

OS Lab 3

Aniruddha Amit Dutta

180905488

Q1.

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>
#include <sys/wait.h>
int main(){
    printf(" I am the previous program \n");
    int status;
    pid_t pid;
    pid = fork();
    if(pid == -1){
        printf(" child not created \n");
        exit(0);
    }

    else if(pid==0){
        printf(" i am in child \n");
        exit(0);
    }

    else{
        wait(&status);
        printf(" i am in parent\n");
        printf(" child returned = %d\n", status);
    }

    return 0;
}
```

```
Terminal
File Edit View Search Terminal Help
$ cd Desktop
$ gcc q1.c
$ ./a.out
i am in child
i am in parent
child returned = 0
$
```

//

Q2.

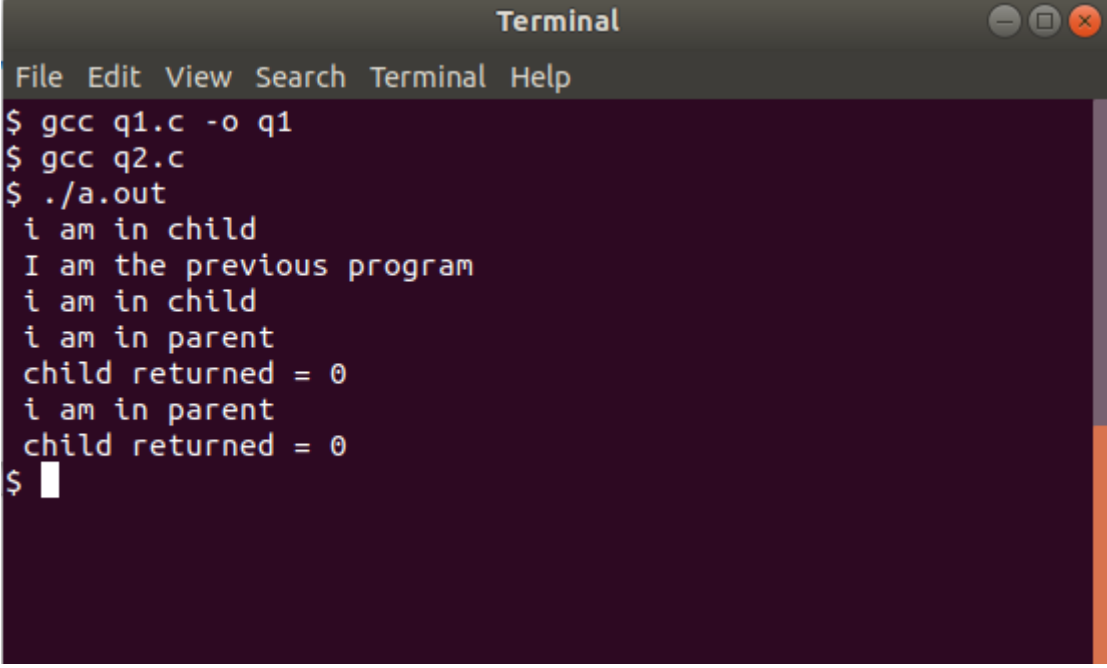
```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>
#include <sys/wait.h>
int main(){
    int status;
    pid_t pid;
    pid = fork();
    if(pid == -1){
        printf(" child not created \n");
        exit(0);
    }

    else if(pid==0){ /* child process */
        printf(" i am in child \n");
        execlp("/home/student/Desktop/q1", "ls",NULL);
        exit(0);
    }

    else{
        wait(&status);
        printf(" i am in parent\n");
        printf(" child returned = %d\n", status);
    }

    return 0;
}
```

}

A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
$ gcc q1.c -o q1
$ gcc q2.c
$ ./a.out
i am in child
I am the previous program
i am in child
i am in parent
child returned = 0
i am in parent
child returned = 0
$
```

//

Q3.

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>
#include <sys/wait.h>
int main(){
    int status;
    pid_t pid;
    pid = fork();
    if(pid == -1){
        printf(" child not created \n");
        exit(0);
    }

    else if(pid==0){
        printf(" i am in child with PID : %d\n",getpid());
        printf(" parent PID : %d\n",getppid());
        exit(0);
    }

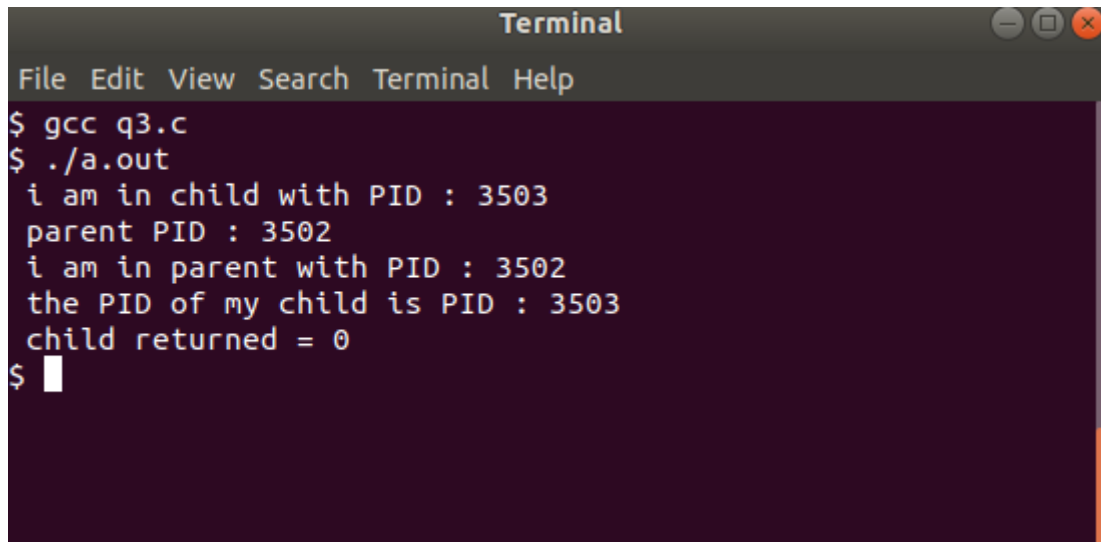
    else{
        wait(&status);
        printf(" i am in parent with PID : %d\n",getpid());
    }
}
```

```

        printf(" the PID of my child is PID : %d\n",pid );
        printf(" child returned = %d\n", status);
    }

    return 0;
}

```



```

Terminal
File Edit View Search Terminal Help
$ gcc q3.c
$ ./a.out
i am in child with PID : 3503
parent PID : 3502
i am in parent with PID : 3502
the PID of my child is PID : 3503
child returned = 0
$ 

```

```
//
```

Q4.

```
// orphan
```

```

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

int main()
{
    char *command = "pstree -p ";
    // Fork returns process id
    // in parent process
    pid_t child_pid = fork();

    // Parent process
    if (child_pid > 0) {
        printf(" i am PARENT , PID: %d\n",getpid() );
    }
}

```

```

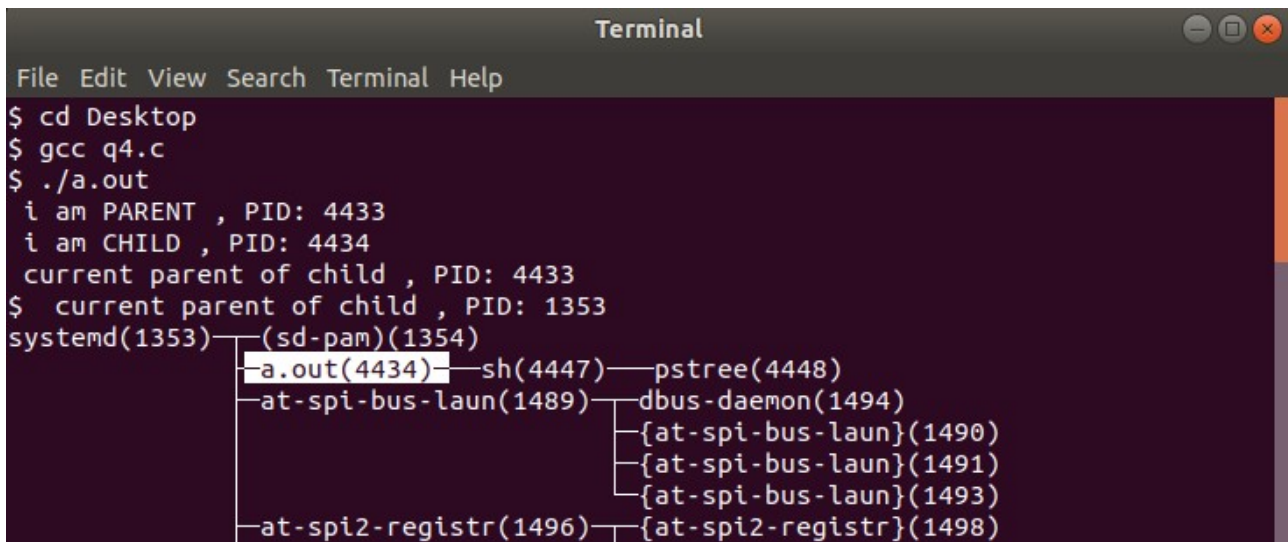
    sleep(5);
}

// Child process
else{
    printf(" i am CHILD , PID: %d\n",getpid() );
    printf(" current parent of child , PID: %d\n",getppid() );
    sleep(6);
    printf(" current parent of child , PID: %d\n",getppid() );
    char buf[32];

    sprintf(buf,"pstree -p %d",getppid());
    printf(" parent of %d is %d \n",getppid(),system(buf) );
    sleep(1);
    exit(0);
}

return 0;
}

```



The image shows a terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The user has navigated to the Desktop directory, compiled a C program named q4.c into a.out, and executed it. The program's output shows it running as the parent process (PID 4433) and then spawning a child process (PID 4434). The child process then prints the output of the 'pstree' command for its parent (PID 1353). The resulting process tree is displayed as follows:

```

systemd(1353)─(sd-pam)(1354)
               └─a.out(4434)─sh(4447)─pstree(4448)
                  └─at-spi-bus-laun(1489)─dbus-daemon(1494)
                                         ├──{at-spi-bus-laun}(1490)
                                         ├──{at-spi-bus-laun}(1491)
                                         └──{at-spi-bus-laun}(1493)
                  └─at-spi2-registr(1496)─{at-spi2-registr}(1498)

```

```
// zombie
```

```

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

```

```

int main()
{

```

```

    // in parent process
    pid_t child_pid = fork();

```

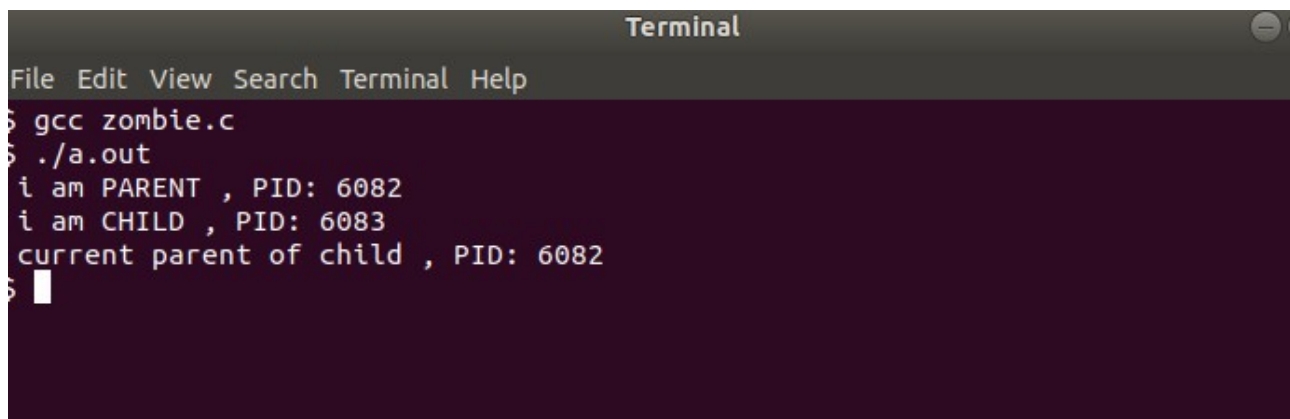
```
// Parent process
if (child_pid > 0) {
    printf(" i am PARENT , PID: %d\n",getpid() );
    sleep(5);
}

// Child process
else{

    printf(" i am CHILD , PID: %d\n",getpid() );
    printf(" current parent of child , PID: %d\n",getppid() );

    exit(0);
}

return 0;
}
```



```
Terminal
File Edit View Search Terminal Help
$ gcc zombie.c
$ ./a.out
i am PARENT , PID: 6082
i am CHILD , PID: 6083
current parent of child , PID: 6082
$
```