Name: Mohamed Ali Mohamed

Session 2

#Task 1: How to hide password while typing?

Sol

```
import getpass
x=getpass.getpass("enter your password?")
```

#Task 2: How to write c++ on jupyter?

Sol

Before you install the modules, you want to set up your own environment to prevent conflicts with your default setup. Open up a terminal and type 'conda activate'. Enter the following commands

```
conda create -n cling
```

Next, you want to install cling to your particular environment.

```
conda install xeus-cling -c conda-forge
```

Finally, install Xeus:

```
conda install xeus -c conda-forge
```

#Task 3: how to write a code that do {do..while} in python?

Sol

The difference between while loop and do while loop that do while has to run at least one time before break but while could be never run so I found that way to make do while loop in python like this :

```
secret_word="python is good"

counter=0

while True:

   word=input("please enter the secret word : ").lower()

   counter=counter+1

   if word==secret_word:

      break

     if word != secret_word and counter>7 :

         break
```

#Task 4:   a python code that pass by 2 one time and pass by times 2 another time ?

#Task 5:make a finite loop using for ?

#Task 6: What is dependency injection?

Sol

Dependency injection is a technique built on the top of the Inversion of Control. The main idea is to separate the construction and usage of objects

#Task 7: informations about clean code

Sol

_Design rules:

1. Keep configurable data at high levels.
2. Prefer polymorphism to if/else or switch/case.
3. Separate multi-threading code.
4. Prevent over-configurability.
5. Use dependency injection.
6. Follow Law of Demeter. A class should know only its direct dependencies.

_ Names rules:

1.Choose descriptive and unambiguous names.

2.Make meaningful distinction.

3.Use pronounceable names.

4.Use searchable names.

5.Replace magic numbers with named constants.

6.Avoid encodings. Don't append prefixes or type information.

_functions rules:

1.Small.

2.Do one thing.

3.Use descriptive names.

4.Prefer fewer arguments.

5.Have no side effects.

6.Don't use flag arguments. Split method into several independent methods that can be called from the client without the flag.

#Task 8: Threads

Sol

What Is a Thread?

A thread is a separate flow of execution. This means that your program will have two things happening at once. But for most Python 3 implementations the different threads do not actually execute at the same time: they merely appear to.

It’s tempting to think of threading as having two (or more) different processors running on your program, each one doing an independent task at the same time. That’s almost right. The threads may be running on different processors, but they will only be running one at a time.

Getting multiple tasks running simultaneously requires a non-standard implementation of Python, writing some of your code in a different language, or using multiprocessing which comes with some extra overhead.

Because of the way CPython implementation of Python works, threading may not speed up all tasks. This is due to interactions with the GIL that essentially limit one Python thread to run at a time.

Tasks that spend much of their time waiting for external events are generally good candidates for threading. Problems that require heavy CPU computation and spend little time waiting for external events might not run faster at all.

This is true for code written in Python and running on the standard CPython implementation. If your threads are written in C they have the ability to release the GIL and run concurrently. If you are running on a different Python implementation, check with the documentation too see how it handles threads.

If you are running a standard Python implementation, writing in only Python, and have a CPU-bound problem, you should check out the multiprocessing module instead.

Architecting your program to use threading can also provide gains in design clarity. Most of the examples you'll learn about in this tutorial are not necessarily going to run faster because they use threads. Using threading in them helps to make the design cleaner and easier to reason about.