Ganpat University | Institute of Computer Technology

॥ विद्यया समाजोत्कर्षः ॥

Name: Tushar Panchal

En.No: 21162101014

Sub: OS(Operating Systems)

Branch: CBA

Batch:41

---------------------------PRACTICAL 05----------------------------

❖ ***Experiment-5 :***

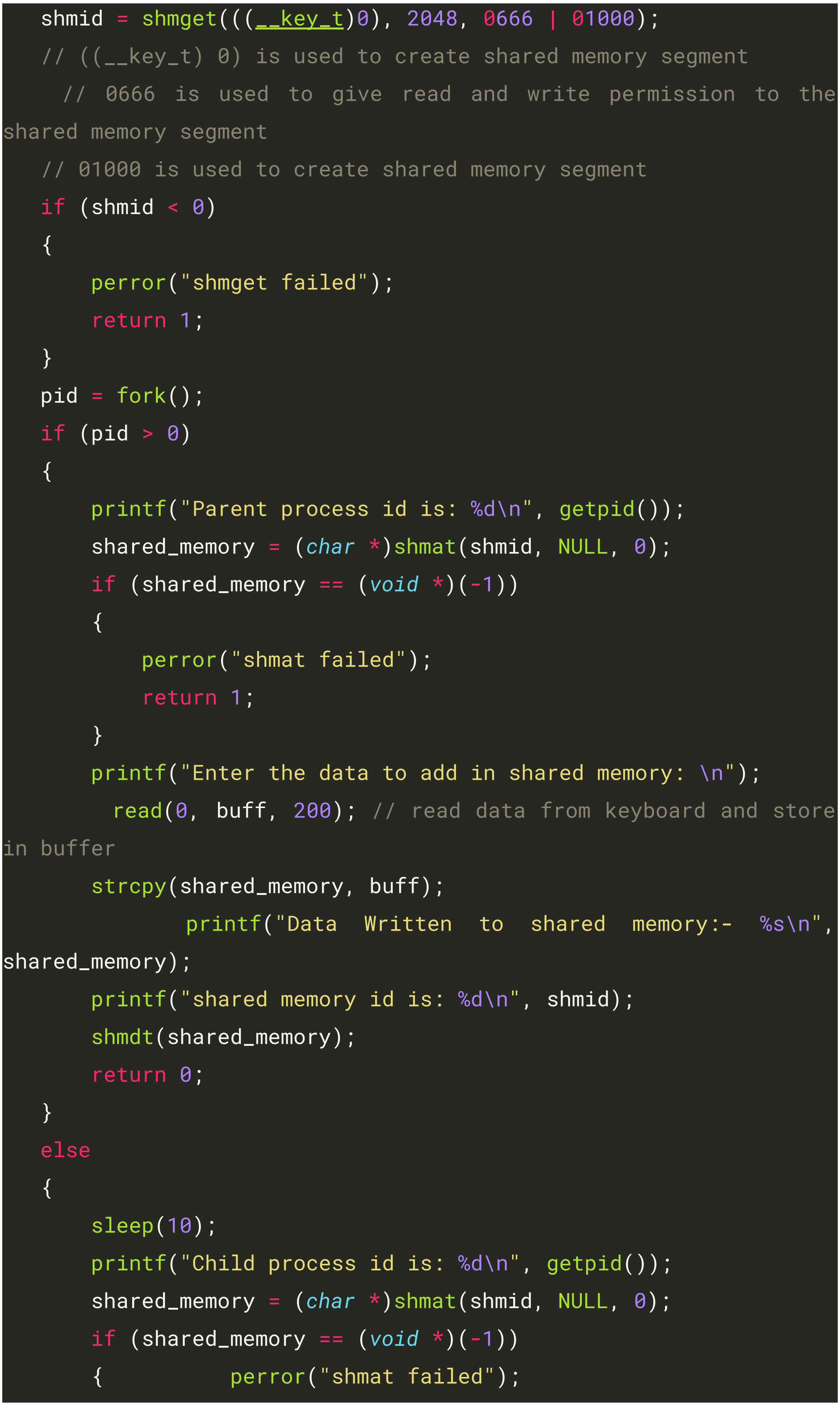
IPC through shared memory.

➢ **Question 1 :**

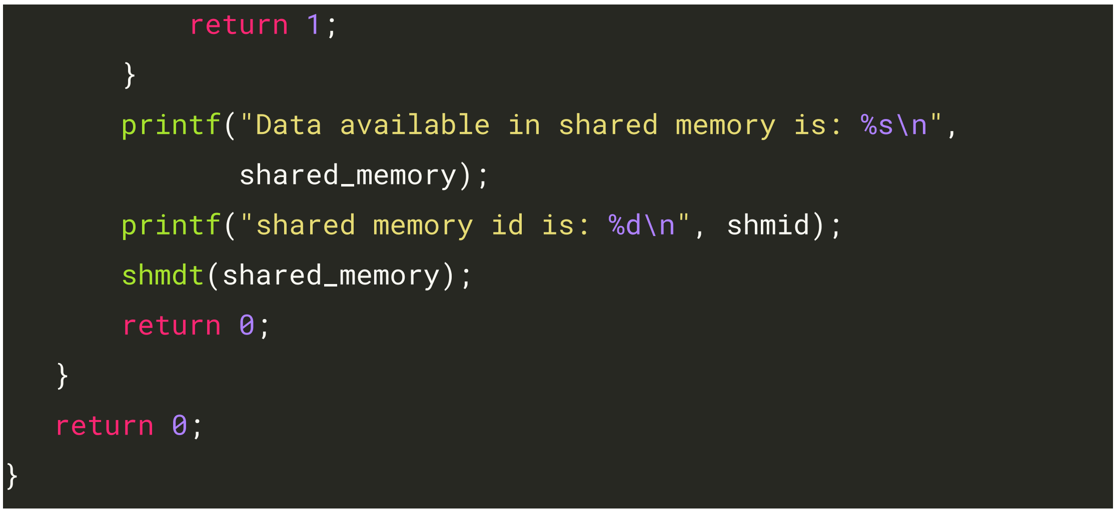
1. Design a program to create a shared memory segment of 2048 bytes and write some content into it. Then create a child process which then reads the content written by the parent process in the shared memory segment.

✔ **Source Code :**

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#define shmget(a, b, c) syscall(29, a, b, c) // 29 is the number of shmget system call
#define shmat(a, b, c) syscall(30, a, b, c)  // 30 is the number of shmat system call
#define shmdt(a) syscall(67, a)              // 67 is the number of shmdt system call
int main()
{
   int shmid, pid;
   char *shared_memory;
   char buff[200];
```



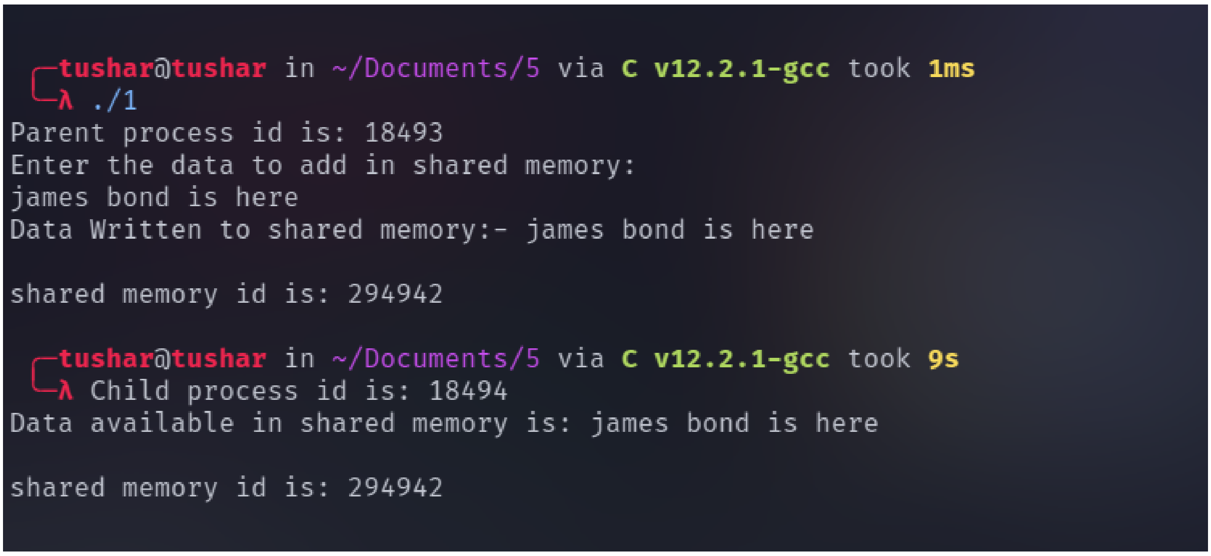
```
    shmid = shmget(((__key_t)0), 2048, 0666 | 01000);
    // ((__key_t) 0) is used to create shared memory segment
      // 0666 is used to give read and write permission to the
shared memory segment
    // 01000 is used to create shared memory segment
    if (shmid < 0)
    {
        perror("shmget failed");
        return 1;
    }
    pid = fork();
    if (pid > 0)
    {
        printf("Parent process id is: %d\n", getpid());
        shared_memory = (char *)shmat(shmid, NULL, 0);
        if (shared_memory == (void *)(-1))
        {
            perror("shmat failed");
            return 1;
        }
        printf("Enter the data to add in shared memory: \n");
          read(0, buff, 200); // read data from keyboard and store
in buffer
        strcpy(shared_memory, buff);
                printf("Data Written to shared memory:- %s\n",
shared_memory);
        printf("shared memory id is: %d\n", shmid);
        shmdt(shared_memory);
        return 0;
    }
    else
    {
        sleep(10);
        printf("Child process id is: %d\n", getpid());
        shared_memory = (char *)shmat(shmid, NULL, 0);
        if (shared_memory == (void *)(-1))
        {           perror("shmat failed");
```



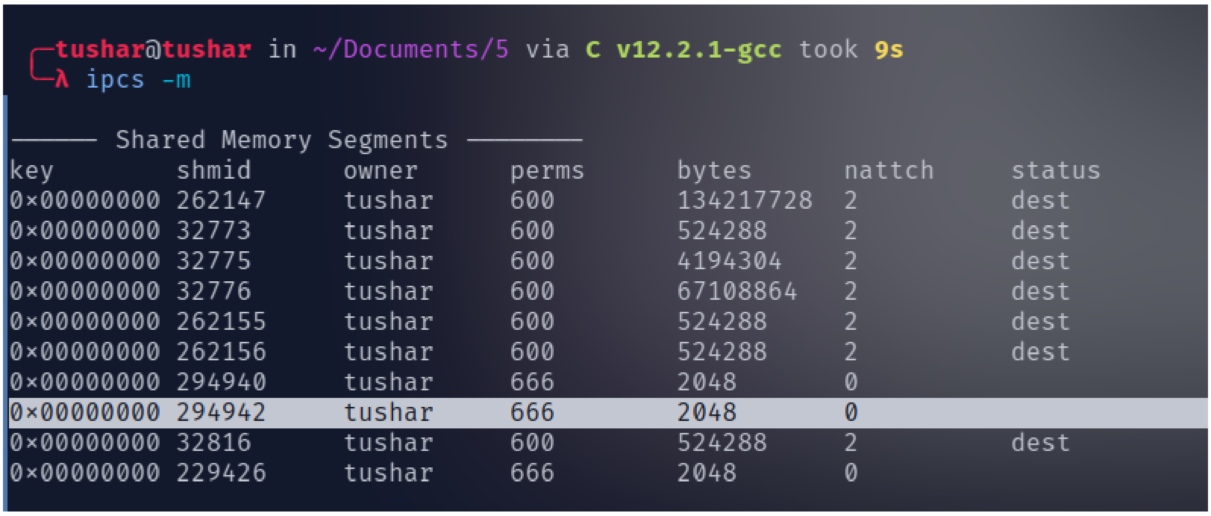
```
            return 1;
        }
        printf("Data available in shared memory is: %s\n",
               shared_memory);
        printf("shared memory id is: %d\n", shmid);
        shmdt(shared_memory);
        return 0;
    }
    return 0;
}
```

This is a syntax of **shmget** system call:-
int **shmget**(*key_t key,size_t size,int shmflg*);
Above code i am used:-
shmid = shmget(((__key_t) 0), 2048, 0666 | 01000)
01000 - is used to create shared memory segment -
IPC_CREAT
2048 - given in question
666 - for rw permission
((__key_t) 0) - default segment for shmget

✔ Output :



```
╭─tushar@tushar in ~/Documents/5 via C v12.2.1-gcc took 1ms
╰─λ ./1
Parent process id is: 18493
Enter the data to add in shared memory:
james bond is here
Data Written to shared memory:- james bond is here

shared memory id is: 294942

╭─tushar@tushar in ~/Documents/5 via C v12.2.1-gcc took 9s
╰─λ Child process id is: 18494
Data available in shared memory is: james bond is here

shared memory id is: 294942
```



```
┌─tushar@tushar in ~/Documents/5 via C v12.2.1-gcc took 9s
└─λ ipcs -m

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 262147     tushar     600        134217728  2          dest
0x00000000 32773      tushar     600        524288     2          dest
0x00000000 32775      tushar     600        4194304    2          dest
0x00000000 32776      tushar     600        67108864   2          dest
0x00000000 262155     tushar     600        524288     2          dest
0x00000000 262156     tushar     600        524288     2          dest
0x00000000 294940     tushar     666        2048       0
0x00000000 294942     tushar     666        2048       0
0x00000000 32816      tushar     600        524288     2          dest
0x00000000 229426     tushar     666        2048       0
```

➢ Question 2 :

2. Using shared Memory Concept, design the below scenario:

Input: Integer array of 10 Nos: 1,2,3,4,5,6,7,8,9,10

Operation : Addition of all ODD no by Parent Process & Addition of all EVEN no by Child Processes.

Output:

Child Sum : 30

Parent Sum : 25

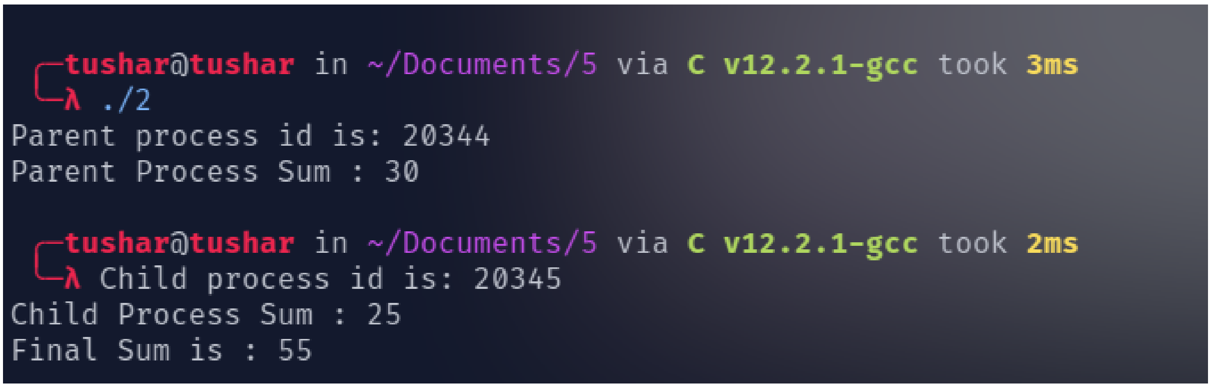
Final Sum is : 55

✔ Source Code :

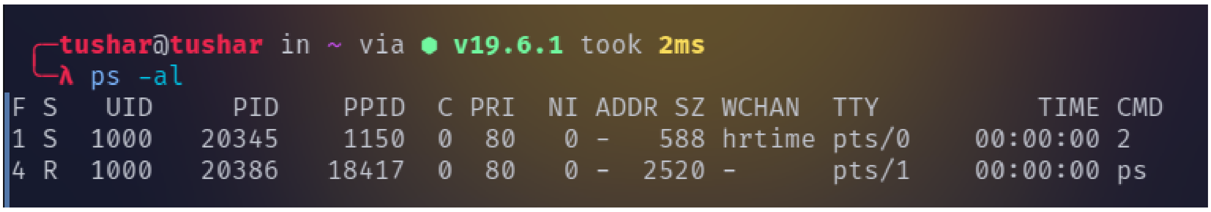
```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

#define shmget(a, b, c) syscall(29, a, b, c) // 29 is the number of
shmget system cell
// shmget is used to create shared memory segment
#define shmat(a, b, c) syscall(30, a, b, c) // 30 is the number of
shmat system call
// shmat is used to attach shared memory segment to the address space
of calling process
```

```
#define shmdt(a) syscall(67, a) // 67 is the number of shmdt system call
// shmdt is used to detach shared memory segment from the address space of calling process

int main()
{
    int shmid, pid;
    char *shared_memory;
    int sum = 0;
    int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

     shmid = shmget(((__key_t)0), 2048, 0666 | 01000); // 01000 is used to create shared memory segment
    if (shmid < 0)
    {
        perror("shmget Failed!");
        return 1;
    }

    pid = fork();
    if (pid > 0)
    {
        printf("Parent process id is : %d\n", getpid());

        shared_memory = (char *)shmat(shmid, NULL, 0);
        if (shared_memory == (void *)(-1))
        {
            perror("shmat Failed!");
            return 1;
        }
        for (int i = 0; i < 10; i++)
        {
            if (arr[i] % 2 == 0)
            {
                sum += arr[i];
            }
        }
        printf("Parent Process Sum : %d\n", sum);
          sprintf(shared_memory, "%d", sum); // store memory in shared memory and sprintf is used to store memory in buffer memory
        shmdt(shared_memory);
        return 0;
    }
    else
    {
        sleep(10);
        printf("Child Process id is : %d\n", getpid());

        shared_memory = (char *)shmat(shmid, NULL, 0);
        if (shared_memory == (void *)(-1))
        {
            perror("shmat Failed!");
            return 1;
        }
        for (int i = 0; i < 10 ; i++)
        {
            if (arr[i] % 2 != 0)
            {
                sum += arr[i];
            }
        }
        printf("Child Process Sum : %d\n", sum);
        printf("Final Sum is : %d\n", sum + atoi(shared_memory));
          // atoi is used to convert String to integer because shared memory is in string format also in buffer memory
        shmdt(shared_memory);
        return 0;
    }
    return 0;
}
```

✔ Output :



```
┌─tushar@tushar in ~/Documents/5 via C v12.2.1-gcc took 3ms
└─λ ./2
Parent process id is: 20344
Parent Process Sum : 30

┌─tushar@tushar in ~/Documents/5 via C v12.2.1-gcc took 2ms
└─λ Child process id is: 20345
Child Process Sum : 25
Final Sum is : 55
```



```
┌─tushar@tushar in ~ via ⬢ v19.6.1 took 2ms
└─λ ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
1 S  1000   20345    1150  0  80   0 -   588 hrtime pts/0    00:00:00 2
4 R  1000   20386   18417  0  80   0 -  2520 -      pts/1    00:00:00 ps
```