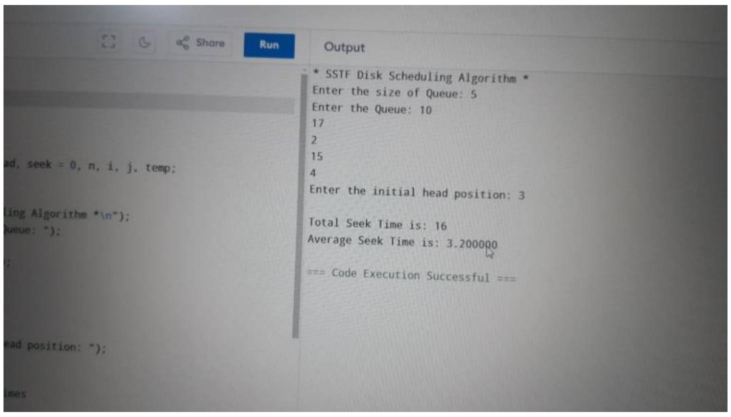
Bank

```
#include<stdio.h>#include<conio.h>int max[100][100];int alloc[100][100];

int need[100][100];int avail[100];int n,r;void input();void show();void cal();

int main(){int i,j;printf("********** Banker's Algorithm ************\n");

input();show();cal();getch();return 0;}void input(){int i,j;

printf("Enter the no of Processes\t");scanf("%d",&n);

printf("Enter the no of resources instances\t");scanf("%d",&r);

printf("Enter the Max Matrix\n");for(i=0;i<n;i++) {

 for(j=0;j<r;j++) { scanf("%d",&max[i][j]);}}

printf("Enter the Allocation Matrix\n");

 for(i=0;i<n;i++) { for(j=0;j<r;j++) {

 scanf("%d",&alloc[i][j]); } }

printf("Enter the available Resources\n");

for(j=0;j<r;j++) { scanf("%d",&avail[j]); }}

void show(){int i,j;

printf("Process\t Allocation\t Max\t Available\t");

 for(i=0;i<n;i++) { printf("\nP%d\t ",i+1);

 for(j=0;j<r;j++) { printf("%d ",alloc[i][j]); }

 printf("\t\t"); for(j=0;j<r;j++) {

 printf("%d ",max[i][j]); }

 printf("\t\t"); if(i==0) {

 for(j=0;j<r;j++) printf("%d ",avail[j]); }

 printf("\t\t"); } printf("\n");}


void cal(){

int finish[100],temp,need[100][100],flag=1,k,c1=0;

int safe[100];int i,j;for(i=0;i<n;i++){

 finish[i]=0; }
//find need matrix

for(i=0;i<n;i++) {

 for(j=0;j<r;j++) {

 need[i][j]=max[i][j]-alloc[i][j];} }

printf("\n");//print need matrix

printf("----Need Matrix-------\n");

for(i=0;i<n;i++) { for(j=0;j<r;j++) {

 printf("%d ",need[i][j]); }

 printf("\n"); } printf("\n");while(flag){

 flag=0; for(i=0;i<n;i++) { int c=0;

 for(j=0;j<r;j++) {

 if((finish[i]==0)&&(need[i][j]<=avail[j])) {

 c++; if(c==r) { for(k=0;k<r;k++) {

 avail[k]+=alloc[i][j];

 finish[i]=1; flag=1; }

 printf("P%d->",i); if(finish[i]==1) {

 i=n;}}}}}}printf("\n\n");

for(i=0;i<n;i++) {

 if(finish[i]==1) { c1++; } else {

 printf("P%d->",i); } }

if(c1==n) {

 printf("\n The system is in safe state"); }

else { printf("\n Process are in dead lock");

 printf("\n System is in unsafe state"); }}
```

ssft

```
#include <stdio.h>

#include <math.h>

#include<stdlib.h>

int main()

{

    int queue[100], t[100], head, seek = 0, n, i, j, temp;

    float avg;


    printf("* SSTF Disk Scheduling Algorithm *\n");

    printf("Enter the size of Queue: ");

    scanf("%d", &n);

    printf("Enter the Queue: ");

    for (i = 0; i < n; i++) {

        scanf("%d", &queue[i]);

    }


    printf("Enter the initial head position: ");

    scanf("%d", &head);


    // Calculate initial seek times

    for (i = 0; i < n; i++) {

        t[i] = abs(head - queue[i]);

    }


    // Sort the queue based on seek times (SSTF)

    for (i = 0; i < n - 1; i++) {

        for (j = i + 1; j < n; j++) {

            if (t[i] > t[j]) {
                temp = t[i];

                t[i] = t[j];

                t[j] = temp;

                temp = queue[i];

                queue[i] = queue[j];

                queue[j] = temp;

            }

        }

    }


    // Calculate total seek time and update head position

    for (i = 0; i < n; i++) {

        seek += abs(head - queue[i]);

        head = queue[i];

    }


    // Display results

    printf("\nTotal Seek Time is: %d\t", seek);

    avg = (float)seek / n;

    printf("\nAverage Seek Time is: %f\t", avg);

    return 0;

}
```

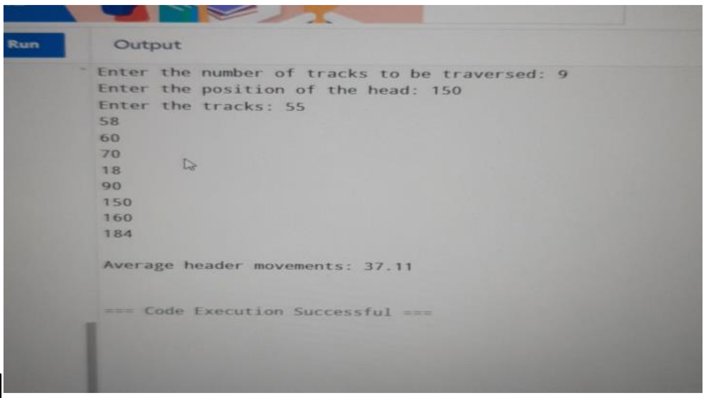


```
ad, seek = 0, n, i, j, temp;

ling Algorithm *\n");
Queue: ");

;

ead position: ");

imes
```

```
* SSTF Disk Scheduling Algorithm *
Enter the size of Queue: 5
Enter the Queue: 10
17
2
15
4
Enter the initial head position: 3

Total Seek Time is: 16
Average Seek Time is: 3.200000

=== Code Execution Successful ===
```

Scan

```
#include <stdio.h>

int main() {

   int t[20], d[20], h, i, j, n, temp, k, atr[20], tot, p, sum = 0;

   printf("Enter the number of tracks to be traversed: ");

   scanf("%d", &n);

   printf("Enter the position of the head: ");

   scanf("%d", &h);

   t[0] = 0;

   t[1] = h;

   printf("Enter the tracks: ");

   for (i = 2; i < n + 2; i++) {

      scanf("%d", &t[i]);

   }

   // Sorting the tracks

   for (i = 0; i < n + 2; i++) {

      for (j = 0; j < (n + 2) - i - 1; j++) {

         if (t[j] > t[j + 1]) {

            temp = t[j];

            t[j] = t[j + 1];

            t[j + 1] = temp;

         }}}

   // Find the position of the head in the sorted array

   for (i = 0; i < n + 2; i++) {

      if (t[i] == h) {

         j = i; // Position of the head in the sorted array

         k = i; // Temporary variable for later use
```



```
Output
Enter the number of tracks to be traversed: 9
Enter the position of the head: 150
Enter the tracks: 55
58
60
70
18
90
150
160
184

Average header movements: 37.11

=== Code Execution Successful ===
```

```
        p = 0; // Initialize position for atr[]
    }}
// Traverse left
while (t[j] != 0) {
    atr[p] = t[j];
    j--;
    p++;
}
atr[p] = t[j]; // Add the 0 position to the array
// Traverse right
for (p = k + 1; p < n + 2; p++, k++) {
    atr[p] = t[k + 1];
}
// Calculate the distances and total seek time
for (j = 0; j < n + 1; j++) {
    if (atr[j] > atr[j + 1]) {
        d[j] = atr[j] - atr[j + 1];
    } else {
        d[j] = atr[j + 1] - atr[j];
    }
    sum += d[j];
}
printf("\nAverage header movements: %.2f\n", (float)sum / n); return o;}
```

first fit

```
#include<stdio.h>#include<conio.h>#define max 25

void main(){

int frag[max],b[max],f[max],i,j,nb,nf,temp;

static int bf[max],ff[max];clrscr();

printf("\n\tMemory Management Scheme - First Fit");

printf("\nEnter the number of blocks:");

scanf("%d",&nb);

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

scanf("%d",&nf);

printf("\nEnter the size of the blocks:-\n");

for(i=1;i<=nb;i++){

printf("Block %d:",i); scanf("%d",&b[i]);}

printf("Enter the size of the files :-\n");

for(i=1;i<=nf;i++){

printf("File %d:",i);

scanf("%d",&f[i]);}

for(i=1;i<=nf;i++){

for(j=1;j<=nb;j++){

if(bf[j]!=1){

temp=b[j]-f[i];if(temp>=0){ff[i]=j;break;}}}

frag[i]=temp;bf[ff[i]]=1;}

printf("\nFile_no:\tFile_size :\tBlock_no:\tBlock_size:\tFragement");

for(i=1;i<=nf;i++)

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d",i,f[i],ff[i],b[ff[i]],frag[i]);

getch();}
```

worst fit

```
#include<stdio.h>#include<conio.h>#define max 25

void main(){

int frag[max],b[max],f[max],i,j,nb,nf,temp,highest=0;

static int bf[max],ff[max];clrscr();

printf("\n\tMemory Management Scheme - Worst Fit");

printf("\nEnter the number of blocks:");

scanf("%d",&nb);

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

scanf("%d",&nf);

printf("\nEnter the size of the blocks:-\n");

for(i=1;i<=nb;i++){

printf("Block %d:",i);

scanf("%d",&b[i]);}

printf("Enter the size of the files :-\n");

for(i=1;i<=nf;i++){

printf("File %d:",i);

scanf("%d",&f[i]);}

for(i=1;i<=nf;i++){

for(j=1;j<=nb;j++){

if(bf[j]!=1){temp=b[j]-f[i];

if(temp>=0)if(highest<temp){

ff[i]=j;highest=temp;}}}

frag[i]=highest;bf[ff[i]]=1;

highest=0;}

printf("\nFile_no:\tFile_size :\tBlock_no:\tBlock_size:\tFragement");

for(i=1;i<=nf;i++)

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d",i,f[i],ff[i],b[ff[i]],frag[i]);

getch();}
```

A. FIFO

```
#include<stdio.h>#include<conio.h>main(){

int i, j, k, f, pf=0, count=0, rs[25], m[10], n;

clrscr();

printf("\n Enter the length of reference string -- ");

scanf("%d",&n); printf("\n Enter the reference string -- ");

for(i=0;i<n;i++)scanf("%d",&rs[i]);

printf("\n Enter no. of frames -- ");

scanf("%d",&f);for(i=0;i<f;i++)m[i]=-1;

printf("\n The Page Replacement Process is -- \n");

for(i=0;i<n;i++){

for(k=0;k<f;k++){

if(m[k]==rs[i]) break;}

if(k==f){

m[count++]=rs[i];pf++;}

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

printf("\t%d",m[j]);if(k==f)

printf("\tPF No. %d",pf); printf("\n");

if(count==f)count=0;

}

printf("\n The number of Page Faults using FIFO are %d",pf);

getch();

}
```

B. LRU

```
#include<stdio.h>

#include<conio.h>

main(){

int i, j , k, min, rs[25], m[10], count[10], flag[25], n, f, pf=0, next=1;

clrscr();printf("Enter the length of reference string -- ");

scanf("%d",&n); printf("Enter the reference string -- ");

for(i=0;i<n;i++){

scanf("%d",&rs[i]);flag[i]=0;}

printf("Enter the number of frames -- ");

scanf("%d",&f);for(i=0;i<f;i++){

count[i]=0;m[i]=-1;}

printf("\nThe Page Replacement process is -- \n");

for(i=0;i<n;i++){for(j=0;j<f;j++){

if(m[j]==rs[i]){

flag[i]=1;count[j]=next; next++;}}

if(flag[i]==0){if(i<f){

m[i]=rs[i];count[i]=next;next++;}

else{min=0;for(j=1;j<f;j++)

if(count[min] > count[j])

min=j;m[min]=rs[i];

count[min]=next;next++;}pf++;}

for(j=0;j<f;j++)printf("%d\t", m[j]);

if(flag[i]==0)printf("PF No. -- %d" , pf);

printf("\n");}

printf("\nThe number of page faults using LRU are %d",pf);

getch();}
```

A. SEQUENTIAL FILE ALLOCATION

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

struct fileTable{char name[20];

int sb, nob;}ft[30];

void main(){

int i, j, n;char s[20];

clrscr();printf("Enter no of files :");

scanf("%d",&n);for(i=0;i<n;i++){

printf("\nEnter file name %d :",i+1);

scanf("%s",ft[i].name);

printf("Enter starting block of file %d :",i+1);

scanf("%d",&ft[i].sb);

printf("Enter no of blocks in file %d :",i+1);

scanf("%d",&ft[i].nob);}

printf("\nEnter the file name to be searched -- ");

scanf("%s",s);for(i=0;i<n;i++)

if(strcmp(s, ft[i].name)==0)break;if(i==n)

printf("\nFile Not Found");else{

printf("\nFILE NAME START BLOCK NO OF BLOCKS BLOCKS OCCUPIED\n");

printf("\n%s\t\t%d\t\t%d\t",ft[i].name,ft[i].sb,ft[i].nob);

for(j=0;j<ft[i].nob;j++)printf("%d, ",ft[i].sb+j);

}getch();}
```

B. INDEXED FILE ALLOCATION

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

struct fileTable{

char name[20];int nob, blocks[30];

}ft[30];void main(){

int i, j, n;char s[20];

clrscr();printf("Enter no of files :");

scanf("%d",&n);

for(i=0;i<n;i++){

printf("\nEnter file name %d :",i+1);

scanf("%s",ft[i].name);

printf("Enter no of blocks in file %d :",i+1);

scanf("%d",&ft[i].nob);

printf("Enter the blocks of the file :");

for(j=0;j<ft[i].nob;j++)

scanf("%d",&ft[i].blocks[j]);}

printf("\nEnter the file name to be searched -- ");

scanf("%s",s);for(i=0;i<n;i++)

if(strcmp(s, ft[i].name)==0)

break;if(i==n)

printf("\nFile Not Found");

else{printf("\nFILE NAME NO OF BLOCKS BLOCKS OCCUPIED");

printf("\n %s\t\t%d\t",ft[i].name,ft[i].nob);

for(j=0;j<ft[i].nob;j++)

printf("%d, ",ft[i].blocks[j]);}getch();}
```