Write a program to generate and print Fibonacci series with the followingrequirements:
- Parent program should create a child and distribute the task of generatingFibonacci no to its child.
- The code for generating Fibonacci no. should reside in different program.- Child should write the generated Fibonacci sequence to a shared memory.- Parent process has to print by retrieving the Fibonacci sequence from theshared memory.
a) Implement the above using shm_open and mmap
b) Implement the above using shmget and shmat
Note: Shared object should be removed at the end in the program .

```
//mfib.c
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/mman.h>
int main(int argc,char *argv[])
{
    int i;
    pid_t pid;
    int k;
    int n1,n2,n3;
    const int SIZE = 4096;
        int shm_fd;
    void *ptr;
    if (argc > 1)
    {
        sscanf(argv[1], "%d", &i);
        if (i < 1)
        {
            printf("Error input: %d\n", i);
            return 0;
        }
    }
    else
    {
        return 1;
    }
    pid = fork();
    if (pid == 0)
    {
```

```
        execlp("./fib","fib",argv[1],NULL);
    }
    else if (pid > 0)
    {
        wait(NULL);
        printf("PARENT: child completed\n");

        shm_fd = shm_open("OS", O_RDONLY, 0666);
        ptr = mmap(0, SIZE, PROT_READ, MAP_SHARED, shm_fd, 0); printf("Parent
        printing:\n");
        printf("%s ", (char *)ptr);
        shm_unlink("OS");
    }
    return 0;
}

//fib.c
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/mman.h>
#include<stdlib.h>
int main(int argc,char *argv[]){
         int k=2,n1,n2,n3;
         void*ptr;
         int shm_fd = shm_open("OS", O_CREAT | O_RDWR, 0666);
         ftruncate(shm_fd,4096);
         ptr = mmap(0,4096,PROT_WRITE, MAP_SHARED, shm_fd, 0);
         printf("CHILD:\n");
         int i=atoi(argv[1]);
         n1=0;
         n2=1;
         sprintf(ptr,"%d ",n1);
         ptr+=strlen(ptr);
         printf("%d ",n1);
         sprintf(ptr,"%d ",n2);
         ptr+=strlen(ptr);
         printf("%d ",n2);
         while (k!=i)
         {
                n3=n1+n2;
                sprintf(ptr,"%d ", n3);
```

```
                printf("%d ", n3);
                n1=n2;
                n2=n3;
                ptr += strlen(ptr);
                k++;}
        }
```

```
// Simulation of ls command

#include<stdio.h>
#include<sys/types.h>
#include<dirent.h>
#include<stdlib.h>
void main(int argc, char *argv[])
{
DIR *dp;
struct dirent *dirp;
 if(argc<2)
{
printf("\n You not passing the directory\n"); exit(1);
}
if((dp=opendir(argv[1]))==NULL)
{
printf("\nCannot open it does't exist %s file!\n",argv[1]); exit(1);
}
while((dirp=readdir(dp))!=NULL)
 printf("%s\n",dirp->d_name);
closedir(dp);
}
```

```
// Simulation of rm command

#include<stdio.h>
#include<fcntl.h>
void main()
{

char fn[10];

printf("Enter source filename\n");
scanf("%s",fn);
if(remove(fn)==0)
printf("File removed\n");
else
printf("File cannot be removed\n");
}
```

```c
// Simulation of grep command

#include<stdio.h>
#include<string.h>
void main()
{
char fn[10],pat[10],temp[200];
FILE *fp;
printf("Enter file name\n");
scanf("%s",fn);
printf("Enter pattern to be searched\n"); scanf("%s",pat);
fp=fopen(fn,"r");
while(!feof(fp))
{
fgets(temp,1000,fp);
if(strstr(temp,pat))
  printf("%s",temp);
}
fclose(fp);
}
```

```c
// Simulation of cat command

#include<stdio.h>

#include<stdlib.h>

int main(int argc,char*argv[])

{

FILE *fp1;

char ch;

if(argc<2)

{

printf("\n You are not passing the File name\n"); exit(1);

}


fp1=fopen(argv[1],"r");

if(fp1 == NULL)

{
printf("File doesn't exists\n"); exit(1);

}
```

```
while((ch=fgetc(fp1))!=EOF) printf("%c",ch);
fclose(fp1);
return 0;
}
```