 

Session 2

30-08-2022

***Tasks***

**1-Library in python that hidden password** 🡪maskpass.askpass or maskpass.advpass

**Code:**

pwd = maskpass.advpass(prompt="enter:",mask="#$")

print(pwd)

**2-Do while in python**

**Code:**

secret\_word = "python"

counter = 0

while True:

word = input("Enter the secret word: ").lower()

counter = counter + 1

if word == secret\_word:

break

if word != secret\_word and counter > 7:

break

**another code:**

C=true

While c :

If x>y ///do something

Else c=false , /// do something

**3-Infinite for loop without extra memory ->**

**Code:**

for \_ in iter(int,1):

print("1")

# 4- for loop with i\*2 step ->

# Code:

# terms=10

# result = list(map(lambda x: 2 \*\* x, range(terms)))

# for i in range(2,terms):

# print(result[i])

# print(result)

**another code:**

for x in (2\*\*p for p in range(1, 11)):

print(x)



# 

# 5-what is Dependency injection?

# [Dependency injection - YouTube](https://www.youtube.com/watch?v=YO4MGNu2xvI)

# **dependency injection** is a technique whereby one object (or static method) supplies the dependencies of another object. A dependency is an object that can be used (a [service](https://en.wikipedia.org/wiki/Service_(systems_architecture))).

# The Dependency injection pattern solves this problem by injecting dependent objects via a constructor, a property, or an interface.

# -ان الكود ميكونش فيه اعتماديه كليه (يعني ميكونش فيه فانكشن او جزء في الكود بيعتمد عليها السيستم اعتماد كلي)

# 6- what are the clean code rules or principles?

# [Summary of 'Clean code' by Robert C. Martin · GitHub](https://gist.github.com/wojteklu/73c6914cc446146b8b533c0988cf8d29)

# Use descriptive names

# Use empty lines to create a readable code

# Do not send more than three parameters into a function

# Remember the functions must do only one thing

# Hide internal structure

**7-Recursion is faster than Iteration**

As using multi threads and it fasts the operation

[**performance - Is recursion ever faster than looping? - Stack Overflow**](https://stackoverflow.com/questions/2651112/is-recursion-ever-faster-than-looping)

 

***Threads***

A thread is a basic unit of CPU utilization; it comprises

a thread ID, a program counter, a register set, and a

stack. It shares with other threads belonging to the same

process its code section, data section, and otheroperating-system resources, such as open files and signals.A thread is a sequential flow of tasks within a process. Each thread has its own set of registers and stack space. There can be multiple threads in a single process having the same or different functionality. Threads are also termed lightweight processes. Threads are used to increase the performance of the applications. Each thread has its own program counter, stack, and set of registers. But the threads of a single process might share the same code and data/file.

Let us take an example of a human body. A human body has different parts having different functionalities which are working parallelly ( Eg: Eyes, ears, hands, etc). Similarly in computers, a single process might have multiple functionalities running parallelly where each functionality can be considered as a thread.

Threads in the operating system provide multiple benefits and improve the overall performance of the system. Some of the reasons threads are needed in the operating system are:

* Since threads use the same data and code, the operational cost between threads is low.
* Creating and terminating a thread is faster compared to creating or terminating a process.
* Context switching is faster in threads compared to processes



# 

### 1. **User Level Thread:**

User-level threads are implemented and managed by the user and the kernel is not aware of it.

* User-level threads are **implemented using user-level libraries and the OS does not recognize these threads**.
* User-level thread is **faster to create and manage compared to kernel-level thread**.
* Context switching**in user-level threads is faster**.
* If one user-level thread performs a blocking operation then the entire process gets blocked. Eg: POSIX threads, Java threads, etc.

### **2**. **Kernel level Thread:**

**Kernel level threads are implemented and managed by the OS**.

* Kernel level threads are **implemented using system calls and Kernel level threads are recognized by the OS**.
* Kernel-level threads are **slower to create and manage compared to user-level threads**.
* **Context switching in a kernel-level thread is slower**.
* Even if one kernel-level thread performs a blocking operation, it does not affect other threads. Eg: **Window Solaris**.

## Advantages of Threading

* Threads improve the overall performance of a program.
* Threads increases the responsiveness of the program
* Context Switching time in threads is faster.