Q1)write an application which creates a process to execute another application in the executable file "/home/hello".your parent process should wait the termination of the child process.Also parent should print "PARENT is WAITING" to he screen before the child process finishes.Child, before executing hello, should print "CHILD executes hello" to the screen .Note that you only need to write the main method.

int main() {

pid\_t pid;

// Create a new process using fork

pid = fork();

if (pid < 0) {

// Fork failed

perror("Fork failed");

return 1;

} else if (pid == 0) {

// This block is executed by the child process

printf("CHILD executes hello\n");

// Replace the current process with the executable at "/home/hello"

execl("/home/hello", "hello", (char \*)NULL);

// If execl() fails

perror("execl failed");

return 1;

} else {

// This block is executed by the parent process

printf("PARENT is WAITING\n");

// Wait for the child process to terminate

wait(NULL);

printf("Child process finished\n");

}

return 0;

}

Q2)write a possible output of the following application :

int main()

{

pid,\_t n,int x=2;

n=fork();

if(n>0)

{

x=x+7;

wait(NULL);

}

else if (n==0)

{

x=x+10;

execlp("/bin/ls","ls",NULL);

}

execlp("x is%d\n",x);

return 0;

}

Answer:

The child process runs ls, which prints the list of files and directories in the current working directory.

The parent process prints x is 9 after the child process has completed.

Note:

The value x = 12 in the child process is lost because execlp() replaces the child process image with the ls program.

The original code had an incorrect use of execlp() at the end, which would have caused a runtime error if reached.