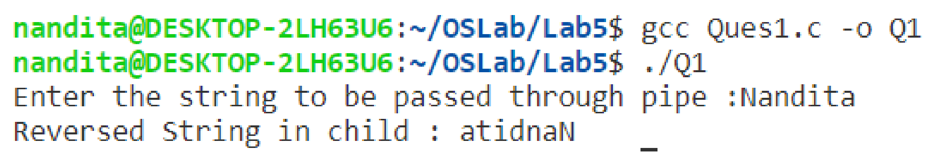
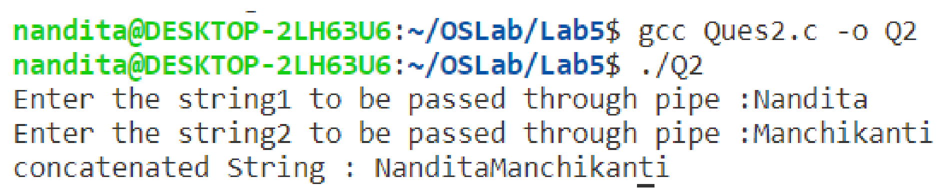
CED19I056

Nandita M

Ques1 Output

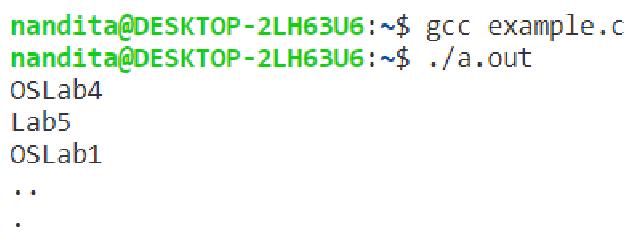

```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques1.c -o Q1
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./Q1
Enter the string to be passed through pipe :Nandita
Reversed String in child : atidnaN
```

Ques2 Output

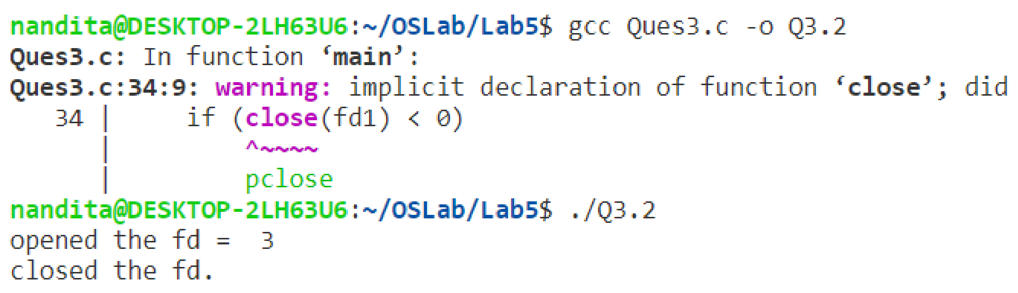

```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques2.c -o Q2
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./Q2
Enter the string1 to be passed through pipe :Nandita
Enter the string2 to be passed through pipe :Manchikanti
concatenated String : NanditaManchikanti
```

Ques3 Output

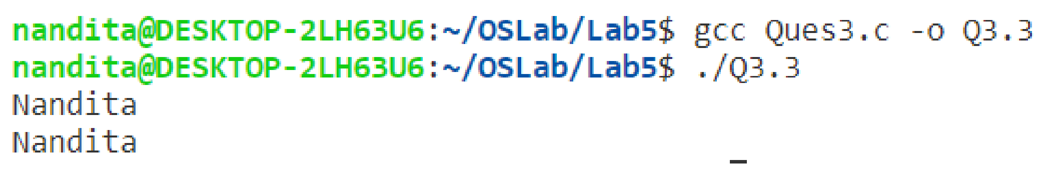
i)


```
nandita@DESKTOP-2LH63U6:~$ gcc example.c
nandita@DESKTOP-2LH63U6:~$ ./a.out
OSLab4
Lab5
OSLab1
..
.
```

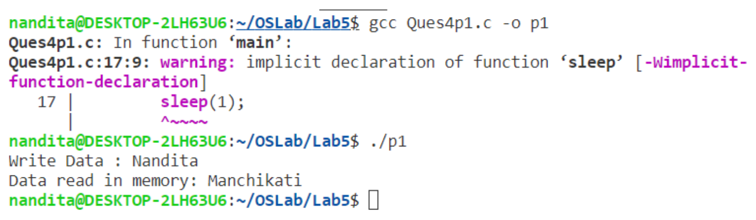
ii)

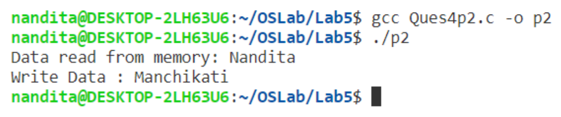

```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques3.c -o Q3.2
Ques3.c: In function ‘main’:
Ques3.c:34:9: warning: implicit declaration of function ‘close’; did
   34 |     if (close(fd1) < 0)
      |         ^~~~~
      |         pclose
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./Q3.2
opened the fd =  3
closed the fd.
```

iii)


```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques3.c -o Q3.3
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./Q3.3
Nandita
Nandita
```

Ques 4 Output


```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques4p1.c -o p1
Ques4p1.c: In function ‘main’:
Ques4p1.c:17:9: warning: implicit declaration of function ‘sleep’ [-Wimplicit-
function-declaration]
   17 |         sleep(1);
      |         ^~~~~
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./p1
Write Data : Nandita
Data read in memory: Manchikati
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$
```


```
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ gcc Ques4p2.c -o p2
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$ ./p2
Data read from memory: Nandita
Write Data : Manchikati
nandita@DESKTOP-2LH63U6:~/OSLab/Lab5$
```

1. First take the input string using scanf. Create 2 pipes, namely pipe1 and pipe2. Create a child process using fork. Write the string to the child in pipe1. Read the string sent by parent to the child in pipe1. Reverse the string in child and send to the parent through pipe2. Read and display the data sent by child.

2. Same as the above question but the only difference is, we take the 2nd string input in child process, concatenate in the child and send it to parent. The parent displays it.

3. i) opendir("~~filename~~ directory name");
   ↳ returns a pointer pointing to the directory of type DIR.
   → If directory is not present, returns a null.

   ~~readdir("directory name")~~.
   readdir(DIR pointer).
      ↳ Returns a pointer to an object of type struct dirent..
      → If error returns NULL.

   ii) Open()
      ↳ used to open the file for reading, writing or both
      eg: open ("filename", O_RDONLY)
         ↳ only Reading
      eg: open ("file name", O_WRONLY)
         ↳ only writing.

   returns a file descriptor number (usually 3).

close().
Tells the OS that you are done with a fd and Close the file pointed by the fd.

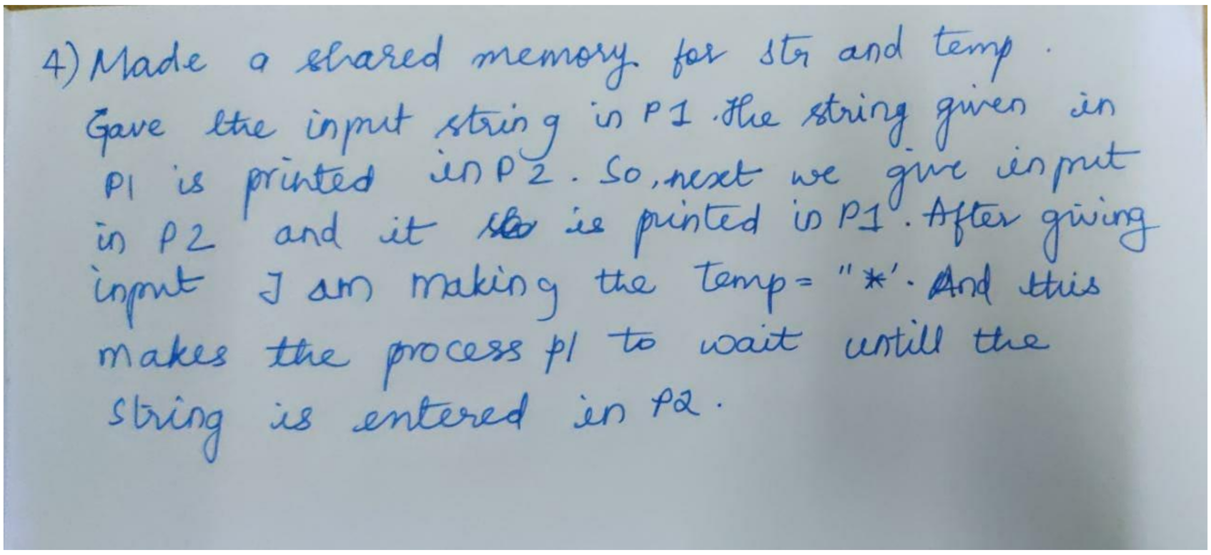
int close (int fd);

iii) read()
read() sys call is used to read from a file descriptor.

0 → fd for standard input device
1 → standard ouput device.
2 → standard error device

read (0, void* buf, size_t count)
eg: read (0, buff, 10).
read from standard input to buffer ~~of~~ of size 10 bytes.

write().
write some data to any file or fd.
eg: write (1, buff, 10)
↳ fd of where to write
from where should ~~of~~ it write (the place of data)
How many bytes of data it should write.



4) Made a shared memory for str and temp.
Gave the input string in P1. The string given in
P1 is printed in P2. So, next we give input
in P2 and it is printed in P1. After giving
input I am making the temp = "*'. And this
makes the process p1 to wait untill the
string is entered in P2.