Operating System Assignment#1

CLO#	Course Learning Outcome (CLO)	Taxonomy Level	Mapping to PLO
CLO 2	**Demonstrate** knowledge of software solutions to various OS problems that have been developed for provisioning of required functionality and services in operating systems.	C3	2

Q1. Process Duplication and PID Relationship

Write a C program that demonstrates process creation using fork().
Your program should:
- Print the PID (process ID) and PPID (parent process ID) of both the parent and child.
- The child process should sleep for 3 seconds, while the parent prints a message every second until the child terminates.
- Clearly indicate which process is running (parent or child) in each output line.

Expected Output Example:

```
Parent process started. PID = 3456
Child process created. PID = 3457, Parent PID = 3456
Parent is waiting...
Child running...
Child finished.
Parent detected child termination.
```

Q2. Executing a New Program with exec()

Write a C program that uses fork() followed by an exec() call in the child process.
The program should:
- Use execlp() (or any exec variant) to replace the child process with the ls -l command.
- The parent process should wait until the child finishes execution, then print "Child execution completed."

Hint: execlp("ls", "ls", "-l", NULL);

Expected Behavior:

```
When the program runs, you should first see the directory listing (output of ls -l), then:
Child execution completed.
```

Q3. Combining Multiple Forks and Execs

Write a C program that:
- Creates two child processes using fork().

- The first child executes the date command using execvp().
- The second child executes the whoami command using execlp().
- The parent waits for both children to complete and then prints a final message:
  "Both child processes have completed successfully."

Expected Output Example:

```
[Child 1] Executing date command:
Thu Oct 17 15:42:23 PKT 2025
[Child 2] Executing whoami command:
Bilal
Parent: Both child processes have completed successfully.
```