Informatik 1 - Biomedical Engineering

Tutor Session 1 - Syntax/Variables and Datatypes

Overview

- Use the Newsgroup
- Syntax
- Variables and Datatypes
- Typecasts
- Operators
- String-operators
- If Then Else
- for-Schleife
- I/O
- while-Schleife
- Examples

Newsgroups

- Use the Newsgroup for Questions
- tu-graz.lv.info-bm
- You have to read it

Syntax

The syntax specifys some rules, you have to follow. Python works with indents. You have to use the same indent in your whole document.

- 2 Spaces
- 4 Spaces
- 1 Tab

wrong

In [42]:
```
def main():
a = 10
```

```
  File "<ipython-input-42-d0e93b0e0af1>", line 2
    a = 10
    ^
IndentationError: expected an indented block
```

right

In [ ]:
```
def main():
    a = 10
    print(a)
```

Variables and Datatypes

- Variables are Placeholder
- Datatypes specify for what a variable needs place
- In Python the datatype declaration is done automatically
- Three types of datatypes:
  - Numbers:
    - Integer [int]
    - Float [float]
    - Double [double]
  - Text:
    - Character [char]
    - String [str]
- Boolean:
  - True
  - False

In [ ]:
```
integer = 12 #int
string = "I am a String" #str
floating_number = 3.45 #float
character = '5' #char
```

In [ ]:
```
#now you guess
var0 = 123
var1 = "234"
var2 = 3.4546
var3 = '2'
```

In [ ]:
```
print(type(var0))
print(type(var1))
print(type(var2))
print(type(var3))
```

Typecasts

you can cast a variable, so it changes its type

In [58]:
```
string = '23'
print(string, type(string))
string = int(string)
print(string, type(string))
```

```
23 <class 'str'>
23 <class 'int'>
```

Operators

Operators have one or more arguments as input and produce a new value.

- assignment operator(=)
- mathematical operators
  - Addition(+)
  - Subtraction(-)
  - Multiplication(*)
  - Division(/)
  - Modulo(%)
- Comparison operators --> returns a boolean(true or false)
  - <,<=,>,>=,==,!=
- Bit operators
  - &,|,^
- Shift operators
  - <<,>>

In [ ]:
```
result = integer + string
print(result)
```

In [ ]:
```
result = integer
print(result,type(result))
```

In [ ]:
```
result += floating_number
print(result,type(result))
```

In [ ]:
```
result = integer + character
print(result,type(result))
```

In [ ]:
```
result = string + character
print(result,type(result))
```

In [ ]:
```
result = integer > floating_number
print(result,type(result))
```

In [ ]:
```
result = string < character
print(result,type(result))
```

String Operators

- len(string)
- string[4]
- string.isdigit()
- string.split()
- string.find()
- string.replace()
- string.format()
- More information: https://docs.python.org/3.1/library/stdtypes.html#string-methods

In [12]:
```
string = "I am a string"
print("length of string: ", len(string))
print("Char on specific position: ", string[4])
print("Part of string: ", string[3:7])
number_string = "23"
print("Is digit: ", number_string.isdigit(), type(number_string))
#split examples
print(string.split( ))
print(string.split('i',1))
print(string.split('a'))
#find
print(string.find("string"))#returns the index
#replace
print(string.replace("string","character list"))
#format
print("I can count from {} to the incredible number {}".format(1, 100))
```

```
length of string:  13
Char on specific position:
Part of string:  m a
Is digit:  True <class 'str'>
['I', 'am', 'a', 'string']
['I am a str', 'ng']
['I ', 'm ', ' string']
7
I am a character list
I can count from 1 to the incredible number 100
```

IF THEN ELSE

In [ ]:
```
condition = 42
if condition == 42:
    print ("the answer is 42")
else:
    print ("So long and thanks for all the fish")
```

In [ ]:
```
condition = 100
if condition == 42:
    print ("the answer is 42")
elif condition == 3.14159:
    print ("today is Pi day")
elif condition == 2:
    print ("2 is the only even prime number")
else:
    print ("var is not an interesting number")
```

for-Schleife

In [ ]:
```
for i in range(0,9)
    print(i)
```

I/O

Python build-in function open() opens a file and returns a File Object

- r - Reading a file
- w - Open for writing (truncating the file first)
- a - Open for writing (appending to the end if exists)
- x - Create a new file and open it for writing
- b - Binary mode (default is text mode)
   The build-in function input() lets the user interact with the program

In [ ]:
```
#reading whole file
tf = open('textfile.txt', 'r')
content = tf.read()
tf.close()
```

In [ ]:
```
#reading line by line
tf = open('textfile.txt', 'r')
for line in tf:
    print(line)
tf.close()
```

In [ ]:
```
#writing to a file
tf = open('textfile.txt', 'w')
for i in range(10):
    line = 'Line number ' + str(i+1) + '\n'
    tf.write(line)
tf.close()
```

In [4]:
```
#input function

var = input("Type in my value: ")
print("Variable: {}, my type is: {}".format(var,type(var)))
```

```
Type in my value: 34
Variable: 34, my type is: <class 'str'>
```

while-Schleife

In [ ]:
```
condition = 0
while condition <= 10:
    print(condition)
    condition = condition+1
```

Example Program

Try to add your Matrikelnumber to your name and print it out.

In [50]:
```
name = "benjamin"
m_number = "01430425"
result = name+m_number
print(result)
```

```
benjamin01430425
```

write a program that lets you guess an integer, the program should only stop, when the number is guessed

In [56]:
```
def guessNum(number):
    #print(number)
    condition = True
    while condition:
        guess =  int(input("Enter a number: "))
        if guess == number:
            print("Congratulations! You guessed it.")
            condition = False
        elif guess > number:
            print("Smaller Number")
        elif guess < number:
            print("Bigger Number")

def main():
    number = int(23)
    guessNum(number)

if __name__ == "__main__":
    main()
```

```
Enter a number: 23
Congratulations! You guessed it.
```