Session 3

Python Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

Creating a Function

In Python a function is defined using the def keyword:

```python
PYTHON

def my_function():
  print("Hello from a function")
```

Calling a Function

To call a function, use the function name followed by parenthesis:

```python
PYTHON

def my_function():
  print("Hello from a function")

my_function()
```

Arguments

Information can be passed into functions as arguments.

Arguments are specified after the function name, inside the parentheses. *You can add as many arguments as you want*, just separate them with a **comma**.

The following example has a function with one argument `fname`. When the function is called, we pass along a first name, which is used inside the function to print a hello message:

PYTHON

```python
def hi(fname):
  print('Hi this is ' + fname)

hi("Armin")
hi("Sara")
hi("Matin")
```

Number of Arguments

By default, a function must be called with the correct number of arguments. Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less.

PYTHON

```python
# This function expects 2 arguments, and gets 2 arguments:
def my_function(fname, lname):
  print(fname + " " + lname)

my_function("Zahra", "Khorshidi")
```

> **Note:** If you try to call the function with 1 or 3 arguments, you will get an error.

Arbitrary Arguments, *args

If you do not know how many arguments that will be passed into your function, add a `*` before the parameter name in the function definition.

This way the function will receive a *tuple* of arguments, and can access the items accordingly:

PYTHON

```python
# If the number of arguments is unknown, add a `*` before the
parameter name:
def my_function(*kids):
  print("The youngest child is " + kids[2])

my_function("Hamid", "Majid", "Nader")
```

Keyword Arguments

You can also send arguments with the *key* = *value* syntax.

This way the order of the arguments **does not** matter.

PYTHON

```python
def my_function(child3, child2, child1):
  print("The youngest child is " + child3)

my_function(child1 = "Hamid", child2 = "Majid", child3 = "Nader")
```

Arbitrary Keyword Arguments, **kwargs

If you do not know how many keyword arguments that will be passed into your function, add two asterisk: `**` before the parameter name in the function definition.

This way the function will receive a *dictionary* of arguments, and can access the items accordingly:

PYTHON

```python
def my_function(**kid):
  print("His last name is " + kid["lname"])

my_function(fname = "Hamid", lname = "Shaiegh")
```

Default Parameter Value

The following example shows how to use a default parameter value.

If we call the function without argument, it uses the default value:

PYTHON

```python
def my_function(country = "Norway"):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```

Return Values

To let a function return a value, use the `return` statement:

PYTHON

```python
def my_function(x):
  return 5 * x

y = my_function(3)
print(y)
```

The pass Statement

`function` definitions cannot be empty, but if you for some reason have a `function` definition with no content, put in the `pass` statement to avoid getting an error.

PYTHON

```python
def myfunction():
  pass
```

File Handling

File handling is an important part of any application.

Python has several functions for creating, reading, updating, and deleting files.

`"r"` - Read - Default value. Opens a file for reading, error if the file does not exist

`"a"` - Append - Opens a file for appending, creates the file if it does not exist

`"w"` - Write - Opens a file for writing, creates the file if it does not exist

`"x"` - Create - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode

`"t"` - Text - Default value. Text mode

`"b"` - Binary - Binary mode (e.g. images)

To open the file, use the built-in `open()` function.

The `open()` function returns a file object, which has a `read()` method for reading the content of the file:

PYTHON

```python
f = open("demofile.txt")
print(f.read())
```

PYTHON

```python
f = open("demofile2.txt", "a")
f.write("Now the file has more content!")
f.close()

# open and read the file after the appending:
f = open("demofile2.txt", "r")
print(f.read())
```

An Example

PYTHON

```python
def passvalidation(password):

    if len(password) < 6:

        print('Your password must be at least 6 characters.')

    elif password.isnumeric():

        print('Your password must at least have 1 letter.')



    elif password.isalpha():

        print('Your password must at least have 1 number.')



    else:

        print('Your password is correct')



inputPass = '123456a'

passvalidation(inputPass)
```