Experiment – 1

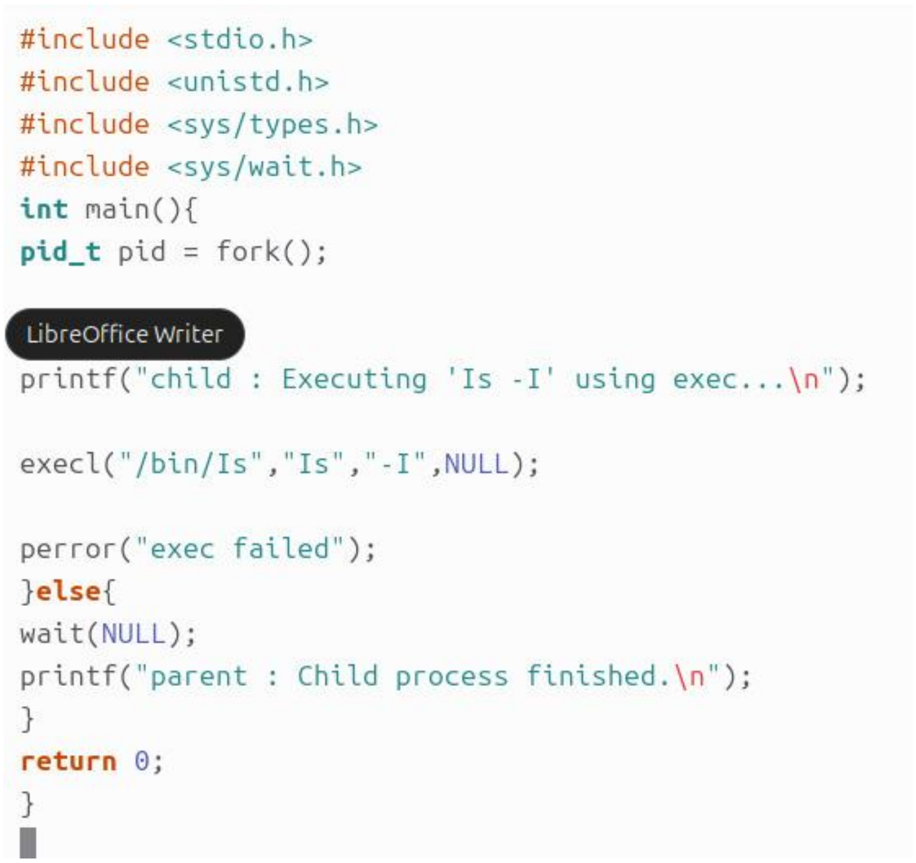
Name : Venkata Sai Shreya Maturi

Reg No. : 23BCE8955

1. exec() system call

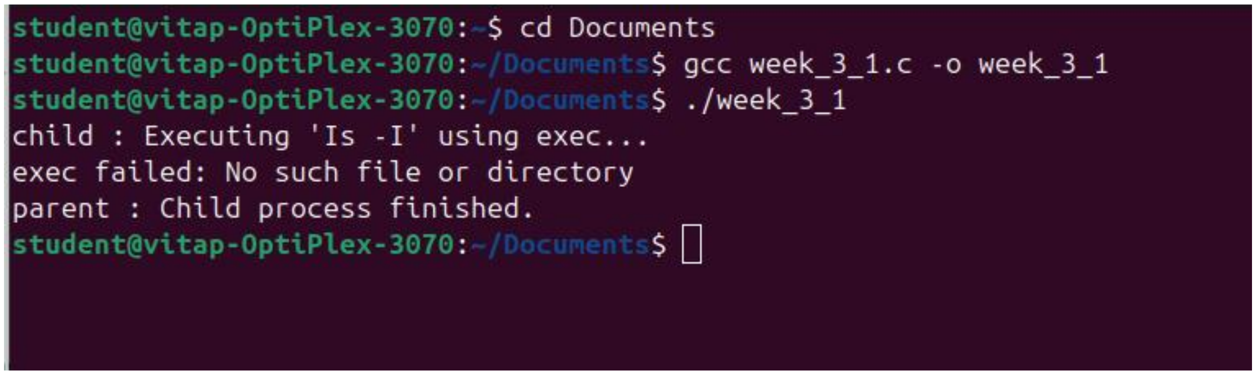
- The exec() family of functions is used in UNIX/Linux to **replace the current process with a new process.**
- exec() **does not create a new process** — it replaces the current one.
- It is often used **after fork()**, where the child process uses exec() to run a different program.

Input :



```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
int main(){
pid_t pid = fork();

printf("child : Executing 'Is -I' using exec...\n");

execl("/bin/Is","Is","-I",NULL);

perror("exec failed");
}else{
wait(NULL);
printf("parent : Child process finished.\n");
}
return 0;
}
```

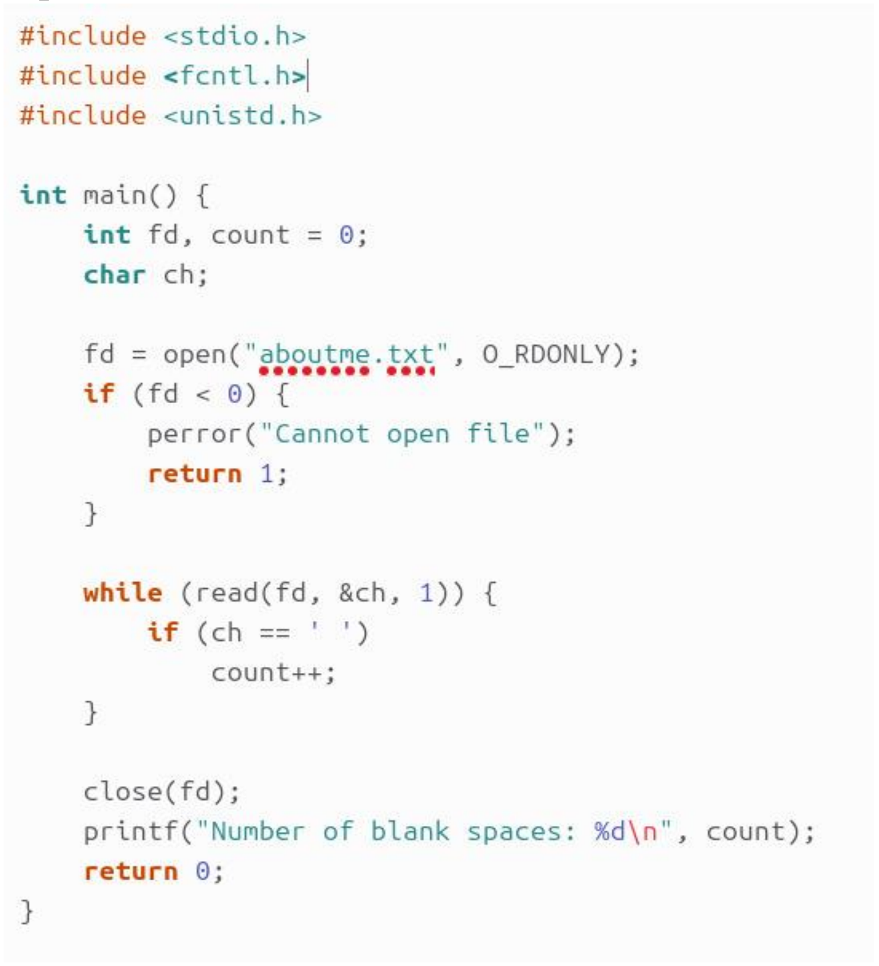
Output :



```
student@vitap-OptiPlex-3070:~$ cd Documents
student@vitap-OptiPlex-3070:~/Documents$ gcc week_3_1.c -o week_3_1
student@vitap-OptiPlex-3070:~/Documents$ ./week_3_1
child : Executing 'Is -I' using exec...
exec failed: No such file or directory
parent : Child process finished.
student@vitap-OptiPlex-3070:~/Documents$ 
```

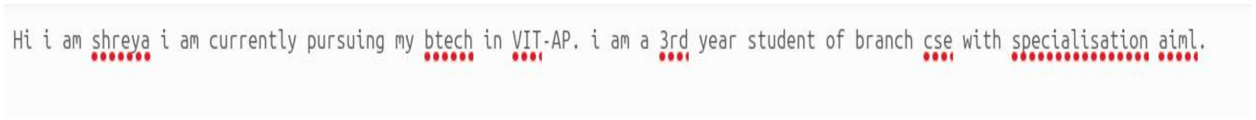
2. **Write a program using UNIX system calls to count the number of blank spaces in a text file.**

Input :

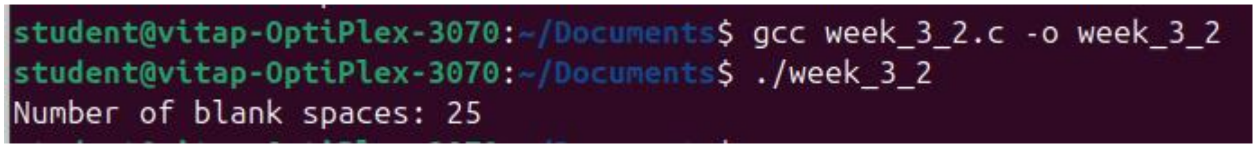


```c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    int fd, count = 0;
    char ch;

    fd = open("aboutme.txt", O_RDONLY);
    if (fd < 0) {
        perror("Cannot open file");
        return 1;
    }

    while (read(fd, &ch, 1)) {
        if (ch == ' ')
            count++;
    }

    close(fd);
    printf("Number of blank spaces: %d\n", count);
    return 0;
}
```

Source file :

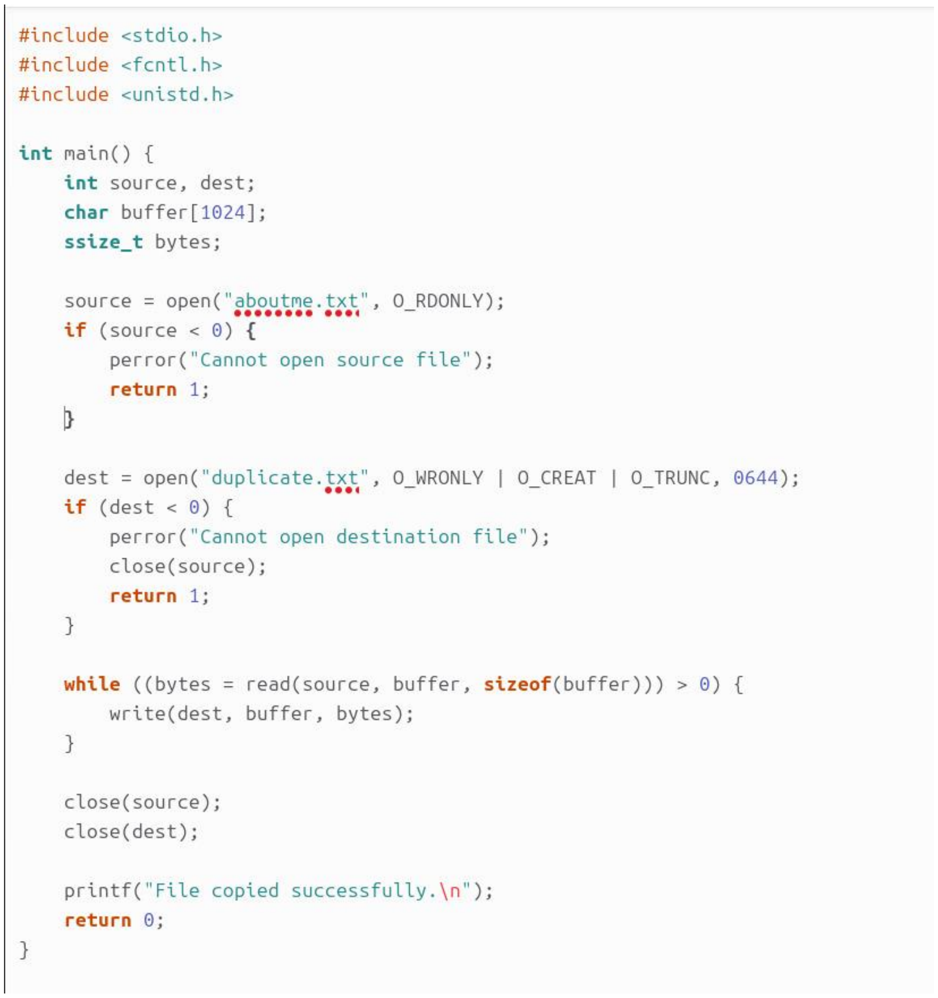

```
Hi i am shreya i am currently pursuing my btech in VIT-AP. i am a 3rd year student of branch cse with specialisation aiml.
```

Output :

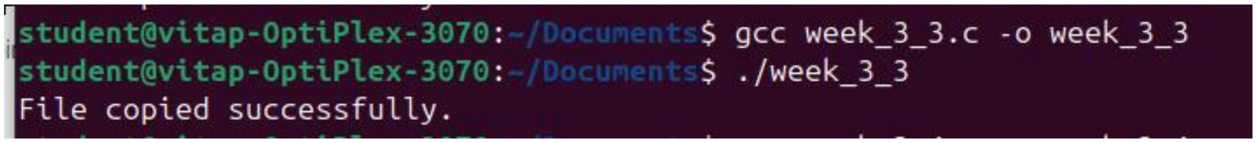

```
student@vitap-OptiPlex-3070:~/Documents$ gcc week_3_2.c -o week_3_2
student@vitap-OptiPlex-3070:~/Documents$ ./week_3_2
Number of blank spaces: 25
```

3. Write a program using UNIX system calls to copy the contents of one file into another file.

Input :

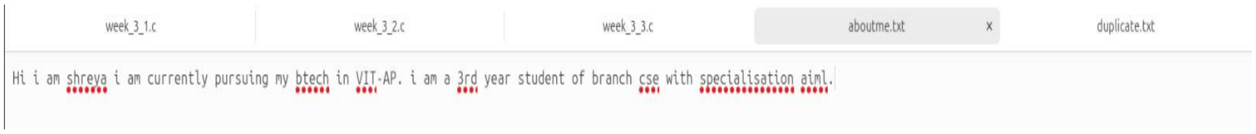

```c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    int source, dest;
    char buffer[1024];
    ssize_t bytes;

    source = open("aboutme.txt", O_RDONLY);
    if (source < 0) {
        perror("Cannot open source file");
        return 1;
    }

    dest = open("duplicate.txt", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    if (dest < 0) {
        perror("Cannot open destination file");
        close(source);
        return 1;
    }

    while ((bytes = read(source, buffer, sizeof(buffer))) > 0) {
        write(dest, buffer, bytes);
    }

    close(source);
    close(dest);

    printf("File copied successfully.\n");
    return 0;
}
```

Output :



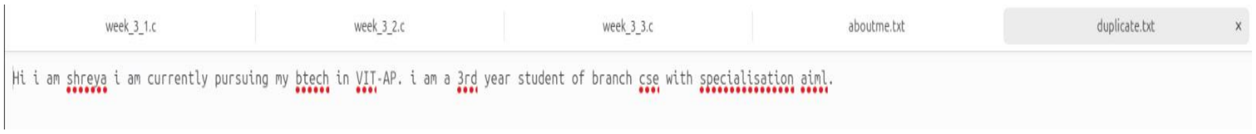
```
student@vitap-OptiPlex-3070:~/Documents$ gcc week_3_3.c -o week_3_3
student@vitap-OptiPlex-3070:~/Documents$ ./week_3_3
File copied successfully.
```

Source file :



```
Hi i am shreya i am currently pursuing my btech in VIT-AP. i am a 3rd year student of branch cse with specialisation aiml.
```

Destination file :



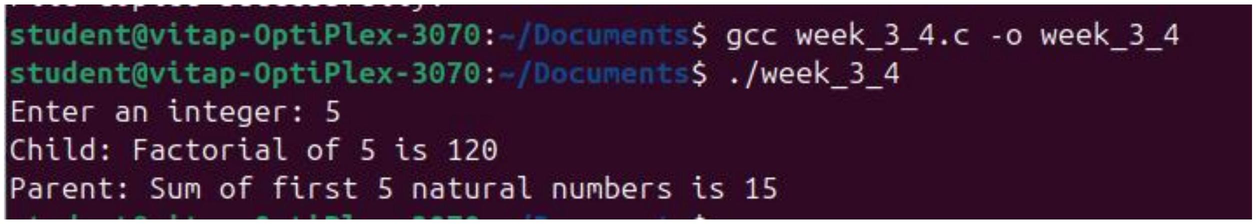
```
Hi i am shreya i am currently pursuing my btech in VIT-AP. i am a 3rd year student of branch cse with specialisation aiml.
```

4. **Write a C program to demonstrate process creation using fork().**
   - The program should read an integer n from the user.
   - The **child process** should compute and display the **factorial of n**.
   - The **parent process** should wait for the child process to finish using wait() and then compute and display the **sum of first n natural numbers**.

Input :

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

int factorial(int n) {
    if (n == 0) return 1;
    return n * factorial(n - 1);
}
int sum_n(int n) {
    return (n * (n + 1)) / 2;
}
int main() {
    int n;
    printf("Enter an integer: ");
    scanf("%d", &n);

    pid_t pid = fork();

    if (pid == 0) {
        // Child process
        int fact = factorial(n);
        printf("Child: Factorial of %d is %d\n", n, fact);
    } else if (pid > 0) {
        // Parent process
        wait(NULL);  // Wait for child to finish
        int sum = sum_n(n);
        printf("Parent: Sum of first %d natural numbers is %d\n", n, sum);
    } else {
        perror("Fork failed");
        return 1;
    }
    return 0;
}
```

Output :



```
student@vitap-OptiPlex-3070:~/Documents$ gcc week_3_4.c -o week_3_4
student@vitap-OptiPlex-3070:~/Documents$ ./week_3_4
Enter an integer: 5
Child: Factorial of 5 is 120
Parent: Sum of first 5 natural numbers is 15
```