Advanced Unix Programming
Lab 8

Purva Tendulkar : 111403049

Q1. Create a new system call wait2, which extends the wait system call.

int wait2(int *wtime, int *rtime, int *iotime)

Where the three arguments are pointers to integers to which the wait2 function will assign:

a. The aggregated number of clock ticks during which the process waited (was able to run but did not get CPU)

b. The aggregated number of clock ticks during which the process was running

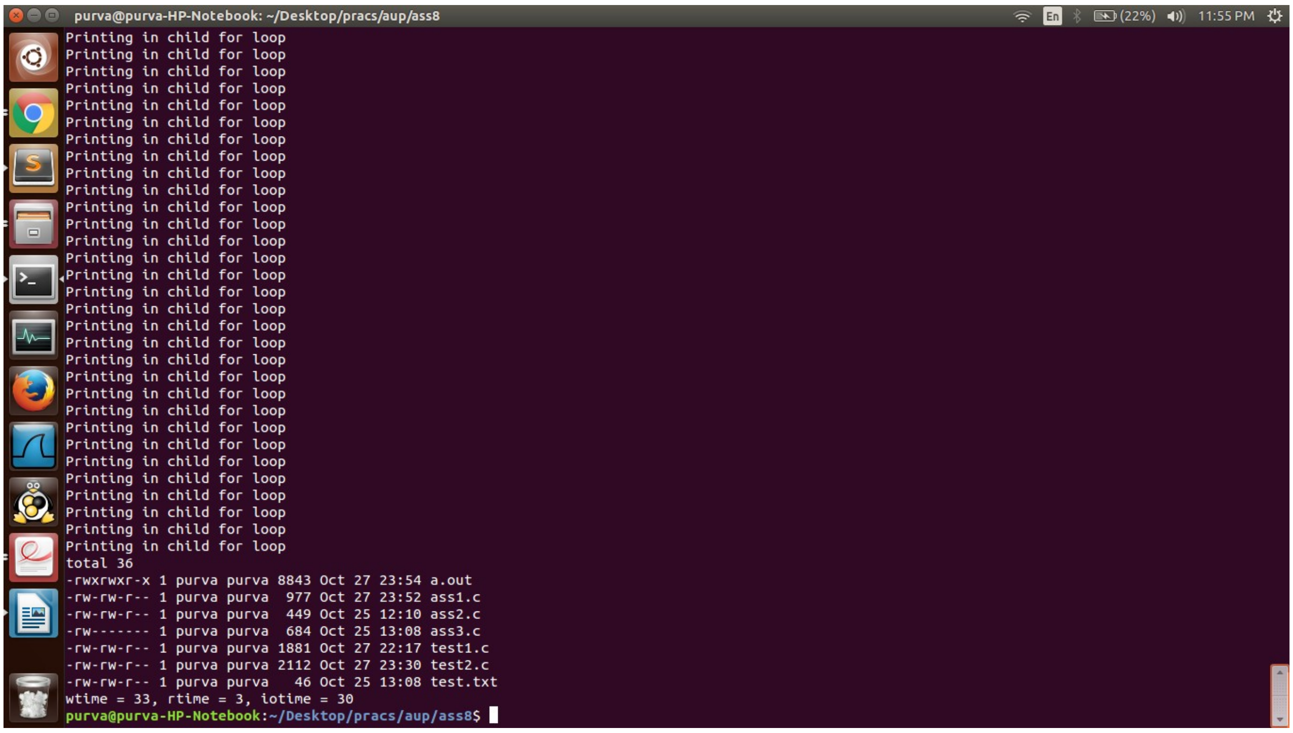
c. The aggregated number of clock ticks during which the process was waiting for I/O (was not able to run).

The wait2 function shall return the pid of the child process caught or -1 upon failure

Code :

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

pid_t wait2(int *wtime, int *rtime, int *iotime) {
	pid_t child;
	struct tms buf;
	int status;

	child = wait(&status);
	times(&buf);

	*wtime = (int)(buf.tms_cstime);
	*rtime = (int)(buf.tms_cutime);
	*iotime = *wtime - *rtime;

	return child;
}

int main(int argc, char *argv[]) {
	pid_t pid;
	int x, wtime, rtime, iotime, i;
	char cmd1[] = "ls"; char *args1[] = {"ls", "-l", NULL};

	pid = fork();
	if (pid < 0) {
		printf("Fork error\n");
		return 1;
	}
	if (pid == 0) {
		/* Child */
		for (i = 0; i < 100000; i++)
			printf("Printing in child for loop\n");
		execvp(cmd1, args1);
	}
	else {
		/* Parent */
		x = wait2(&wtime, &rtime, &iotime);
		if (x == -1) {
			printf("wait2 failure\n");
			return 1;
		}

		printf("wtime = %d, rtime = %d, iotime = %d\n", wtime, rtime, iotime);
	}
	return 0;
}
```

Input & Output Screenshots :



```
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
Printing in child for loop
total 36
-rwxrwxr-x 1 purva purva 8843 Oct 27 23:54 a.out
-rw-rw-r-- 1 purva purva  977 Oct 27 23:52 ass1.c
-rw-rw-r-- 1 purva purva  449 Oct 25 12:10 ass2.c
-rw------- 1 purva purva  684 Oct 25 13:08 ass3.c
-rw-rw-r-- 1 purva purva 1881 Oct 27 22:17 test1.c
-rw-rw-r-- 1 purva purva 2112 Oct 27 23:30 test2.c
-rw-rw-r-- 1 purva purva   46 Oct 25 13:08 test.txt
wtime = 33, rtime = 3, iotime = 30
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$ 
```

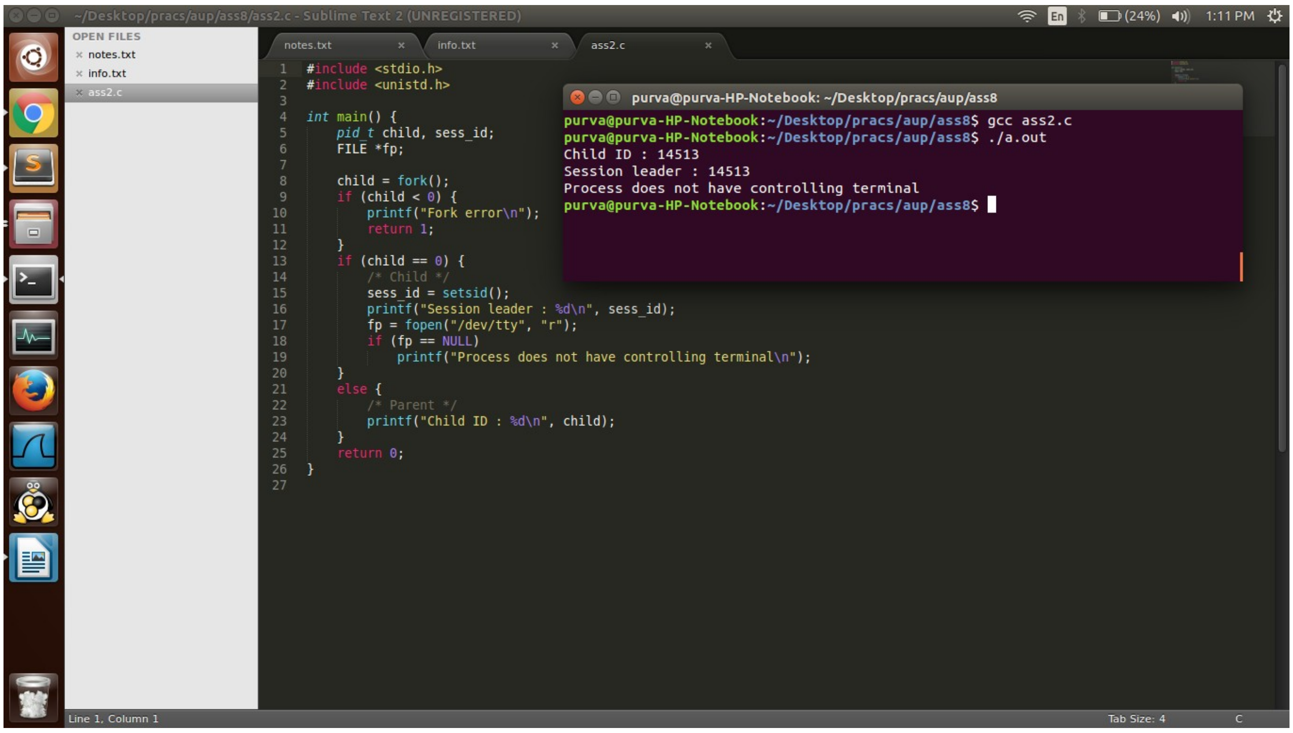
Q2. Call fork. Let the child create a new session. Verify that the child becomes the process group leader and it does not have a controlling terminal.

Code :

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

```
int main() {
        pid_t child, sess_id;
        FILE *fp;

        child = fork();
        if (child < 0) {
                printf("Fork error\n");
                return 1;
        }
        if (child == 0) {
                /* Child */
                sess_id = setsid();
                printf("Session leader : %d\n", sess_id);
                fp = fopen("/dev/tty", "r");
                if (fp == NULL)
                        printf("Process does not have controlling terminal\n");
        }
        else {
                /* Parent */
                printf("Child ID : %d\n", child);
        }
        return 0;
}
```

Input & Output Screenshots :


~/Desktop/pracs/aup/ass8/ass2.c - Sublime Text 2 (UNREGISTERED)
OPEN FILES
notes.txt
info.txt
ass2.c
Line 1, Column 1
Tab Size: 4
C

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

int main() {
    pid_t child, sess_id;
    FILE *fp;

    child = fork();
    if (child < 0) {
        printf("Fork error\n");
        return 1;
    }
    if (child == 0) {
        /* Child */
        sess_id = setsid();
        printf("Session leader : %d\n", sess_id);
        fp = fopen("/dev/tty", "r");
        if (fp == NULL)
            printf("Process does not have controlling terminal\n");
    }
    else {
        /* Parent */
        printf("Child ID : %d\n", child);
    }
    return 0;
}
```

```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$ gcc ass2.c
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$ ./a.out
Child ID : 14513
Session leader : 14513
Process does not have controlling terminal
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$
```

Q3. Write a program to verify that a parent process can change the process group ID of one of its children before the child performs an exec(), but not afterward.

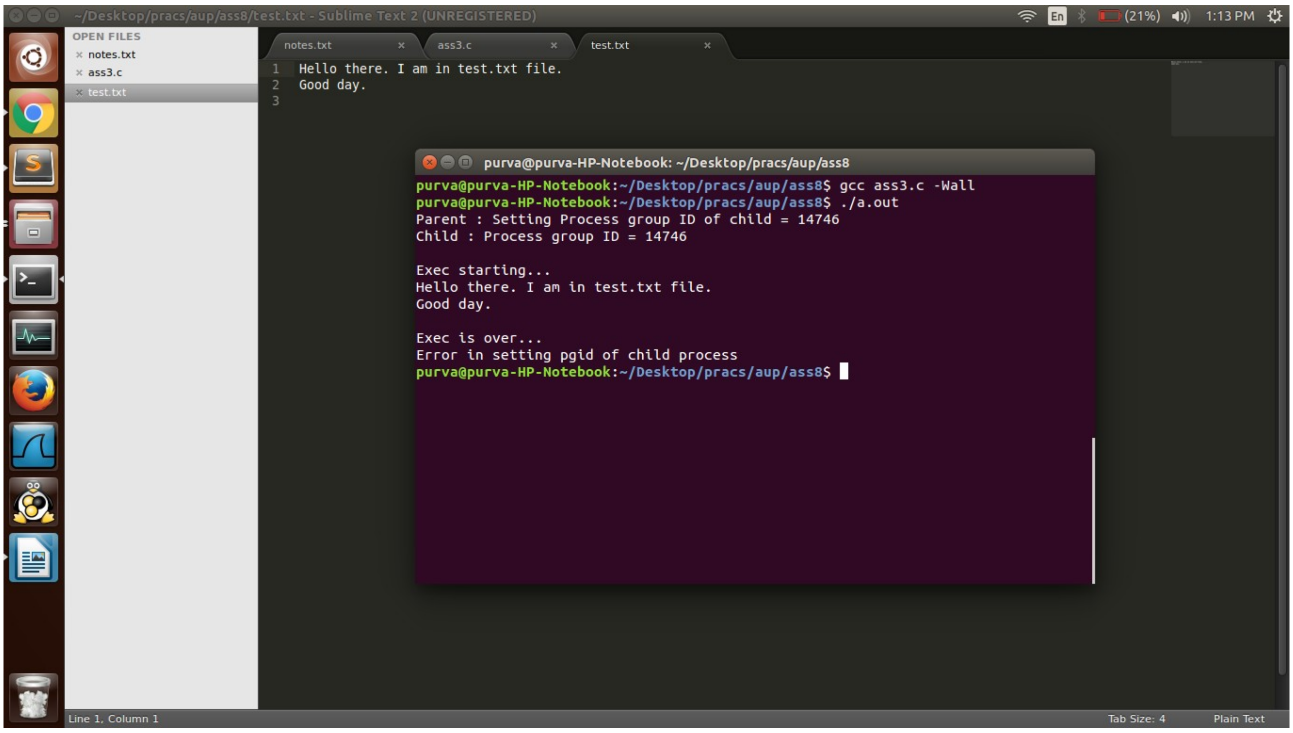
Code :
(A) ass2.c

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

int main() {
	pid_t pid;
	int x;
	char cmd1[] = "cat"; char *args1[] = {"cat", "test.txt", NULL};

	pid = fork();
	if (pid < 0) {
		printf("Fork error\n");
		return 1;
	}
	if (pid == 0) {
		/* Child */
		sleep(3);
		printf("Child : Process group ID = %d\n\n", getpgid(0));
		printf("Exec starting...\n");
		execvp(cmd1, args1);
	}
	else {
		/* Parent */
		printf("Parent : Setting Process group ID of child = %d\n", pid);
		setpgid(pid, pid);
		wait(&x);
		printf("\nExec is over...\n");
		x = setpgid(pid, pid);
		if (x == -1)
			printf("Error in setting pgid of child process\n");
	}
	return 0;
}
```

(B) test.txt

```
Hello there. I am in test.txt file.
Good day.
```

Input & Output Screenshots :



```
Hello there. I am in test.txt file.
Good day.
```

```
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$ gcc ass3.c -Wall
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$ ./a.out
Parent : Setting Process group ID of child = 14746
Child : Process group ID = 14746

Exec starting...
Hello there. I am in test.txt file.
Good day.

Exec is over...
Error in setting pgid of child process
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass8$
```