Practical - 5

Aim : Write the following programs using inter process communication – shared memory.
The program ‘writer.c’ will print 1 to 100 in shared memory region. Another program ‘reader.c’ that will read all the numbers from shared memory to make addition of it and display it.

writer.c :

```
#include<bits/stdc++.h>
#include<sys/types.h>
#include<stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <time.h>
#include<sys/ipc.h>
#include<sys/shm.h>

int main(){
	// key_t somekey;
	// somekey = 12345;
	key_t k = ftok("./",'x');
	// printf("%d\n",somekey);
	printf("generated key : %d\n",k);
	int shm_id = shmget(k,100*sizeof(int),IPC_CREAT | 0666);
	if(shm_id == -1){
		printf("Error in shmget!!!\n");
	}
	printf("shared area key : %d\n",shm_id);
	int *shm_ptr = (int*)shmat(shm_id, NULL, 0);
	if(shm_ptr == (int*) -1){
		printf("Error in shmat!!!\n");
	}
	int *s = shm_ptr;
	for(int i = 1 ; i <= 100 ;i++){
		*s++ = i;
	}
	while(*shm_ptr != '*'){
		sleep(1);
	}
	printf("End!!!\n");
	shmdt(shm_ptr);
	return 0;
}
```

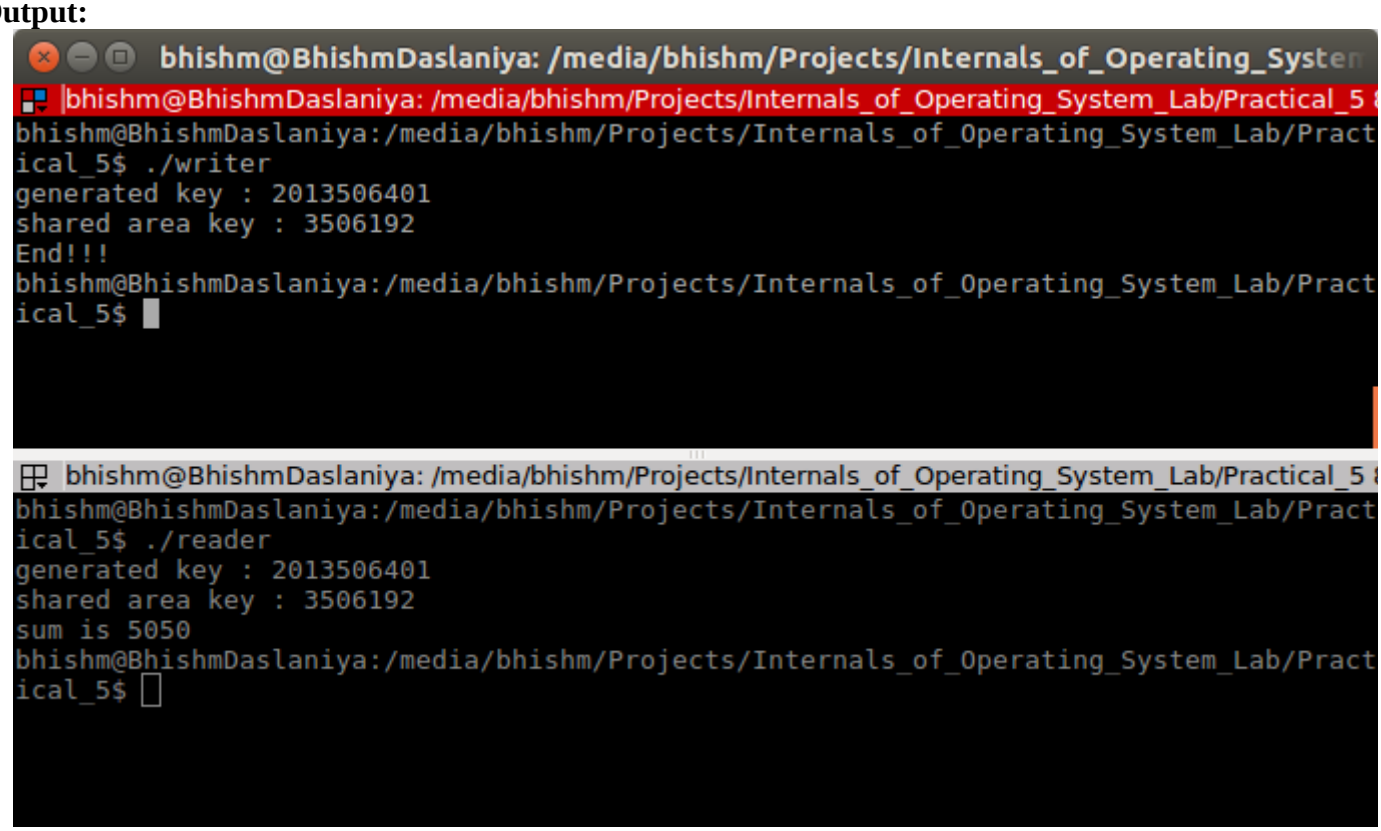
reader.c :

```
#include<bits/stdc++.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/ipc.h>
#include<sys/shm.h>

int main(){
	// key_t somekey;
```

```
        // somekey = 12345;
        key_t k = ftok("./",'x');
        // printf("%d\n",somekey);
        printf("generated key : %d\n",k);
        int shm_id = shmget(k,100*sizeof(int),IPC_CREAT | 0666);
        if(shm_id == -1){
                printf("Error in shmget!!!\n");
        }
        printf("shared area key : %d\n",shm_id);
        int *shm_ptr = (int*)shmat(shm_id, NULL, 0);
        if(shm_ptr == (int*) -1){
                printf("Error in shmat!!!\n");
        }
        int a[100];
        int sum = 0;
        int *s;
        s= shm_ptr;
        for(int i = 0 ; i < 100 ;i++){
                a[i] = *s++;
                sum+=a[i];
        }
        printf("sum is %d \n",sum);
        *shm_ptr = '*';
        shmdt(shm_ptr);
        shmctl(shm_id,IPC_RMID,NULL);
        return 0;
}
```

Output:



Assignment: Solve above issue using pipe().

Code:

```
#include<bits/stdc++.h>
#include<unistd.h>
using namespace std;

int main() {
  int pipefds[2],status,pid,sum = 0;
  int count=101,writeArray[101] = {0},readArray[101] = {0};
  for(int i=1;i<=100;i++){
    writeArray[i]=i;
  }
  status = pipe(pipefds);
  if (status == -1) {
    perror("Error in creating pipe\n");
    return 1;
  }
  pid = fork();

  // Child process
  if (pid != 0) {
    printf("Parent Process - Writing to pipe - Message \n");
    write(pipefds[1], writeArray, (count*sizeof(int)));

  } else { //Parent process

    read(pipefds[0], readArray, (count*sizeof(int)));
    printf("Child Process - Reading from pipe - Message\n");
    for(int i=1;i<=100;i++){
      sum+=readArray[i];
    }
  printf("Sum is : %d\n",sum);
  }
  return 0;
}
```

Output:

```
bhishm@BhishmDaslaniya: /media/bhishm/Projects/Internals_of_Operating_System_Lab/Practi
bhishm@BhishmDaslaniya:/media/bhishm/Projects/Internals_of_Operating_System_Lab/
Practical_5$ ./a.out
Parent Process - Writing to pipe - Message
Child Process - Reading from pipe - Message
Sum is : 5050
```

Conclusion: From this practical I have learnt about Interporcess communication and also learnt about how to use shared resources in program.