Q1. What is multiprocessing in python? Why is it useful?

Ans: - Multiprocessing refers to the ability of a system to support more than one processor at the same time.and it will enable the breaking of applications into smaller threads that can run independently.

Multiprocessing reduce latency of particular program, multiple instances able to run easy to possible and it also reliable. Multiprocessing Use in performing parallel computing. There are several processors in a system, and each of them can run multiple processes simultaneously. The system's throughput will be significantly increased as a result of this. Parallel computing is performed by multiprocessing.

Q2. What are the differences between multiprocessing and multithreading?

Multiprocessing	**Multiprogramming**
The availability of more than one processor per system, that can execute several set of instructions in parallel is known as multiprocessing.	The concurrent application of more than one program in the main memory is known as multiprogramming.
The number of CPU is more than one.	The number of CPUs is one.
It takes less time for job processing.	It takes more time to process the jobs.
In this, more than one process can be executed at a time.	In this, one process can be executed at a time.
It is not economical.	It is economical.
The number of users is can be one or more than one.	The number of users is one at a time.
Throughput is maximum.	Throughput is less.
Its efficiency is maximum.	Its efficiency is Less.

Q3. Write a python code to create a process using the multiprocessing module.

Ans:: python code is in ipynb file

Q4. What is a multiprocessing pool in python? Why is it used?

Ans: **multiprocessing Pool** is used for heterogeneous tasks,

Pool will accumulate all the data or collect dat and execute

Class when you need to execute different types of ad hoc tasks, such as calling different target task functions.

Multiprocessing.Pool class when you need to execute many short- to modest-length tasks throughout the duration of your application. Class when the types of tasks of and timing of when you need to execute tasks varies at runtime.

The **multiprocessing.Pool** class when you need to be able to queue up a large number of tasks.

The **multiprocessing.Pool** class when you need to be able to check on the status of tasks during their execution. class when you need to take action based on the results of tasks, such as the first task to complete or results as they become available.

Q5. How can we create a pool of worker processes in python using the multiprocessing module?

Ans: answer is in ipynb file

Q6. Write a python program to create 4 processes, each process should print a different number using the multiprocessing module in python.

Ans: Answer is in ipynb file

4 process

Pool

Process

Pipe

Queue