Assignment 4

IPC through shared memory

SYSTEM CALLS USED ARE:

ftok()*: is use to generate a unique key.*

shmget()*: int shmget(key_t,size_tsize,intshmflg);*

shmat()*: Before you can use a shared memory segment, you have to attach yourself*

*to it using shmat(). void *shmat(int shmid ,void *shmaddr ,int shmflg);*

shmid is shared memory id. shmaddr specifies specific address to use but we should set

it to zero and OS will automatically choose the address.

shmdt()*: When you're done with the shared memory segment, your program should*

*detach itself from it using shmdt(). int shmdt(void *shmaddr);*

shmctl()*: when you detach from shared memory,it is not destroyed. So, to destroy*

shmctl() is used. shmctl(int shmid,IPC_RMID,NULL);

SHARED MEMORY FOR WRITER PROCESS

```
#include <iostream>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
using namespace std;

int main()
{
        // ftok to generate unique key
        key_t key = ftok("shmfile",65);

        // shmget returns an identifier in shmid
        int shmid = shmget(key,1024,0666|IPC_CREAT);

        // shmat to attach to shared memory
        char *str = (char*) shmat(shmid,(void*)0,0);

        cout<<"Write Data : ";
        gets(str);

        printf("Data written in memory: %s\n",str);

        //detach from shared memory
        shmdt(str);

        return 0;
}
```

SHARED MEMORY FOR READER PROCESS

```
#include <iostream>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
using namespace std;

int main()
{
    // ftok to generate unique key
    key_t key = ftok("shmfile",65);

    // shmget returns an identifier in shmid
    int shmid = shmget(key,1024,0666|IPC_CREAT);

    // shmat to attach to shared memory
    char *str = (char*) shmat(shmid,(void*)0,0);

    printf("Data read from memory: %s\n",str);

    //detach from shared memory
    shmdt(str);

    // destroy the shared memory
    shmctl(shmid,IPC_RMID,NULL);

    return 0;
}
```

Exercise :

Q1: Implement a code for the following task:

1.Create a shared memory segment *of 2048 bytes* and print its key.

2.writes some content into the shared memory segment.

3.reads the content of the shared memory and also prints the id of the shared memory segment you are accessing.

Q2. Show what shmget() function returns on success implement that in error handling?

Q3. Show what does shmat() function returns on success implement that in error handling?

Q4. Try segment sizes(100,1000) and observe the changes. The value of segment size in shmget() is a round-up value to a multiple of ___.

Q5. Create two shared segment in a process. Will there returned identifiers be same or different?

Q6. Which function can be used to detach the shared segment from the address space of the process?