**Objective of today’s lab:**

Bu the end of this practical you have completed:

* Python programming refresher and getting familiarised with the Astah tool
* Analysis and Design of Home Automation System
* Try Git

As a refresher, the following exercises are similar to the kind of programming exercises that you were required to do in your first year.

**Things to refresh:** Classes, Objects, attributes, methods, Constructors, Destructors, Inheritance, Encapsulation (public, private, protected), Static variables/methods, Polymorphism etc.

**OOP in Python:** [Self-study] If you would like to have a quick reminder of OOP concepts in Python then please review Beginner’s Guide to Python 3 Programming by John Hunt (check reading list: [UFCF8S-30-2-Advanced Software Development | UWE Bristol (talis.com)](https://rl.talis.com/3/uwe/lists/B64DD90B-D73F-4427-EF96-1FF0CF41CC14.html?lang=en-GB)) or here is a well explained online resource from GeeksforGeeks.org with Python examples: <https://www.geeksforgeeks.org/python-classes-and-objects/?ref=lbp>

# **Exercise 1** Reading from file and loop

You must design and write a Python program called **OddEvenAverage** which:

* Loops over the supplied input file (**mynumbers.txt**)
* Finds the integer averages of all even numbers and all odd numbers

Your program output must look exactly like this:

***Average of even numbers is XX***

***Average of odd numbers is YY***

(Where **XX** and **YY** are the numbers you have calculated)

**Hint**.

You can use two arrays, one stores even numbers and the other stores odd numbers while reading input from the file.

# **Exercise 2** Reading from file and loop

A record is kept of the sales made by three people called Sofia, Olivia and Emma. The records show their name, the price of an item and the number of items. E.g.

**Emma 13 3**

Shows that Emma sold 3 items worth $13 each (total value $39).

You must design and write a Python program called **Incomes** which:

* Loops over the supplied input file (**pay.txt**)
* Calculates the total incomes of the 3 people and prints out their totals

The file **pay.txt** should be put into the same package as your **Incomes** program.

Your program output must look exactly like this:

***Income***

**Emma*: $xx***

***Sofia: $yy***

***Olivia: $zz***

Where xx, yy and zz each person's total earnings.

# **Exercise 3 Classes and Objects**

Given the following classes:

The **Holiday** class has:

* Properties of destination (String), duration (int days) and cost (int $).
* All expected getters and setters.
* A fully parameterised constructor (which takes all properties with default values)
* A **\_\_str()\_\_**method which returns a String showing all the properties.

The **TravelAgent** class has:

* Properties of name (String), postcode (String) and holidays (an Array List of Holidays) – complete with getters and setters.
* A parameterised constructor (name and postcode only) which also initialises the holidays Array List.
* A method (**addHoliday**) which adds a holiday to the Travel Agents list.
* A **\_\_str()\_\_** method which returns a String showing all the properties including details of the Holidays.

Implement these two classes. You are supplied with a test file called **RunTravelAgent.py** which should produce this output:

CheapAsChips at MA99 1CU

Holiday{destination=Bermuda, duration=2 days, cost=$800}

Holiday{destination=Hull, duration=14 days, cost=$8}

Holiday{destination=Los Angeles, duration=12 days, cost=$2100}

Holiday h3 Duration=$12 & Cost =$2100

TravelAgent t2 Shoe String Tours CO33 2DX

# **Exercise 4 BankAccount class**

Create a class called BankAccount. The methods listed below should be supported by your class. Create appropriate class instances and call the methods you've written to see your class's output.

class BankAccount:

#Bank Account should be protected by a pin number.

def \_\_init\_\_(self, pinnumber):

#Initial account balance is 100 and pin is 'pinnumber'.

def deposit(self, pinnumber, amount):

#Increment account balance by amount and return the new balance.

def withdraw(self, pinnumber, amount):

#Decrement account balance by amount and return the amount withdrawn.

def get\_balance(self, pinnumber):

#Return the current account balance.

def change\_pin(self, oldpinnumber, newpinnumber):

#Change pin from oldpinnumber to newpinnumber.

# **Exercise 5 Inheritance**

Create a Python class **Person** with attributes: name and age of type string. Create a display() method that displays the name and age of an object created via the Person class. Create a child class **Lecturer** which inherits from the Person class and which also has a lecturerID attribute. Create a method displayLecturer() that displays the name, age and ID of an object created via the Lecturer class. Create a lecturer object via an instantiation on the Lecturer class and then test the displayLecturer() method.

class Person:

# define constructor with name and age as parameters

# create display method for Person class

# create child class Lecturer of Person class

class Lecturer(Person):

# define constructor of Lecturer class with ID as an additional parameter

# Create display method for Lecturer class

# Testing Person class

P = Person("Peter Miller", 30)

P.display()

S = Lecturer("John King", 45 , "j-king")

S.display()

# **Exercise 6 Analysis and Design of Home Automation System**

Design a computer-controlled home automation system which controls the following parts of the house: ventilation, heating, lighting and security/locks, control panel.​

Home owner should be able to define settings for temperature, ventilation, lighting through control panel. Control panel should also display current setting and notify the home owner through pop-ups if any action is being taken due to change in monitored objects e.g., temperature, humidity, light etc. Control panel should also allow authorized users to manually take actions e.g., turn the heating off, turn the garden lights on, open the door lock, etc. Homeowner or authorized users should also be able to access the control panel features and perform all functionalities through a mobile app. ​

​**Task:** How will you approach above problem and what sort of activities you will consider to design and implement the above system?​

# **Exercise 7 Astah tool demo**

If you've finished the above activities, take a look at the tutorials below and try drawing some diagrams. Although your diagrams do not have to be perfect, you should learn how to use the tool.

**Use Case** [**https://www.youtube.com/embed/2sWJ876s4KY**](https://www.youtube.com/embed/2sWJ876s4KY)

**Class diagram** [**https://www.youtube.com/embed/y9s\_S6uRAek**](https://www.youtube.com/embed/y9s_S6uRAek)

**Sequence diagram** [**https://www.youtube.com/embed/Qi2CsTY4LSk**](https://www.youtube.com/embed/Qi2CsTY4LSk)

**Task:** Draw use case model of Home automation system scenario (given in Exercise 6).

# **Exercise 8 Version Control – Git demo**

Version control is an important element of software development. This becomes an important tool when a group of developers are simultaneously working on different parts of the project and have to share code (as well as keep track of changes) with each other.

**Git** is a free and open-source distributed version control system. It can be downloaded from here: <https://git-scm.com/> . In your self-study time you should read Git documentation <https://git-scm.com/doc>

**Ben Gaster** has prepared a step-by-step tutorial here: <http://www.cems.uwe.ac.uk/~br-gaster/courses/GIT/>

Please follow this tutorial to learn how Git can be used. You should be able to create/clone a repository on your machine, make changes i.e. add/remove files and commit changes so that Git can maintain proper versioning.

You should then try <https://gitlab.uwe.ac.uk> and push/pull your changes to/from remote repository/project. You can invite your group members (or tutors) to access remote repository. They can either work in master branch or you can fork out branches for each of your team member to work separately and later merge child branch(es) to master branch. When merging branches, conflicts (e.g., a file is changed/updated in master branch and the same file is updated by your team member in child branch) which must be resolved.

Here is also a nice Git tutorial on youtube which you can enjoy in your self-study time: <https://www.youtube.com/watch?v=4lxvVj7wlZw&ab_channel=ValentinDespa>