**Operating Systems - Lab 04 Post-Lab Tasks**

Ibrahim Johar Farooqi

23K-0074

BAI-4A

12 February 2025

**Post-Lab Questions:**

**Task 1:**

**#include <iostream>**

**#include <sys/types.h>**

**#include <sys/wait.h>**

**#include <unistd.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t child1, child2;**

**child1 = fork();**

**if(child1 == 0)**

**{**

**cout << "child 1: my pid is " << getpid() << "\n";**

**return 0;**

**}**

**child2 = fork();**

**if(child2 == 0)**

**{**

**cout << "child 2: my parent pid is "<< getppid() << "\n";**

**return 0;**

**}**

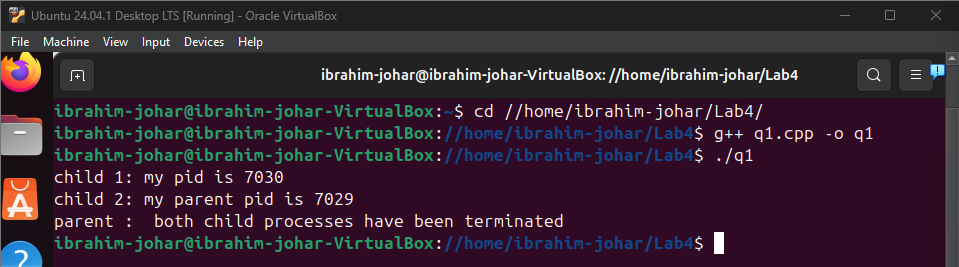
**wait(NULL);**

**wait(NULL);**

**cout << "parent : both child processes have been terminated\n";**

**return 0;**

**}**

****

**Task 2:**

**#include <iostream>**

**#include <unistd.h>**

**#include <sys/types.h>**

**#include <sys/wait.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t pid = fork();**

**if(pid<0)**

**{**

**cerr << "fork failed\n";**

**return 1;**

**}**

**else if(pid == 0)**

**{**

**for(int i = 0; i < 100; i++)**

**{**

**cout << "i am a child process.\n";**

**}**

**}**

**else**

**{**

**for(int i = 0; i < 100; i++)**

**{**

**cout << "i am a parent process.\n";**

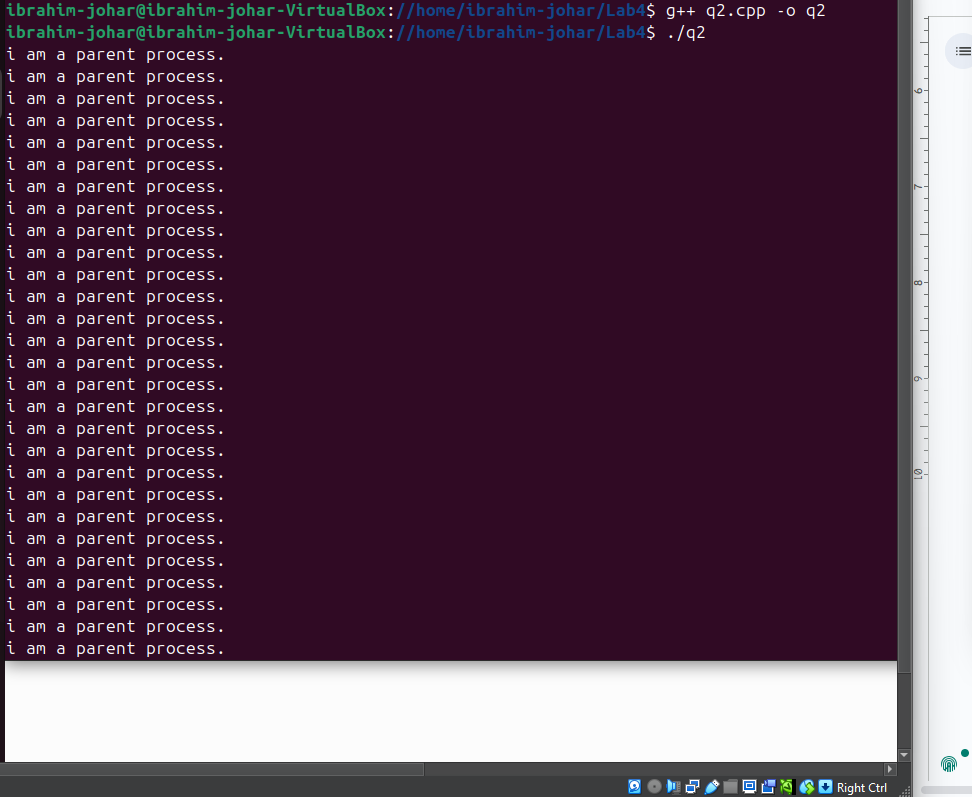
**}**

**wait(NULL);**

**}**

**return 0;**

**}**



**Task 3:**

**#include <iostream>**

**#include <unistd.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t pid = getpid();**

**cout << "current process id: "<< pid << "\n";**

**pid\_t ppid= getppid();**

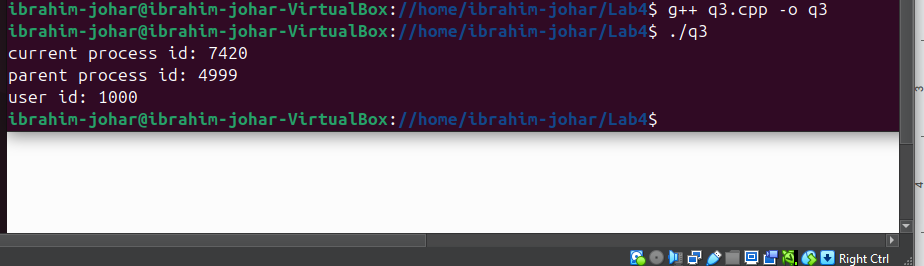
**cout<<"parent process id: "<< ppid << "\n";**

**uid\_t uid = getuid();**

**cout<<"user id: "<< uid << "\n";**

**return 0;**

**}**

****

**Task 4:**

**#include <iostream>**

**#include <fstream>**

**using namespace std;**

**int main()**

**{**

**ifstream input\_file("input.txt");**

**ofstream output\_file("output.txt");**

**if(!input\_file)**

**{**

**cerr << "error: unable to open input file!\n";**

**return 1;**

**}**

**if(!output\_file)**

**{**

**cerr << "error: unable to open output file!\n";**

**return 1;**

**}**

**string line;**

**while(getline(input\_file,line))**

**{**

**output\_file<<line<<"\n";**

**}**

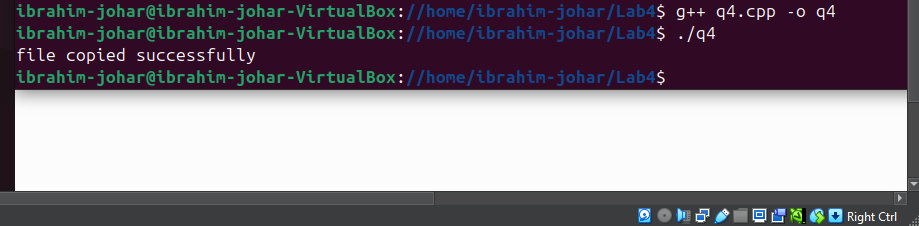
**input\_file.close();**

**output\_file.close();**

**cout<<"file copied successfully\n";**

**return 0;**

**}**



**Task 5:**

**#include <iostream>**

**#include <sys/types.h>**

**#include <sys/wait.h>**

**#include <unistd.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t pid = fork();**

**if(pid == 0)**

**{**

**cout << "child process: listing all files in directory: " << "\n";**

**execlp("ls","ls","-l", NULL);**

**cerr << "excelp failed\n";**

**return 1;**

**}**

**else**

**{**

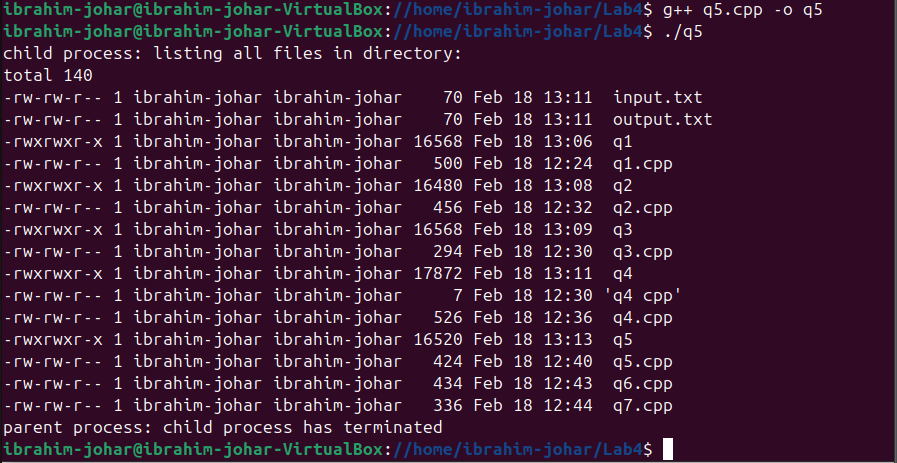
**wait(NULL);**

**cout << "parent process: child process has terminated\n";**

**}**

**return 0;**

**}**



**Task 6:**

**#include <iostream>**

**#include <unistd.h>**

**#include <signal.h>**

**using namespace std;**

**void handlealarm(int signum)**

**{**

**cout << "alarm received, program terminating now.\n";**

**exit(0);**

**}**

**int main()**

**{**

**signal(SIGALRM,handlealarm);**

**alarm(5);**

**cout << "alarm set, entering sleep loop now\n";**

**int count = 1;**

**while(true)**

**{**

**cout << "sleeping..." << count << " seconds elapsed.\n";**

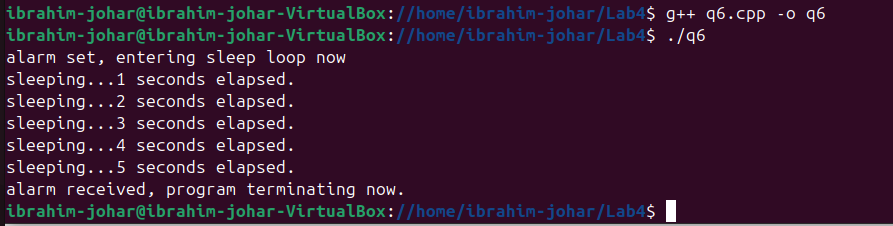
**sleep(1);**

**count++;**

**}**

**return 0;**

**}**

****

**Task 7:**

**#include <iostream>**

**#include <unistd.h>**

**#include <sys/wait.h>**

**using namespace std;**

**int main()**

**{**

**pid\_t pid = fork();**

**if(pid == 0)**

**{**

**execlp("ls","ls","-l", NULL);**

**cerr<<"excelp failed\n";**

**return 1;**

**}**

**else**

**{**

**wait(NULL);**

**cout<<"parent process. waiting for child process to terminate.\n";**

**}**

**return 0;**

**}**

