1. How many child processes are created upon execution of this program?

3 child processes are created when this file is executed.

2. When you start a browser, you will notice the browser process appear in the top display. What does it consume?

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
COMMAND
   1048 tallulah  20   0 5925608 333052 132488 S  18.3   6.0   0:38.98
gnome-s+
   1702 tallulah  20   0 3498272 404044 154760 S   3.3   7.3   0:25.95
firefox
   3486 tallulah  20   0  562320  52712  40060 S   3.0   0.9   0:00.88
```

3. How much memory is available in the system?

```
top - 18:13:11 up 6 min,  1 user,  load average: 0.28, 0.24, 0.11
Tasks: 247 total,   1 running, 246 sleeping,   0 stopped,   0 zombie
%Cpu(s):  2.8 us,  0.8 sy,  0.0 ni, 96.5 id,  0.0 wa,  0.0 hi,  0.0 si
,  0.0 st
MiB Mem :   5434.6 total,   2919.6 free,   1152.7 used,   1362.3 buff/
cache
MiB Swap:   2278.0 total,   2278.0 free,      0.0 used.   4007.2 avail
 Mem
```

4. Which process consumes the most CPU?

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
COMMAND
   1048 tallulah  20   0 5925608 333052 132488 S  18.3   6.0   0:38.98
gnome-s+
   1702 tallulah  20   0 3498272 404044 154760 S   3.3   7.3   0:25.95
firefox
   3486 tallulah  20   0  562320  52712  40060 S   3.0   0.9   0:00.88
gnome-t+top - 18:13:12 up 6 min,  1 user,  load average: 0.28, 0.24,
0.11
```

5. Which process has the most memory?

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
COMMAND
   1048 tallulah  20   0 5925608 333052 132488 S  18.3   6.0   0:38.98
gnome-s+
   1702 tallulah  20   0 3498272 404044 154760 S   3.3   7.3   0:25.95
firefox
   3486 tallulah  20   0  562320  52712  40060 S   3.0   0.9   0:00.88
gnome-t+top - 18:13:12 up 6 min,  1 user,  load average: 0.28, 0.24,
0.11
```

6. Could you please explain the following commands?
   apt-get, yum, wget, gzip, tar, rar

Apt-get: managing packages, can us I to install, remove, update and upgrade packages in unbutu

Yum: managing packages in red-hat based Linux; to install, remove, update and upgrade packages

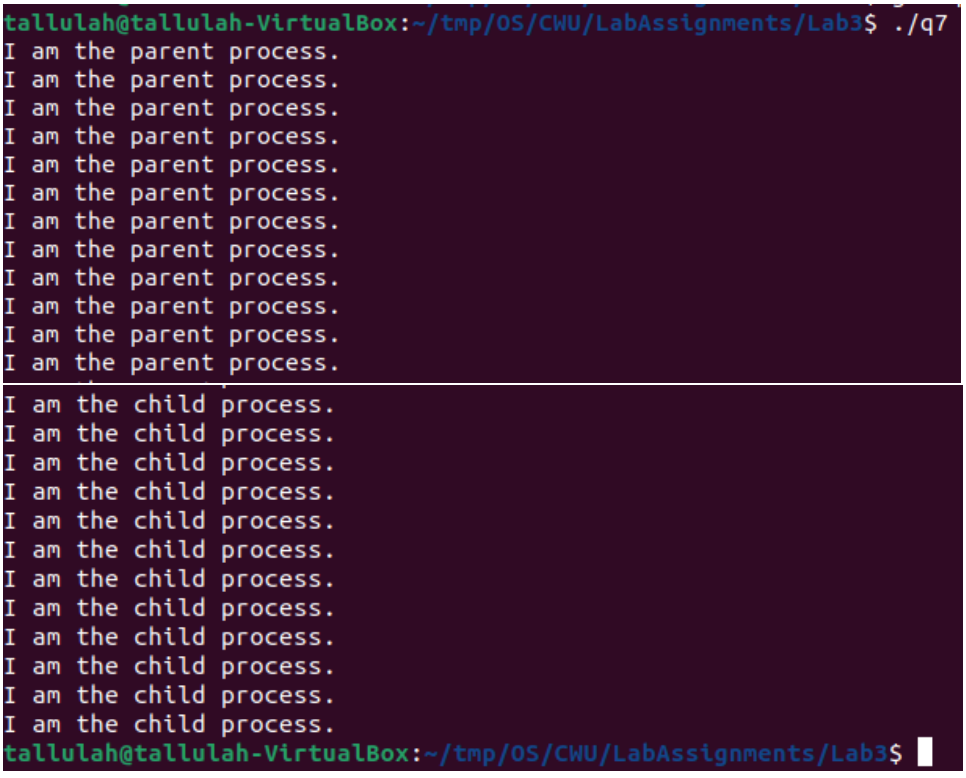
Wget: download files from the internet

Gzip: compress and decompress files

Tar: create extract and archive files

Rar: commercial software for RAR files for compressing archives

7. Write a program that will generate a child process. In a loop, the child process writes "I am a child process" 200 times and the parent process repeatedly prints "I am a parent process" in a loop.



```c
#include <unistd.h>
#include <stdio.h>

int main(){
        pid_t pid;
        pid = fork();
        for(int i=1;i<=200;i++){
                if(pid<0){
                        printf("Fork failed.\n");
                        return 1;
                }else if(pid == 0){
                        printf("I am the child process.\n");
                }else{
                        printf("I am the parent process.\n");
                }

        }
        return 0;
```

8. Write a program that create a child process with the fork () system call. The parent process waits for the child process to finish before printing the contents of the current directory.

```
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$ ./q8
this is the child process
this is the child process
this is the parent process
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$
```

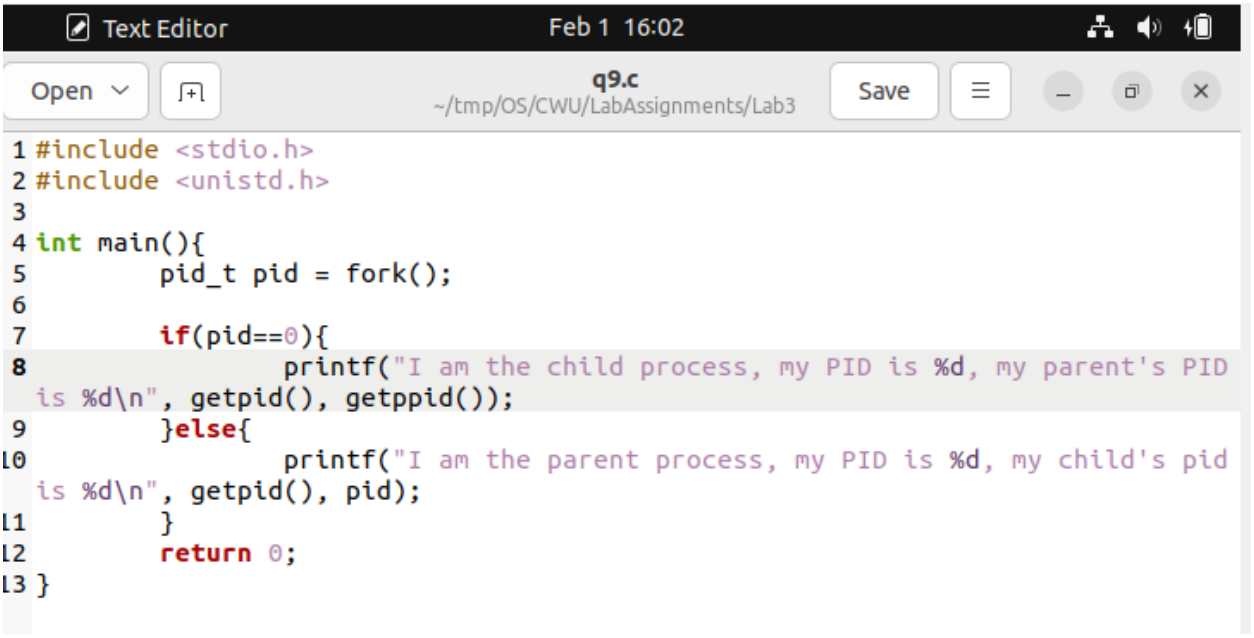
```c
#include <unistd.h>
#include <stdio.h>

#include <sys/wait.h>

int main(){
        pid_t pid, pid2, pid3;
        pid = fork();
        if(pid == 0){
                fork();
                pid2 = getpid();
                printf("this is the child process\n");
        }else{
                wait(NULL);
                //system("ls -l");
                pid3 = getpid();
                printf("this is the parent process\n");
        }
}
```

9. Write a program that create a child process with the fork () system call and print its PID. Following a fork () system call, both parent and child processes print their process type and PID. Additionally, the parent process prints the PID of its child, and the child process prints the PID of its parent.

```
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$ gedit q9.c
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$ gcc -g q9.c -o q
9
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$ ./q9
I am the parent process, my PID is 2153, my child's pid is 2154
I am the child process, my PID is 2154, my parent's PID is 2153
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab3$
```




```c
#include <stdio.h>
#include <unistd.h>

int main(){
        pid_t pid = fork();

        if(pid==0){
                printf("I am the child process, my PID is %d, my parent's PID is %d\n", getpid(), getppid());
        }else{
                printf("I am the parent process, my PID is %d, my child's pid is %d\n", getpid(), pid);
        }
        return 0;
}
```