PRACTICAL 1

import multiprocessing

import os

def execute\_same\_code():

    print(f"Process {os.getpid()} executing same code")

def execute\_different\_code():

    print(f"Process {os.getpid()} executing different code")

def child\_process():

    execute\_same\_code()

def main():

    print(f"Parent process ID: {os.getpid()}")

    # a) Same program, same code

    process\_a = multiprocessing.Process(target=child\_process)

    process\_a.start()

    process\_a.join()

    # b) Same program, different code

    process\_b = multiprocessing.Process(target=execute\_different\_code)

    process\_b.start()

    process\_b.join()

    print("Parent process finished.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

PRACTICAL 5

import subprocess

import sys

def copy\_file(source\_path, destination\_path):

    try:

        # Using system call to copy file (Linux/Unix)

        subprocess.run(["cp", source\_path, destination\_path])

        # For Windows, you can use the following system call:

        # subprocess.run(["copy", source\_path, destination\_path])

        print(f"File copied from {source\_path} to {destination\_path}")

    except Exception as e:

        print(f"Error: {e}")

if \_\_name\_\_ == "\_\_main\_\_":

    # Check if source and destination paths are provided as command-line arguments

    if len(sys.argv) != 3:

        print("Usage: python copy\_file.py <source\_path> <destination\_path>")

        sys.exit(1)

    source\_path = sys.argv[1]

    destination\_path = sys.argv[2]

    copy\_file(source\_path, destination\_path)