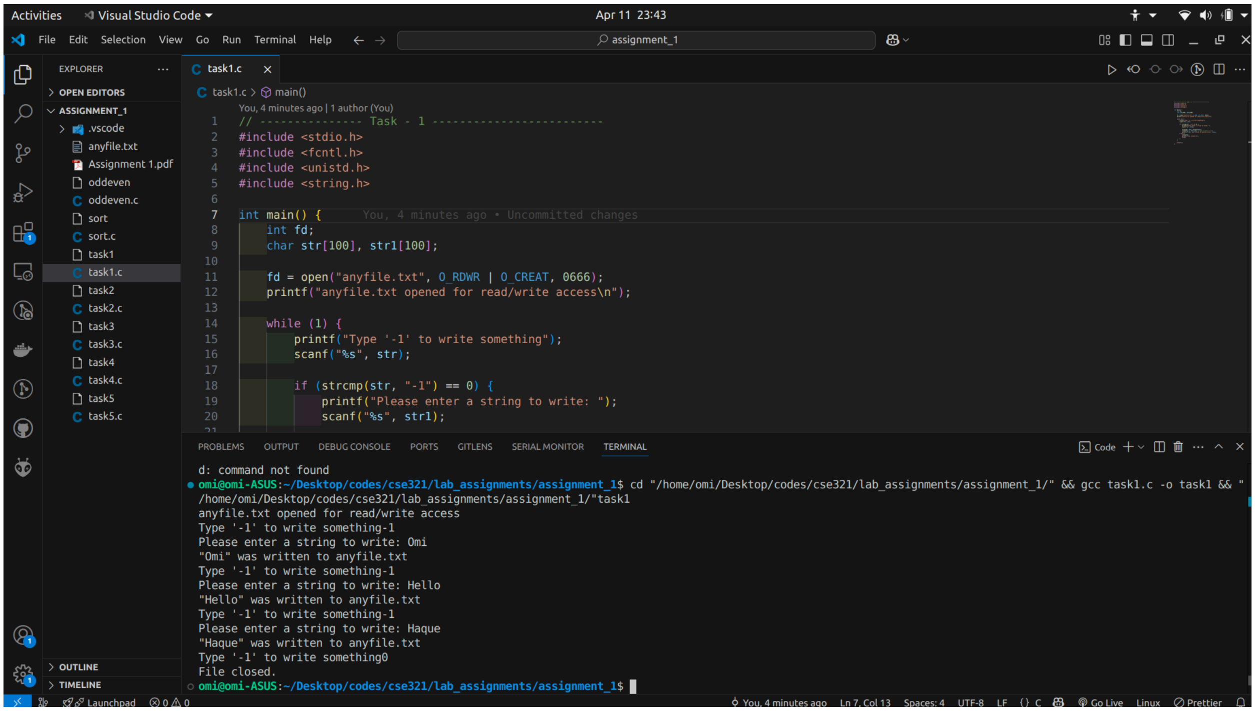
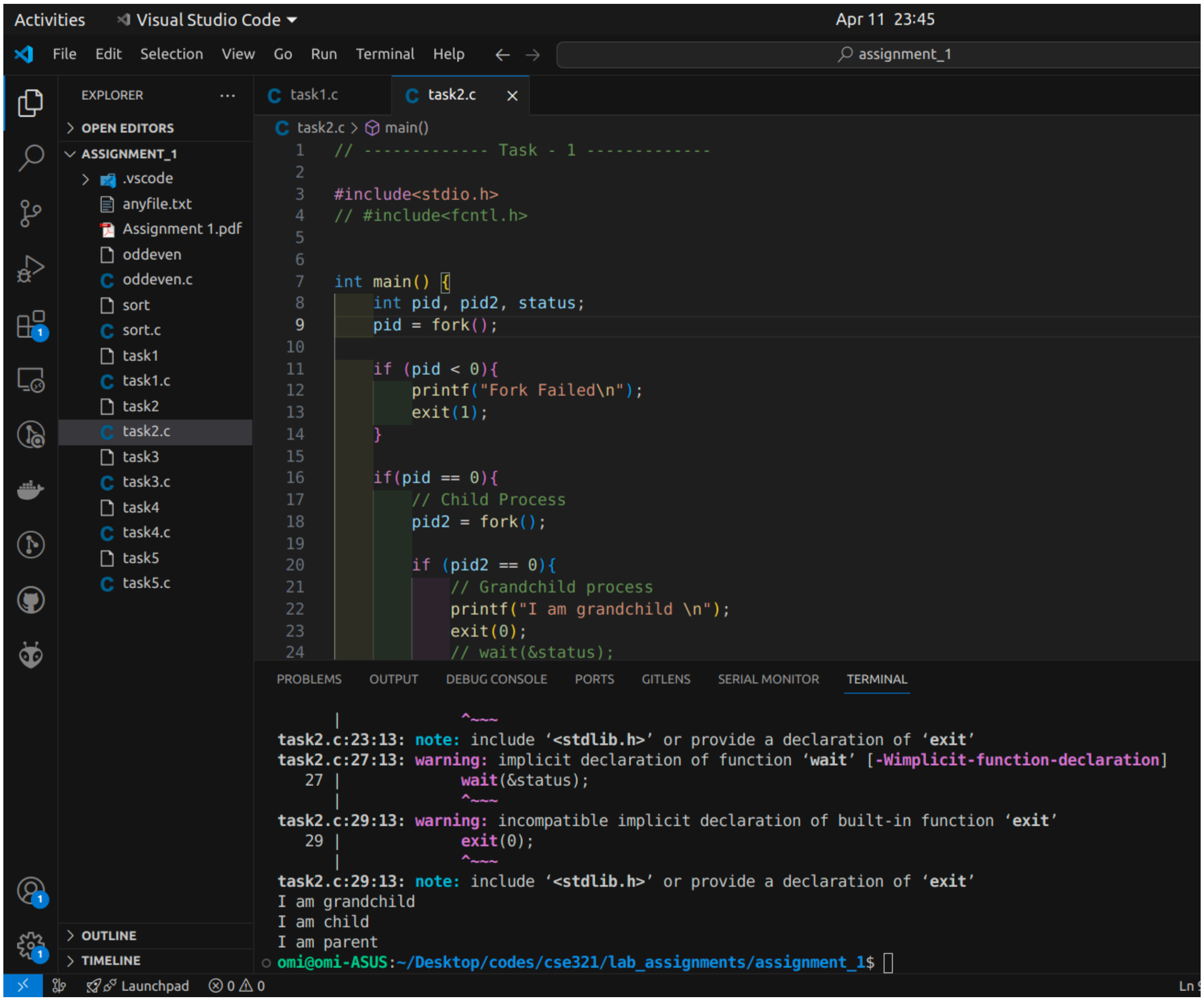
Task - 1



```c
// --------------- Task - 1 -------------------------
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>

int main() {      You, 4 minutes ago • Uncommitted changes
    int fd;
    char str[100], str1[100];

    fd = open("anyfile.txt", O_RDWR | O_CREAT, 0666);
    printf("anyfile.txt opened for read/write access\n");

    while (1) {
        printf("Type '-1' to write something");
        scanf("%s", str);

        if (strcmp(str, "-1") == 0) {
            printf("Please enter a string to write: ");
            scanf("%s", str1);
```

```
d: command not found
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$ cd "/home/omi/Desktop/codes/cse321/lab_assignments/assignment_1/" && gcc task1.c -o task1 && "
/home/omi/Desktop/codes/cse321/lab_assignments/assignment_1/"task1
anyfile.txt opened for read/write access
Type '-1' to write something-1
Please enter a string to write: Omi
"Omi" was written to anyfile.txt
Type '-1' to write something-1
Please enter a string to write: Hello
"Hello" was written to anyfile.txt
Type '-1' to write something-1
Please enter a string to write: Haque
"Haque" was written to anyfile.txt
Type '-1' to write something0
File closed.
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$
```

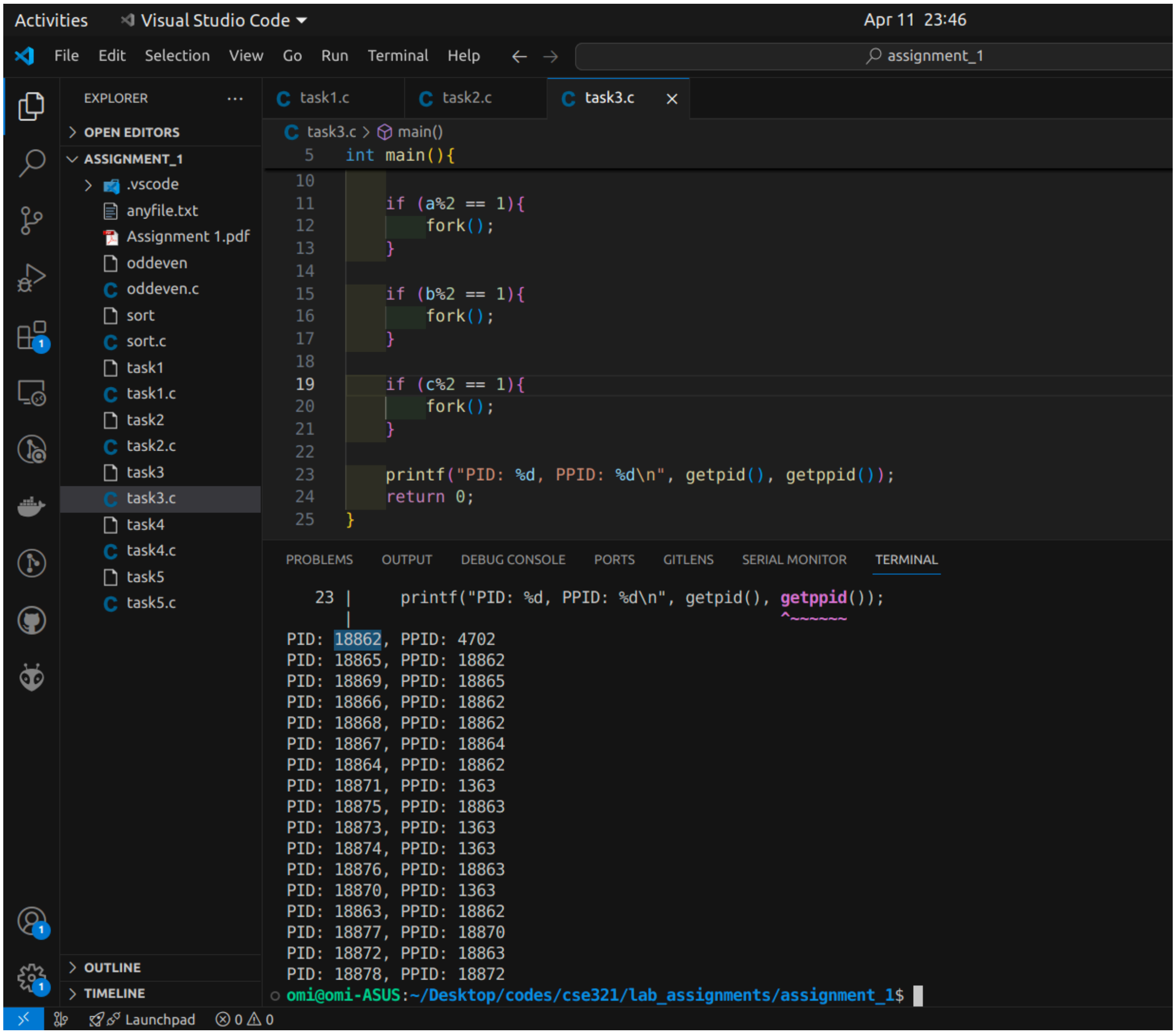
Task - 2



```c
// ------------ Task - 1 ------------

#include<stdio.h>
// #include<fcntl.h>


int main() {
    int pid, pid2, status;
    pid = fork();

    if (pid < 0){
        printf("Fork Failed\n");
        exit(1);
    }

    if(pid == 0){
        // Child Process
        pid2 = fork();

        if (pid2 == 0){
            // Grandchild process
            printf("I am grandchild \n");
            exit(0);
            // wait(&status);
```

```
      |              ^~~~
task2.c:23:13: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
task2.c:27:13: warning: implicit declaration of function ‘wait’ [-Wimplicit-function-declaration]
   27 |             wait(&status);
      |             ^~~~
task2.c:29:13: warning: incompatible implicit declaration of built-in function ‘exit’
   29 |             exit(0);
      |             ^~~~
task2.c:29:13: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
I am grandchild
I am child
I am parent
○ omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$ 
```

Task - 3

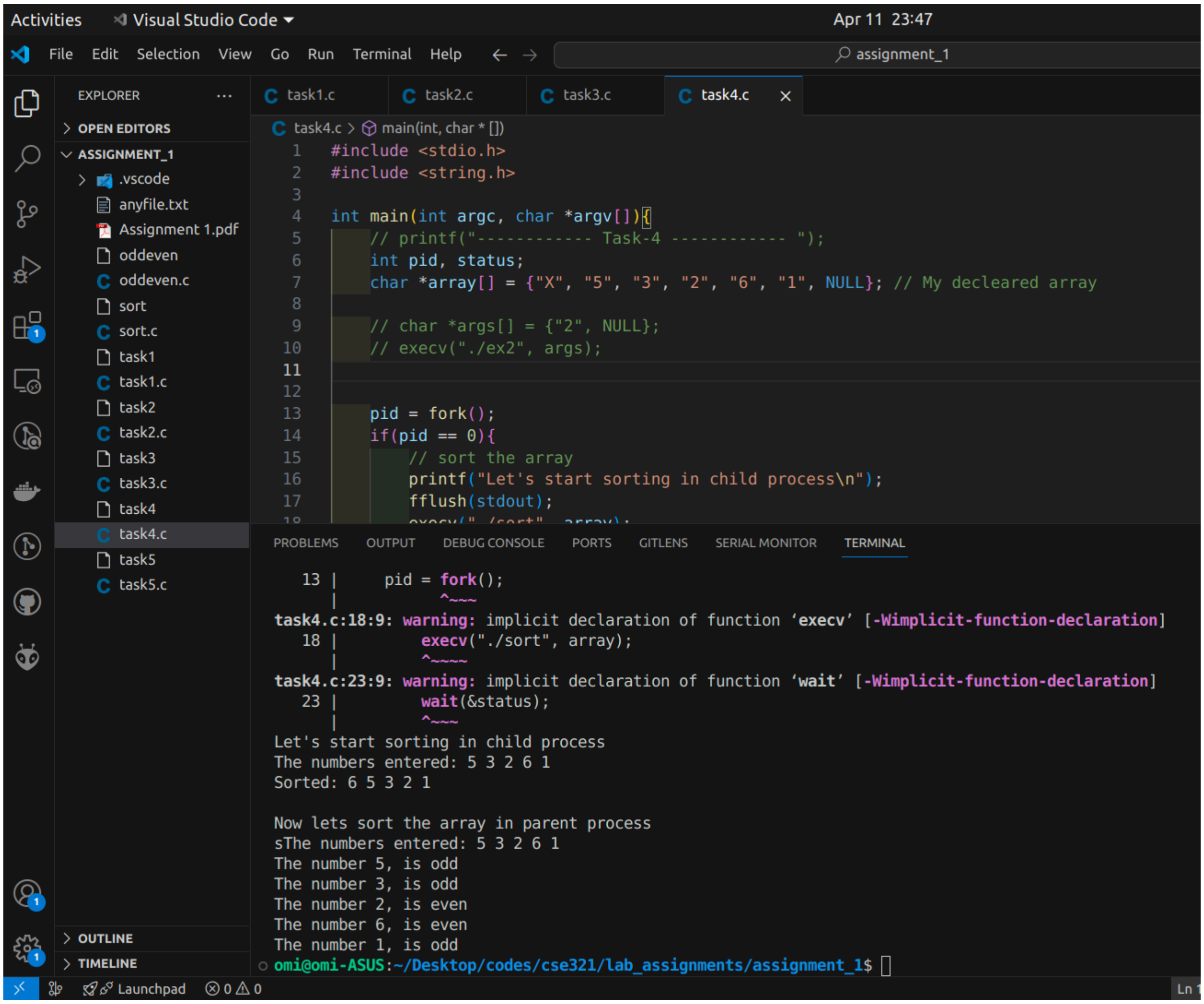


```
int main(){

    if (a%2 == 1){
        fork();
    }

    if (b%2 == 1){
        fork();
    }

    if (c%2 == 1){
        fork();
    }

    printf("PID: %d, PPID: %d\n", getpid(), getppid());
    return 0;
}
```

```
   23 |     printf("PID: %d, PPID: %d\n", getpid(), getppid());
      |                                             ^~~~~~~
PID: 18862, PPID: 4702
PID: 18865, PPID: 18862
PID: 18869, PPID: 18865
PID: 18866, PPID: 18862
PID: 18868, PPID: 18862
PID: 18867, PPID: 18864
PID: 18864, PPID: 18862
PID: 18871, PPID: 1363
PID: 18875, PPID: 18863
PID: 18873, PPID: 1363
PID: 18874, PPID: 1363
PID: 18876, PPID: 18863
PID: 18870, PPID: 1363
PID: 18863, PPID: 18862
PID: 18877, PPID: 18870
PID: 18872, PPID: 18863
PID: 18878, PPID: 18872
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$
```

More than 8 Process will be created, In My case 17 process here.

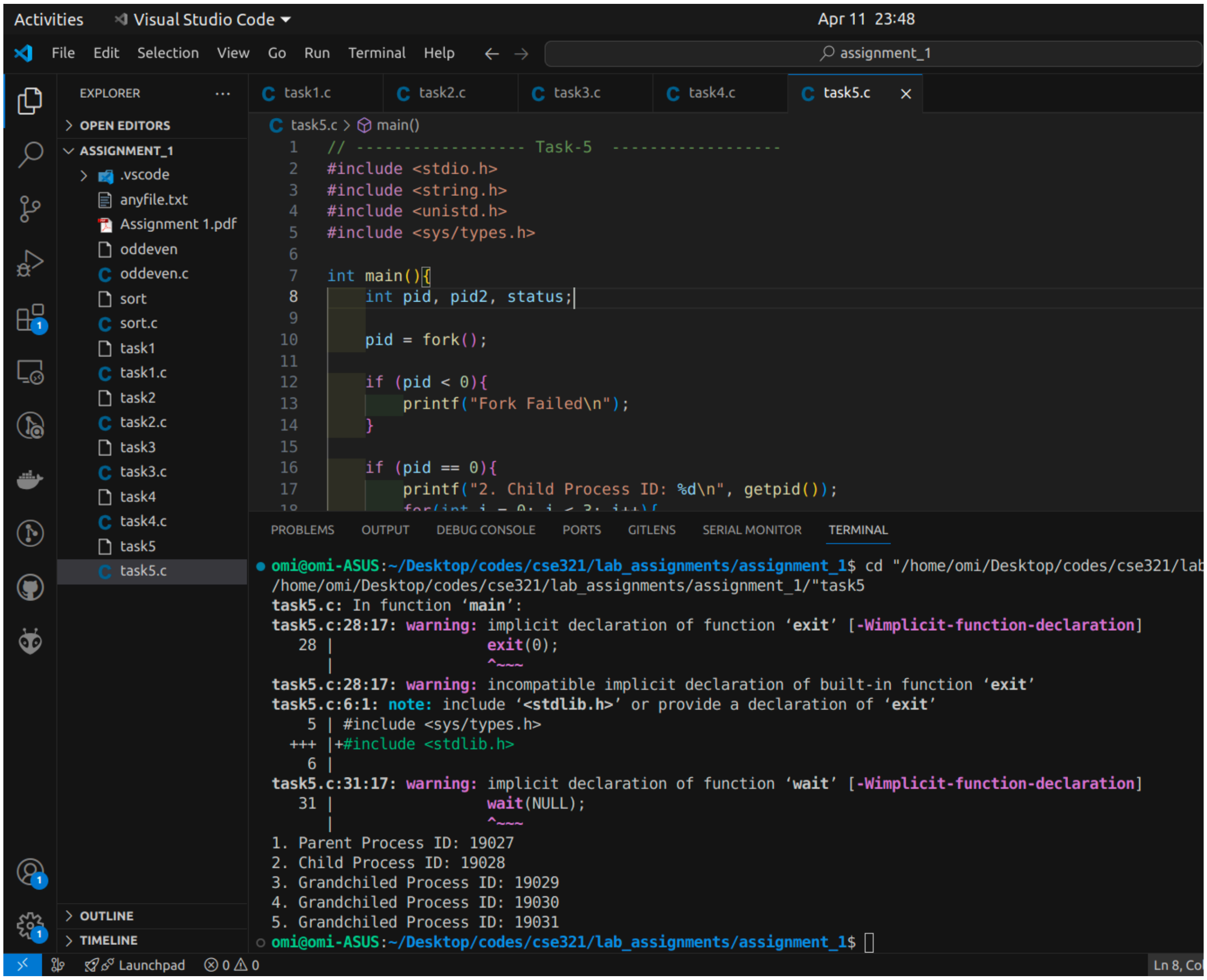
Task - 4



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

int main(int argc, char *argv[]){
    // printf("----------- Task-4 ----------- ");
    int pid, status;
    char *array[] = {"X", "5", "3", "2", "6", "1", NULL}; // My decleared array

    // char *args[] = {"2", NULL};
    // execv("./ex2", args);


    pid = fork();
    if(pid == 0){
        // sort the array
        printf("Let's start sorting in child process\n");
        fflush(stdout);
        execv("./sort", array);
```

```
   13 |     pid = fork();
      |           ^~~~
task4.c:18:9: warning: implicit declaration of function ‘execv’ [-Wimplicit-function-declaration]
   18 |         execv("./sort", array);
      |         ^~~~~
task4.c:23:9: warning: implicit declaration of function ‘wait’ [-Wimplicit-function-declaration]
   23 |         wait(&status);
      |         ^~~~
Let's start sorting in child process
The numbers entered: 5 3 2 6 1
Sorted: 6 5 3 2 1

Now lets sort the array in parent process
sThe numbers entered: 5 3 2 6 1
The number 5, is odd
The number 3, is odd
The number 2, is even
The number 6, is even
The number 1, is odd
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$ 
```

Task - 5



```c
// ----------------- Task-5  -----------------
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>

int main(){
    int pid, pid2, status;

    pid = fork();

    if (pid < 0){
        printf("Fork Failed\n");
    }

    if (pid == 0){
        printf("2. Child Process ID: %d\n", getpid());
        for(int i = 0; i < 3; i++){
```

```
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$ cd "/home/omi/Desktop/codes/cse321/lab
/home/omi/Desktop/codes/cse321/lab_assignments/assignment_1/"task5
task5.c: In function ‘main’:
task5.c:28:17: warning: implicit declaration of function ‘exit’ [-Wimplicit-function-declaration]
   28 |                 exit(0);
      |                 ^~~~
task5.c:28:17: warning: incompatible implicit declaration of built-in function ‘exit’
task5.c:6:1: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
    5 | #include <sys/types.h>
  +++ |+#include <stdlib.h>
    6 |
task5.c:31:17: warning: implicit declaration of function ‘wait’ [-Wimplicit-function-declaration]
   31 |                 wait(NULL);
      |                 ^~~~
1. Parent Process ID: 19027
2. Child Process ID: 19028
3. Grandchiled Process ID: 19029
4. Grandchiled Process ID: 19030
5. Grandchiled Process ID: 19031
omi@omi-ASUS:~/Desktop/codes/cse321/lab_assignments/assignment_1$
```