite** a `while` loop using a `for` loop

Exercise 3. MATHEMATICS!

- Write a function that will calculate the average (mean) of a set of numbers.

1. Define function with a name mean(array_numbers)

2. Test function:
   1. Initialize array with some numbers
   2. array_num = [12312,4315,456346,4353,476,458346,7345732]
   3. Call function mean(array_num)

https://pixabay.com/en/atm-pin-number-field-withdraw-cash-1524869/

Exercise 4. Shopping done!

http://www.kegkits.com/kegerator2.htm

http://koshersamurai.wordpress.com/2012/10/17/unhealthy-interests-part-7-grocery-shopping/

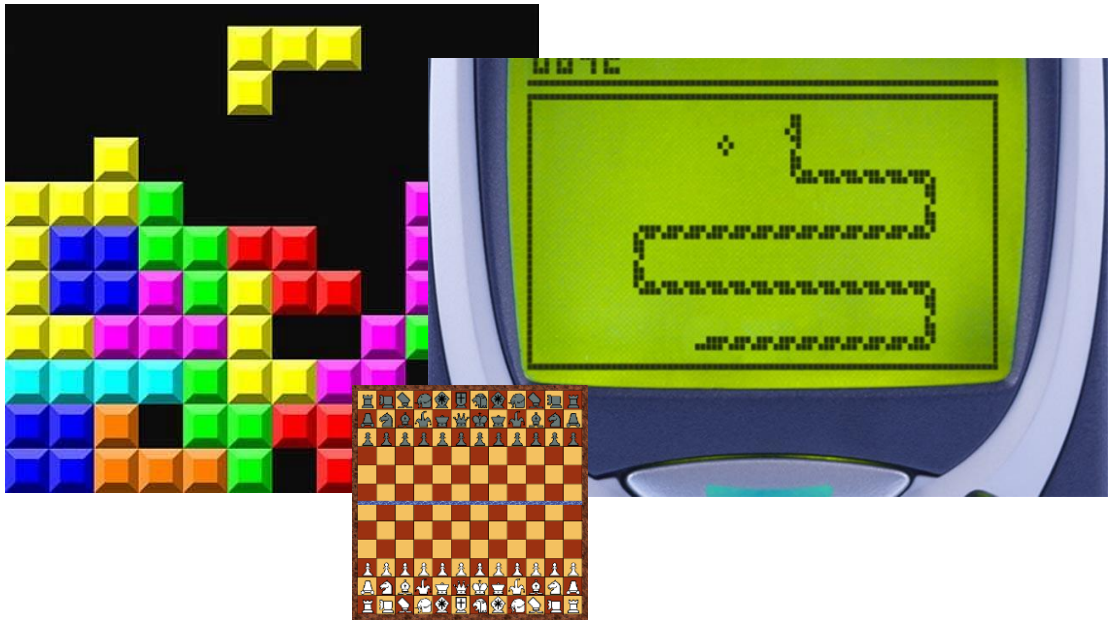
http://www.wheeleybaskets.co.uk/supermarket_basket.aspx?ref=41

Multidimentional array (list)

List that has lists.

1. Spreadsheets.
2. Games (field of the game: tetris, snake, chess, rock–paper–scissors)

Students	HW1	HW2	HW3
h0906415			
h1152610	5		
h1153937	5		
h11705542	5		
h11778023	3		
h1254044	5		
h1351555			
h1351736	5		
h1451363			
h1451536			
h1452220			



HTTP://CHESSVARIANTS.WDFILES.COM/LOCAL--FILES/GRAND-COURIER-EURASIAN-CHESS/GCE-START-ARRAY.PNG, HTTP://IMG.PHONANDROID.COM/2015/05/SNAKE-GRAND-RETOUR-SMARTPHONE.JPG, HTTPS://SECURE.I.TELEGRAPH.CO.UK/MULTIMEDIA/ARCHIVE/02933/TETRIS_2933521B.JPG

Quizzz #2!!!

- Kahoot!

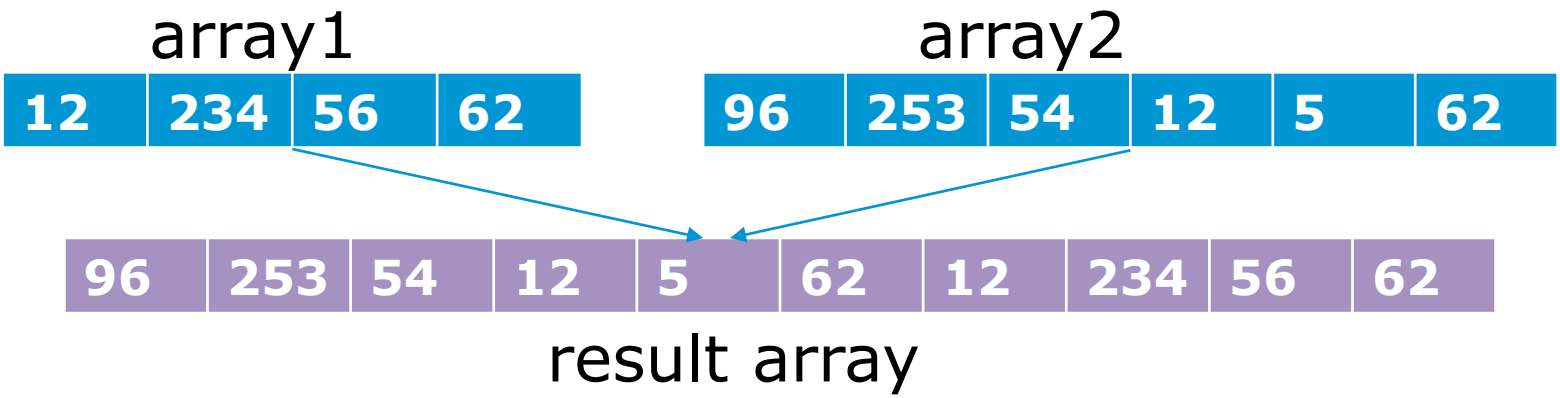
Recap today! It is getting

synonyms=['complicated', 'complex', 'difficult', 'perplexing', 'sophisticated', 'entangled', 'elaborate']

- If
- While
- For
- Break
- List
- list(list)

Homework 3!
Deadline tomorrow 21.03.2018 22:00

- 1. Write program that asks user 10 numbers, stores them into array, calculates mean (average) and prints on the screen.
- 2. Write program that has two arrays as an input, creates third array and adds all elements from first and second to the third. Print result on the screen



- 3. Write program that prints all array on the screen, leaving out numbers that are less than 50. (use **for** loop on array and **if** statement)

Next time!

- **Programming:** algorithms, syntax and semantics, programming, compiler, interpreter
- **Basics and types:** variables, operations, primitive data types, Strings, static vs dynamic typing, explicit vs implicit type casting
- **Control flow and functions:** if-else branches, loops, functions (parameters, return values)
- **Lists:** Arrays (lists), create and fill Arrays, multidimensional Arrays
- **Classes:** Class vs Instance of class, objects, create objects, instance variable, constructor, method overloading
- **Inheritance:** inherit classes, method overriding, problems and solutions for multiple inheritance
- **Information hiding:** variable access, access modifier (Java) and naming conventions, get- and set-methods
- **Object oriented programming:** why OOP, inheritance ("is-a"- and "is-part-of"-relations), information hiding/encapsulation, abstract classes, Super-constructor, polymorphism