OPERATING SYSTEMS LABORATORY (CS3009)

NAME: ATISH KUMAR

ROLL No: 120CS0173

Date - 03/08/22

1) Write all linux Commands used along with their description?

Answer:

i. **pwd –** Stands for 'present working directory'. As the name suggests, this command prints the full name of the directory that we are currently working in.
ii. **cd –** Stands for 'change directory'. We can change the current working directory to the specified directory by using the command as follows: *cd desired Directory Path*
iii. **mkdir –** Stands for 'make directory'. Using this command, we can create a new directory by specifying its name. Example: *mkdir dirName*
iv. **ls –** This command lists all the directories and files in the current working directory in alphabetical order. This command uses different types of flags to produce different kinds of results. Some of these flags are as follows:
    a. **-a** – Prints even the names of the hidden files and directories.
    b. **-r** – Prints the list of directories and files in reverse alphabetical order
    c. **-l** – Prints the list in a long listing format, that is, with all the details associated with a file or directory
    d. **-R** – Prints all the subdirectories and subfiles present in the current working directory in a recursive manner.
v. **mv –** We can move one or more than one, directories and files from source to desired destination. Syntax for using this command is as follows:
    a. For moving only one file or directory:
       mv sourcePath/fileName destinationPath
    b. For moving multiple files or directories:
       mv file1 file2 file3 destinationPath

vi. **<u>cp</u> –** Copies the desired file(s) and directory (or directories) from source to destination. The command is as follows:

cp sourcePath/fileName destinationPath

vii. **<u>top</u> -** The top program provides a dynamic real-time view of a running system. It can display system summary information as well as a list of processes or threads currently being managed by the Linux kernel. The types of system summary information shown and the types, order and size of information displayed for processes are all user configurable and that configuration can be made persistent across restarts.

viii. **<u>ps</u> -** Reports a snapshot of the current processes.

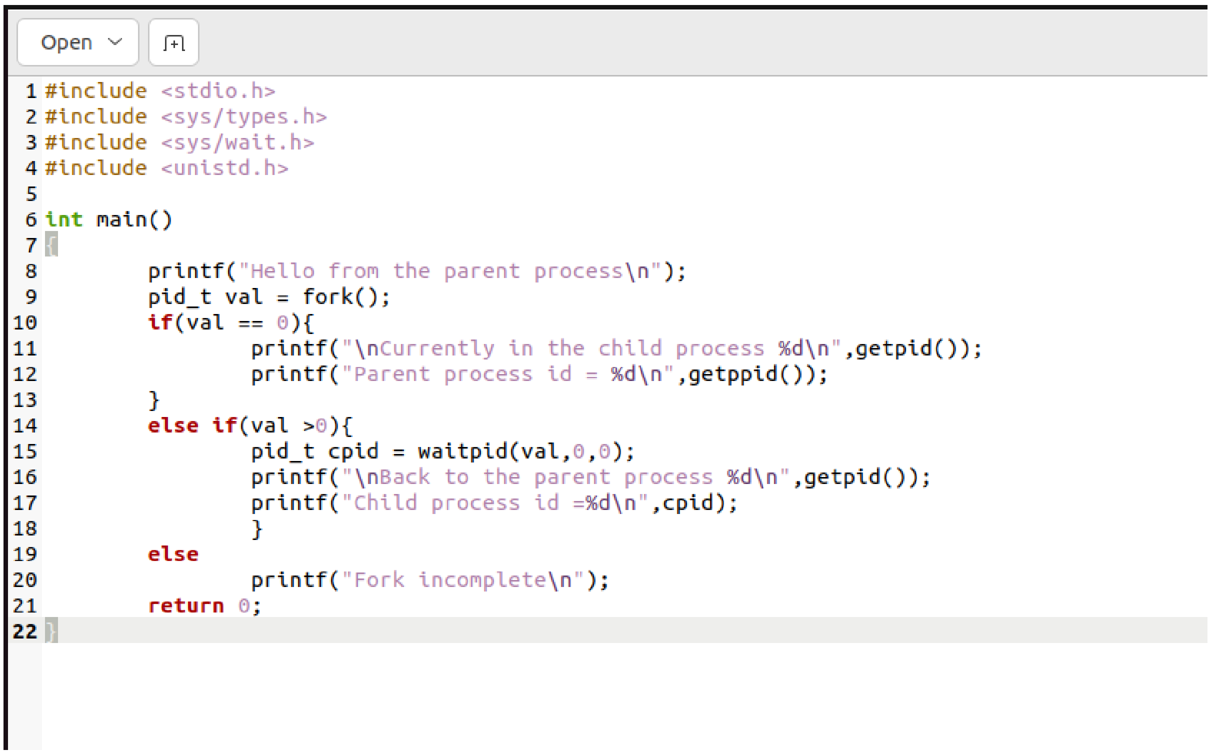
ix. **<u>iostat</u>** - Reports CPU (Central Processing Unit) statistics and input/output statistics for devices and partitions.

ATISH KUMAR (120CS0173)

Date - 10/08/22

Q1) Write a program to demonstrate fork() call. Show the process ID of parent process in its child process and vice-versa.

Code:

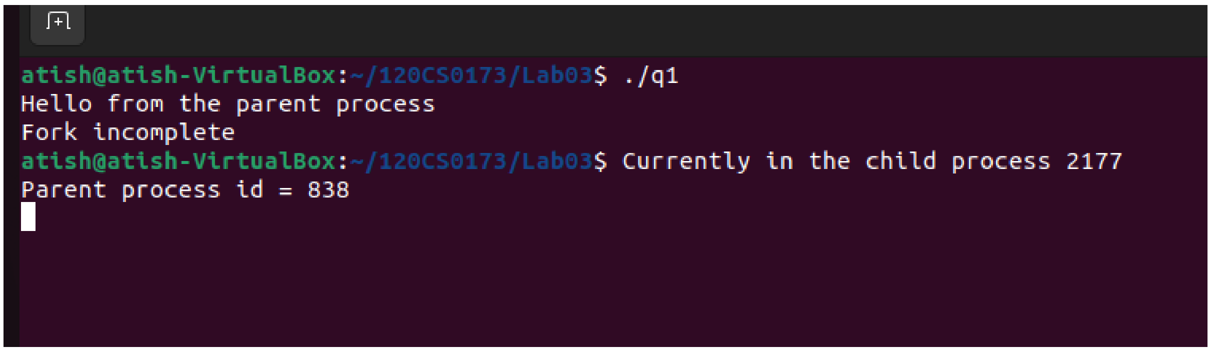


Open

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>

int main()
{
        printf("Hello from the parent process\n");
        pid_t val = fork();
        if(val == 0){
                printf("\nCurrently in the child process %d\n",getpid());
                printf("Parent process id = %d\n",getppid());
        }
        else if(val >0){
                pid_t cpid = waitpid(val,0,0);
                printf("\nBack to the parent process %d\n",getpid());
                printf("Child process id =%d\n",cpid);
                }
        else
                printf("Fork incomplete\n");
        return 0;
}
```

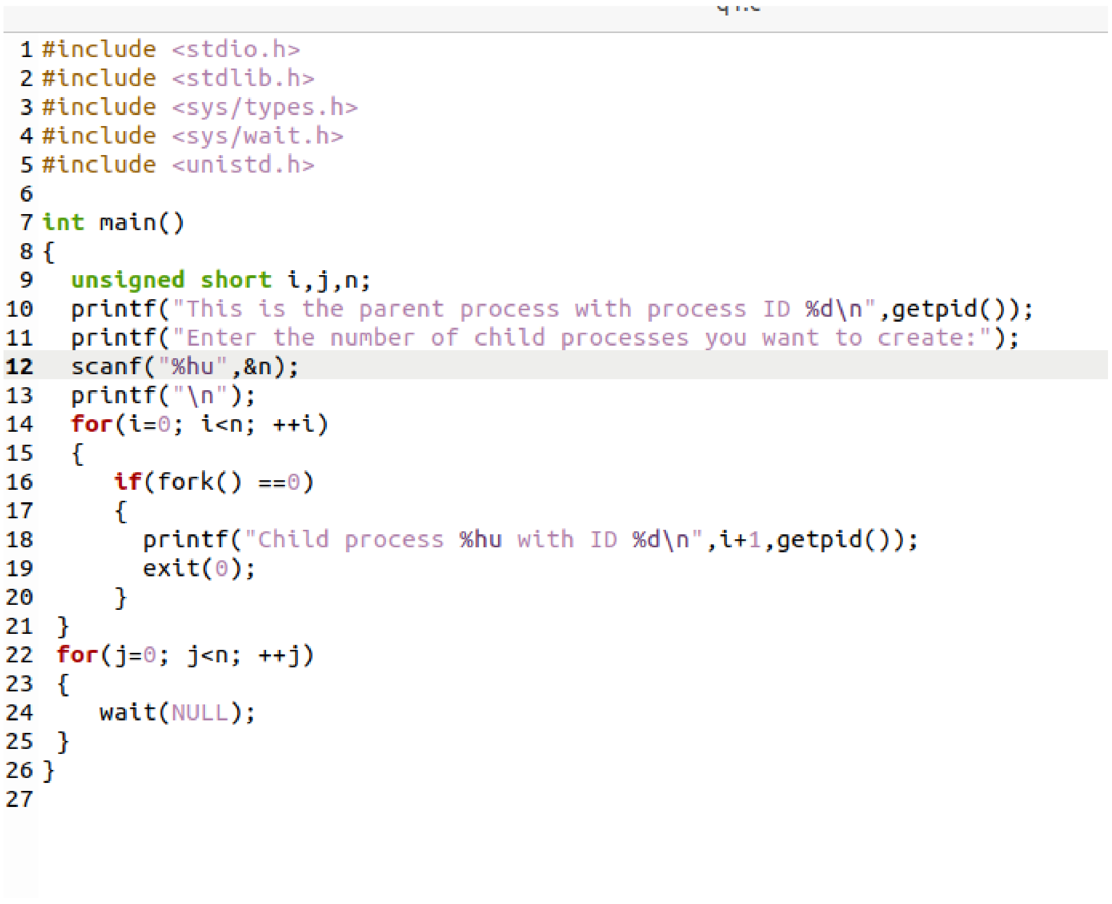
Output:



```
atish@atish-VirtualBox:~/120CS0173/Lab03$ ./q1
Hello from the parent process
Fork incomplete
atish@atish-VirtualBox:~/120CS0173/Lab03$ Currently in the child process 2177
Parent process id = 838
```

Q2) Write a program to create N number of child processes of the parent process using fork(), getpid(), getppid() and wait() functions

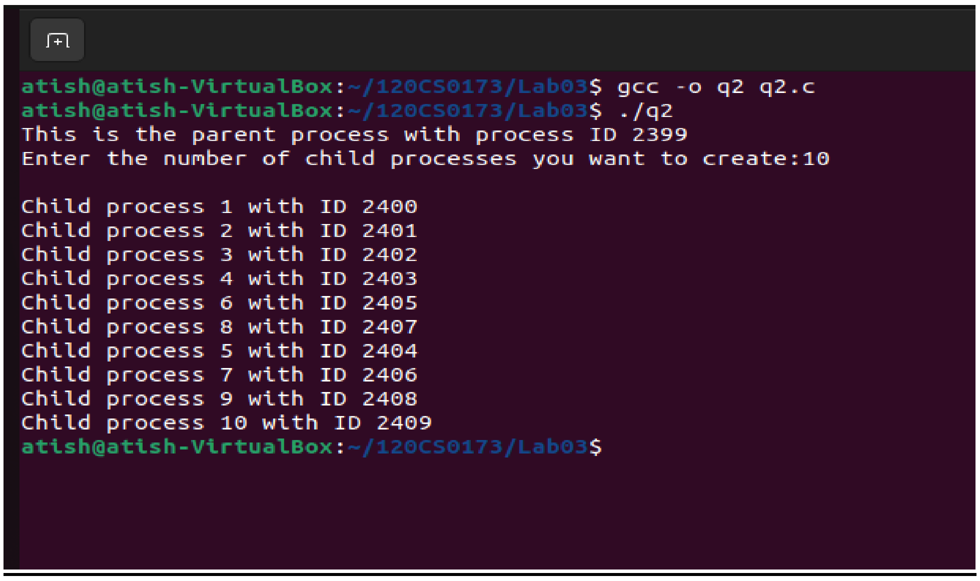
Code:



```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>

int main()
{
  unsigned short i,j,n;
  printf("This is the parent process with process ID %d\n",getpid());
  printf("Enter the number of child processes you want to create:");
  scanf("%hu",&n);
  printf("\n");
  for(i=0; i<n; ++i)
  {
     if(fork() ==0)
     {
       printf("Child process %hu with ID %d\n",i+1,getpid());
       exit(0);
     }
 }
 for(j=0; j<n; ++j)
 {
    wait(NULL);
 }
}
```

Output:



```
atish@atish-VirtualBox:~/120CS0173/Lab03$ gcc -o q2 q2.c
atish@atish-VirtualBox:~/120CS0173/Lab03$ ./q2
This is the parent process with process ID 2399
Enter the number of child processes you want to create:10

Child process 1 with ID 2400
Child process 2 with ID 2401
Child process 3 with ID 2402
Child process 4 with ID 2403
Child process 6 with ID 2405
Child process 8 with ID 2407
Child process 5 with ID 2404
Child process 7 with ID 2406
Child process 9 with ID 2408
Child process 10 with ID 2409
atish@atish-VirtualBox:~/120CS0173/Lab03$
```