

Labsheet – 7

1. Write a C program that allows communication between a parent process and child process using Shared Memory.

Parent process has to take input n from the user, where n is an integer.

The parent process should then write n to the shared memory.

The child process has to read the contents of the shared memory and then print all the odd numbers till the limit n.

```
#include<stdio.h>
```

```
#include<sys/ipc.h>
```

```
#include<sys/shm.h>
```

```
#include<unistd.h>
```

```
#include<sys/wait.h>
```

```
int main()
```

```
{
```

```
    int shmid;
```

```
shmid = shmget(IPC_PRIVATE, sizeof(int), 0777|IPC_CREAT);
```

```
if (fork() == 0)
```

```
{
```

```
    int *a;
```

```
    a = (int *) shmat(shmid, 0, 0);
```

```
    sleep(4);
```

```
    printf("Child reads: %d\n",a[0]);
```

```
    printf("All odd numbers till %d : ",a[0]);
```

```
    for(int i=1;i<=a[0];i++)
```

```
    {
```

```
        if(i%2!=0)
```

```
        {
```

```
            printf("%d ",i);
```

```
        }
```

```
    }
```

```
    shmdt(a);
```

```
}
```

```
else
{
    int *a;

    a = (int *) shmat(shmid, 0, 0);

    printf("Enter the Number : ");

    int i;

    scanf("%d",&i);

    a[0] = i;

    printf("Parent writes: %d\n",a[0]);

    wait(NULL);

    shmdt(a);

    shmctl(shmid, IPC_RMID, 0);
}
}
```

```
Enter the Number : 5
Parent writes: 5
Child reads: 5
All odd numbers till 5 : 1 3 5
```

```
Enter the Number : 10
Parent writes: 10
Child reads: 10
All odd numbers till 10 : 1 3 5 7 9
```

2. Write a C program that allows communication between a parent process and child process using shared memory

a. Parent process has to take input string from the user.

b. The child process has to read the contents of the shared memory and convert it to capital letters and print it.

```
#include<stdio.h>
```

```
#include<sys/ipc.h>
```

```
#include<sys/shm.h>
```

```
#include<unistd.h>
```

```
#include<sys/wait.h>
```

```
int main()
```

```
{
```

```
    int shmid;
```

```
    shmid = shmget(IPC_PRIVATE, sizeof(char), 0777|IPC_CREAT);
```

```
    if (fork() == 0)
```

```
    {
```

```
        char *a;
```

```
        a = (char *) shmat(shmid, 0, 0);
```

```

sleep(5);

printf("Child reads: %s\n",a);

for(int i=0; a[i]!='\0'; i++)
{
    if(a[i]>='a' && a[i]<='z')
    {
        a[i] = a[i] - 32;
    }
}

printf("Upper Case : %s",a);

shmdt(a);
}

else
{
    char *a;

    a = (char *) shmat(shmid, 0, 0);

    printf("Enter the String : ");

    scanf("%s",a);

    printf("Parent writes: %s\n",a);

    wait(NULL);

    shmdt(a);
}

```

```
shmctl(shmid, IPC_RMID, 0);  
  
}  
  
}
```

```
Enter the String : Abhi  
Parent writes: Abhi  
Child reads: Abhi  
Upper Case : ABHI
```

```
Enter the String : AbhishekS  
Parent writes: AbhishekS  
Child reads: AbhishekS  
Upper Case : ABHISHEKS
```

3. Write a program that creates a shared memory segment and waits until two other separate processes writes something into that shared memory segment after which it prints what is written in shared memory.

For the communication between the processes to take place assume that the process 1 writes 1 in first position of shared memory and waits;

process 2 writes 2 in first position of shared memory and goes on to write 'hello' and then process 3 writes 3 in first position of shared memory and goes on to write 'memory' and finally the process 1 prints what is in shared memory written by two other processes.

```
#include <stdio.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <unistd.h>

#include <string.h>

#include <ctype.h>

#include<sys/wait.h>
```

```
int main()

{

    int shmid;

    int *a, *b, *c;

    int i=0, n;

    shmid = shmget(IPC_PRIVATE, sizeof(int), 0777|IPC_CREAT);

    b = (int *)shmat(shmid, 0, 0);

    b[0]=1;

    if(!fork())

    {

        c = (int *)shmat(shmid, 0, 0);

        c[0] = 2;

        printf("Hello\n");
```

```
shmdt(c);

if(!fork())

{

    a = (int *)shmat(shmid, 0, 0);

    a[0]=3;

    printf("Memory\n");

    shmdt(a);

}

}

else

{

    wait(NULL);

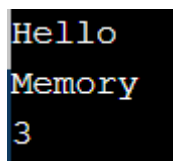
    printf("%d\n",b[0]);

    shmdt(b);

}

shmctl(shmid, IPC_RMID, 0);

}
```



```
Hello
Memory
3
```


4. Write a c program [using shared memory] to find average of square of numbers supplied by a user using 3 processes. 1 parent and two children. [Without buffer]

Parent should continuously take integers as input from the user until a special character, square it and supply it to both children.

Child #1 should find sum of these numbers, send it to the parent and exit.

Child #2 should count these numbers, send it to the parent and exit

Parent on getting response from both the children should find mean of square of numbers supplied by the user by dividing the child #1's result with child 2's and give it to the user.

```
#include<stdio.h>
```

```
#include<unistd.h>
```

```
#include<string.h>
```

```
#include<ctype.h>
```

```
#include<stdlib.h>
```

```
#include <sys/ipc.h>
```

```
#include <sys/shm.h>
```

```
#include<sys/wait.h>
```

```
int main()
```

```
{
```

```
int shmid,*arr,index=2,f=0;
```

```
char str[10];
```

```
shmid = shmget(IPC_PRIVATE, 100*sizeof(int), 0777|IPC_CREAT);
```

```
arr = (int *)shmat(shmid, 0, 0);
```

```
printf("PID --> %d and PPID --> %d\n",getpid(),getppid());
```

```
while(1)
```

```
{
```

```
    printf("Enter the number : ");
```

```
    scanf ("%s", str);
```

```
    int len = strlen (str);
```

```
    for (int i=0;i<len; i++)
```

```
    {
```

```
        if (!isdigit(str[i]))
```

```
        {
```

```
            f = 1;
```

```
            break;
```

```
        }
```

```
}
```

```
if (f == 1)
```

```
{
```

```
    printf("Oops!!..Special Character Found!\n\n");
```

```
    break;
```

```
}
```

```
int x = atoi(str);
```

```
arr[index] = x*x;
```

```
index++;
```

```
}
```

```
if(fork())
```

```
{
```

```
    if(!fork())
```

```
    {
```

```
        int sum=0,i=2;
```

```
        arr = (int *)shmat(shmid, 0, 0);
```

```
        printf("\nPID --> %d and PPID --> %d.",getpid(),getppid());
```

```
        printf("\nNumbers to be added are : ");
```

```

while(arr[i])

    {

        sum = sum+arr[i];

        printf("%d ",arr[i]);

        i++;

    }

arr[0] = sum;

    }

else

    {

        wait(NULL);

        printf("\n\nPID --> %d and PPID -->
%d.\n",getpid(),getppid());

        printf("Mean of Square of Numbers is
%.2f.",(float)arr[0]/arr[1]);

        shmdt(arr);

        shmctl(shmid, IPC_RMID, 0);

    }

}

else

{

    int count=2;

```

```

arr = (int *)shmat(shmid, 0, 0);

while(arr[count])

{

    count++;

}

arr[1] = count;

printf("PID --> %d and PPID --> %d.\n",getpid(),getppid());

printf("Count is %d\n",count);

shmdt(arr);

}

```

```

PID --> 1499 and PPID --> 1494
Enter the number : 1
Enter the number : 2
Enter the number : t
Oops!!..Special Character Found!

PID --> 1500 and PPID --> 1499.
Count is 4

PID --> 1501 and PPID --> 1499.
Numbers to be added are : 1 4
Sum of Numbers is 5.

PID --> 1499 and PPID --> 1494.
Mean of Square of Numbers is 1.25.

```

```
PID --> 2564 and PPID --> 2559
Enter the number : 6
Enter the number : 3
Enter the number : 2
Enter the number : r
Oops!!..Special Character Found!

PID --> 2565 and PPID --> 2564.
Count is 5

PID --> 2566 and PPID --> 2564.
Numbers to be added are : 36 9 4
Sum of Numbers is 49.

PID --> 2564 and PPID --> 2559.
Mean of Square of Numbers is 9.80.
```

One Drive : [Click Me!!](#)

Thankyou!