**LAB 10**

1. Write a program using fork system call such that parent process should calculate the sum of the two numbers and child process should calculate the multiplication of the same number.

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main()

{

int sum = 0, product = 0, n, i,j;

printf("input two number:");

scanf("%d %d", &i,&j);

n = fork();

// Checking if n is not 0

if (n > 0) {

printf(" Parent process \n");

sum=i+j;

printf("Sum of two no. is %d ", sum);

}

// If n is 0 i.e. we are in child process

else {

printf( "Child process \n");

product=i\*j;

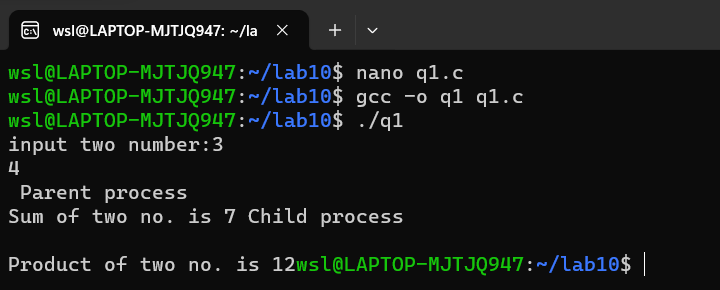
printf( "\nProduct of two no. is %d" ,product);

}

return 0;

}

**Output:**



2. WAP to simulate producer consumer problem using mutex.

**Code:**

#include<stdio.h>

#include<stdlib.h>

int mutex=1,full=0,empty=3,x=0;

int main()

{

int n;

void producer();

void consumer();

int wait(int);

int signal(int);

printf("\n1.Producer\n2.Consumer\n3.Exit");

while(1)

{

printf("\nEnter your choice:");

scanf("%d",&n);

switch(n)

{

case 1: if((mutex==1)&&(empty!=0)) producer();

else

printf("Buffer is full!!");

break;

case 2: if((mutex==1)&&(full!=0)) consumer();

else

printf("Buffer is empty!!");

break;

case 3:

exit(0);

break;

}

}

return 0;

}

int wait(int s)

{

return (--s);

}

int signal(int s)

{

return(++s);

}

void producer()

{

mutex=wait(mutex);

full=signal(full);

empty=wait(empty);

x++;

printf("\nProducer produces the item %d",x); mutex=signal(mutex);

}

void consumer()

{

mutex=wait(mutex);

full=wait(full);

empty=signal(empty);

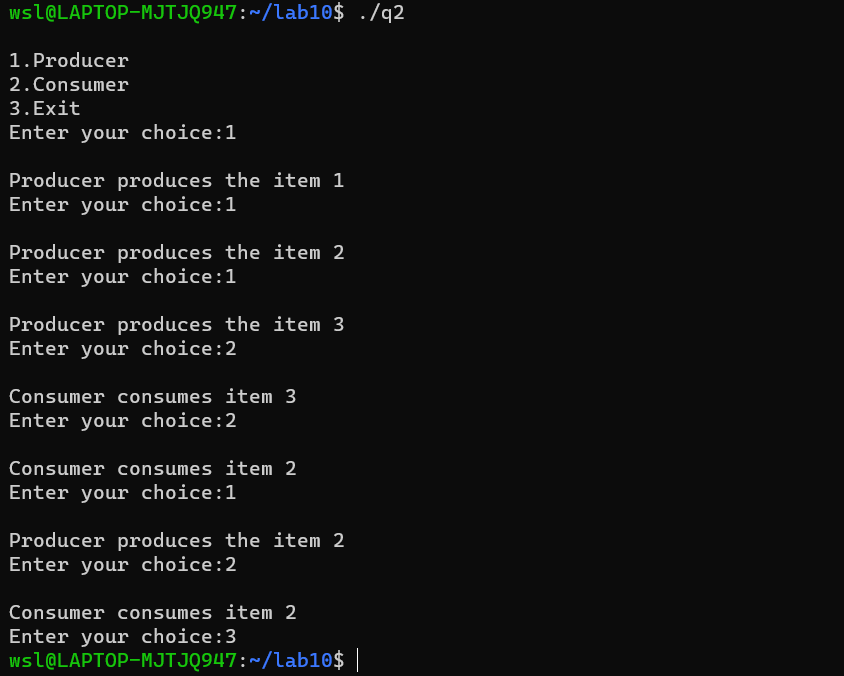
printf("\nConsumer consumes item %d",x);

x--;

mutex=signal(mutex);

}

**Output:**

****

4. WAP to implement sequential file allocation technique.

**Code:**

#include<stdio.h>

#include<conio.h>

int main()

{

int f[50], i, st, len, j, c, k, count = 0;

for(i=0;i<50;i++)

f[i]=0;

printf("Files Allocated are : \n");

begin:

printf("\n Enter the starting block and length of file:");

scanf("%d%d", &st,&len);

for(k=st;k<(st+len);k++)

if(f[k]==0)

count++;

if(len==count)

{

for(j=st;j<(st+len);j++)

if(f[j]==0)

{

f[j]=1;

printf("%d\t%d\n",j,f[j]);

}

if(j!=(st+len-1))

printf("The file is allocated to disk\n");

}

else

printf(" The file is not allocated \n");

printf("Do you want to enter more file(Yes - 1/No - 0)"); scanf("%d", &c);

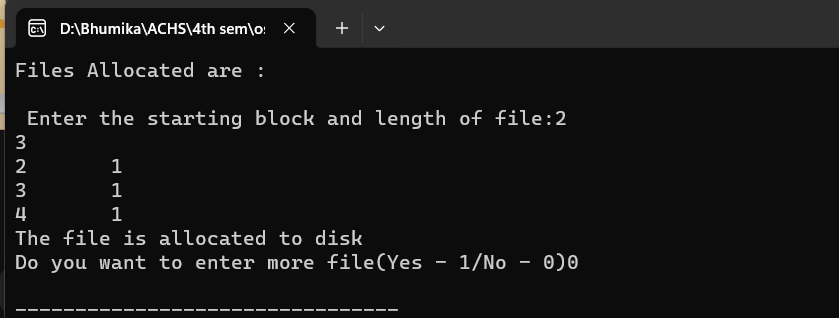
if(c==1)

goto begin;

return 0;

}

**Output:**



5. WAP to implement linked list file allocation technique. **Code:**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

int main()

{

int f[50], p,i, st, len, j, c, k, a;

for(i=0;i<50;i++)

f[i]=0;

printf("Enter how many blocks already allocated: "); scanf("%d",&p);

printf("Enter blocks already allocated: ");

for(i=0;i<p;i++)

{

scanf("%d",&a);

f[a]=1;

}

x: printf("Enter index starting block and length: "); scanf("%d%d", &st,&len);

k=len;

if(f[st]==0)

{

for(j=st;j<(st+k);j++)

{

if(f[j]==0)

{

f[j]=1;

printf("%d-------->%d\n",j,f[j]);

}

else

{

printf("%d Block is already allocated \n",j);

k++;

}

}

}

else

printf("%d starting block is already allocated \n",st); printf("Do you want to enter more file(Yes - 1/No - 0)"); scanf("%d", &c);

if(c==1)

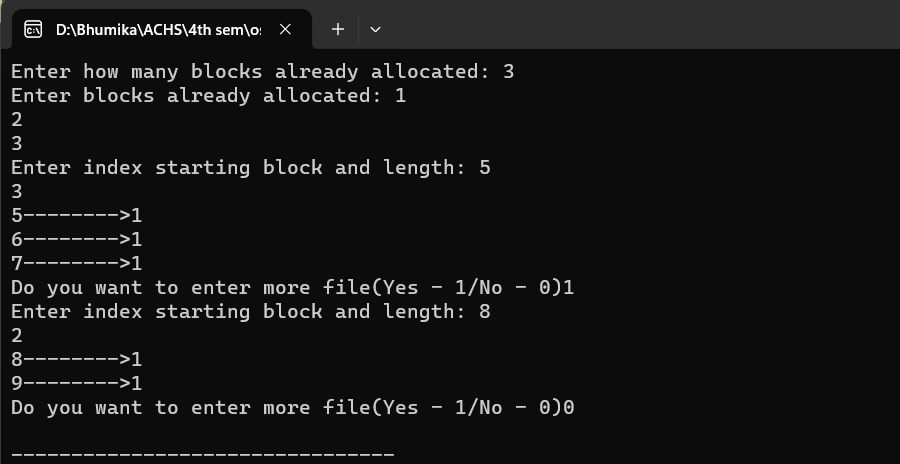
goto x;

else

exit(0);

}

**Output:**

****

6. WAP to implement single level directory.

**Code:**

#include<stdio.h>

#include<conio.h>

#include<string.h>

int main()

{

int nf=0,i=0,j=0,ch;

char mdname[10],fname[10][10],name[10];

printf("Enter the directory name:");

scanf("%s",mdname);

printf("Enter the number of files:");

scanf("%d",&nf);

do

{

printf("Enter file name to be created:");

scanf("%s",name);

for(i=0;i<nf;i++)

{

if(!strcmp(name,fname[i]))

break;

}

if(i==nf)

{

strcpy(fname[j++],name);

nf++;

}

else

printf("There is already %s\n",name);

printf("Do you want to enter another file(yes - 1 or no - 0):"); scanf("%d",&ch);

}

while(ch==1);

printf("Directory name is:%s\n",mdname);

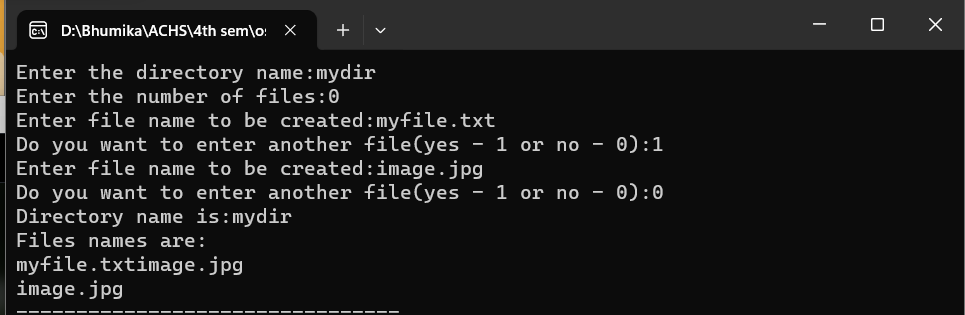
printf("Files names are:");

for(i=0;i<j;i++)

printf("\n%s",fname[i]);

}

**Output:**

****