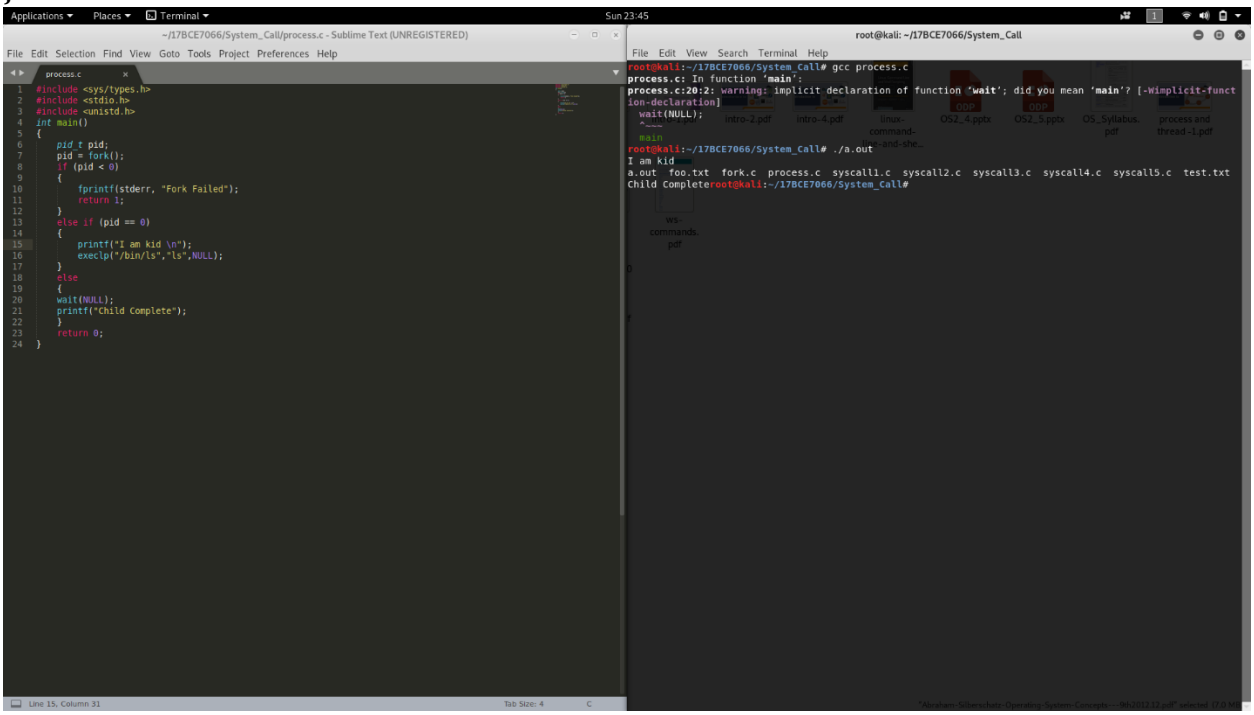
Inter-Process Communication (IPC)

Aditya Jain(17BCE7066)

Program 1: To create a child process and execute some task in it

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int main()
{
        pid_t pid;
        pid = fork();
        if (pid < 0)
        {
                fprintf(stderr, "Fork Failed");
                return 1;
        }
        else if (pid == 0)
        {
                printf("I am kid \n");
                execlp("/bin/ls","ls",NULL);
        }
        else
        {
        wait(NULL);
        printf("Child Complete");
        }
        return 0;
}
```



Program 2: To use shared-memory for IPC

Producer:

```
#include<stdio.h>
#include<stdlib.h>
#include <string.h>
#include<fcntl.h>
#include<sys/shm.h>
#include<sys/mman.h>
#include<sys/stat.h>

int main()
{
	const int SIZE = 4096;
	const char *name = "Adi";
	const char *message1 = "Hello \n";
	const char *message2 = "My name is Aditya Jain \n";
	int shm_fd;
	void *ptr;

	shm_fd = open(name, O_CREAT | O_RDWR, 0666);
	ftruncate(shm_fd, SIZE);
	ptr = mmap(NULL, SIZE, PROT_WRITE, MAP_SHARED, shm_fd, 0);
	sprintf(ptr, "%s", message1);
	ptr+=strlen(message1);
	sprintf(ptr,"%s",message2);
	ptr+=strlen(message2);
	return 0;
}
```

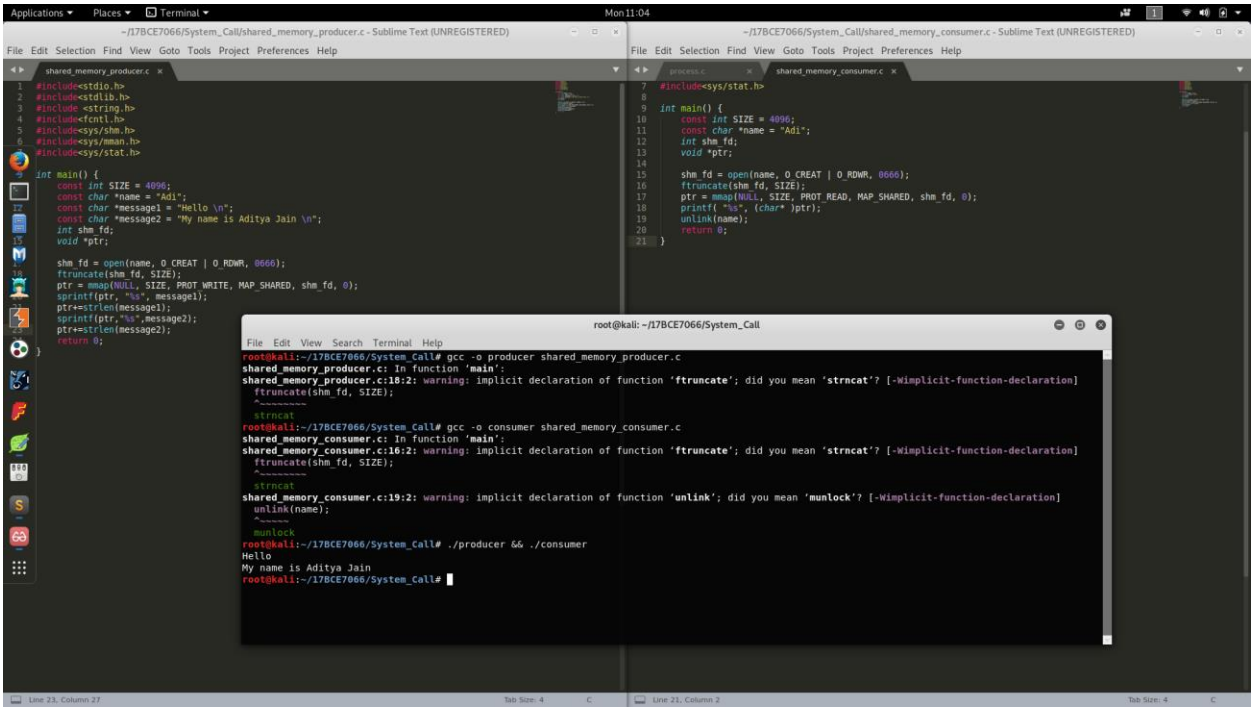
Consumer:

```
#include<stdio.h>
#include<stdlib.h>
#include <string.h>
#include<fcntl.h>
#include<sys/shm.h>
#include<sys/mman.h>
#include<sys/stat.h>

int main()
{
	const int SIZE = 4096;
	const char *name = "Adi";
	int shm_fd;
	void *ptr;

	shm_fd = open(name, O_CREAT | O_RDWR, 0666);
	ftruncate(shm_fd, SIZE);
	ptr = mmap(NULL, SIZE, PROT_READ, MAP_SHARED, shm_fd, 0);
```

```
        printf( "%s", (char* )ptr);
        unlink(name);
        return 0;
}
```



Program 3: To send a message using one pipe from parent to child and the child changes the case of every character and sends it back to the parent.

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

int main()
{
   int fd1[2];
   int fd2[2];

   char input_str[100];
   pid_t p;

   if (pipe(fd1)==-1)
   {
      fprintf(stderr, "Pipe Failed" );
      return 1;
   }
   if (pipe(fd2)==-1)
   {
      fprintf(stderr, "Pipe Failed" );
      return 1;
   }
   printf("Enter the message :");
   scanf("%s", input_str);
   p = fork();

   if (p < 0)
   {
      fprintf(stderr, "fork Failed" );
      return 1;
   }

   else if (p > 0)
   {
      char new_str[100];
      close(fd1[0]);
      write(fd1[1], input_str, strlen(input_str)+1);
      close(fd1[1]);
      wait(NULL);
      close(fd2[1]);
      read(fd2[0], new_str, 100);
      printf("Changed string: %s\n", new_str);
```

```
        close(fd2[0]);
    }
    else
    {
        close(fd1[1]);
        char original_str[100];
        read(fd1[0], original_str, 100);
        int i;
        for (i=0; i<strlen(original_str); i++)
        {
            int x=(int)(original_str[i]);
            if(x>=65&&x<=90)
            {
                original_str[i]=(char)(x+32);
            }
            if(x>=97&&x<=122)
            {
                original_str[i]=(char)(x-32);
            }
        }
        original_str[i]='\0';
        close(fd1[0]);
        close(fd2[0]);
        write(fd2[1], original_str, strlen(original_str)+1);
        close(fd2[1]);
        exit(0);
    }
}
```