Muchamad Rif'an

17.01.53.2021

Teknik Informatika R2

MANAJEMEN MEMORI

Objective

**Write a C program to simulate the following contiguous memory allocation techniques.

1. Worst-fit.
2. Best-fit.
3. First-fit.

Description

One of the simplest methods for memory allocation is to divide memory into several fixed-sized partitions. Each partition may contain exactly one process. In this multiple-partition method, when a partition is free, a process is selected from the input queue and is loaded into the free partition. When the process terminates, the partition becomes available for another process. The operating system keeps a table indicating which parts of memory are available and which are occupied. Finally, when a process arrives and needs memory, a memory section large enough for this process is provided. When it is time to load or swap a process into main memory, and if there is more than one free block of memory of sufficient size, then the operating system must decide which free block to allocate. Best-fit strategy chooses the block that is closest in size to the request. First-fit chooses the first available block that is large enough. Worst-fit chooses the largest available block.

Program

1. Worst-fit

```
#include<stdio.h>
#define max 25
void main()
{
int frag[max],b[max],f[max],i,j,nb,nf,temp;
static int bf[max],ff[max];
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]);
}
}
```

INPUT

Enter the number of blocks : 4

Enter the number of files : 5

Enter the size of blocks:

Block 1: 10

Block 2: 20

Block 3: 30

Block 4: 40

Enter the size of the files:

File 1 : 5

File 2 : 20

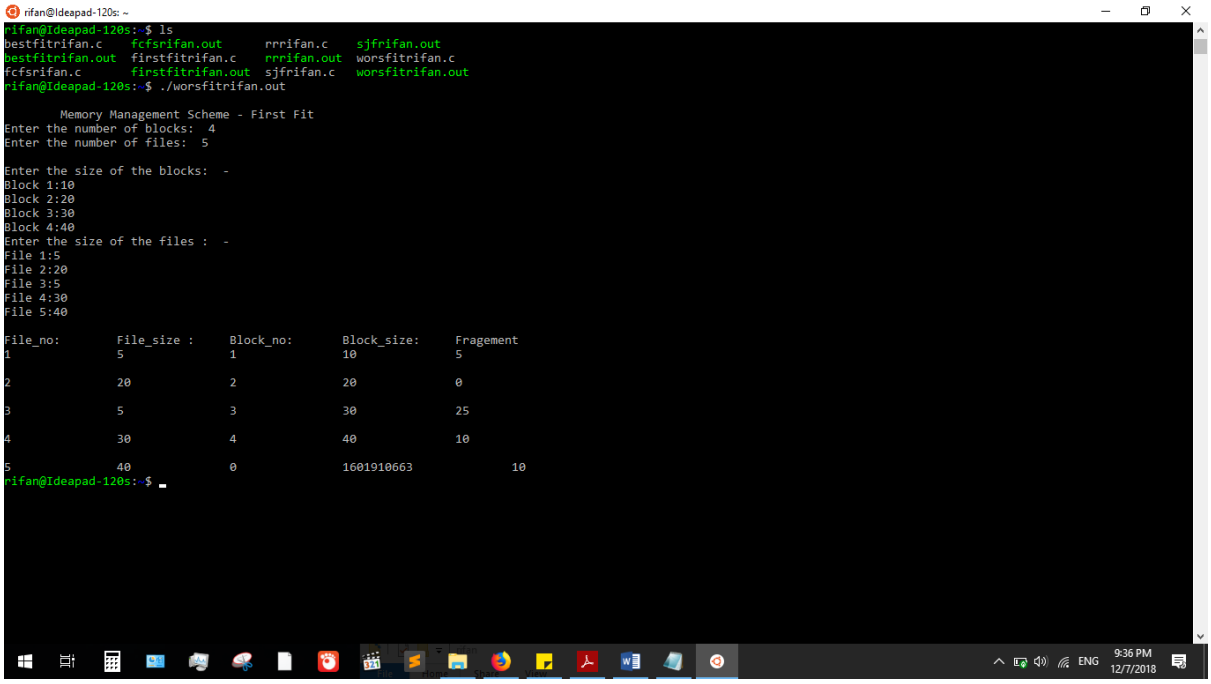
File 3 : 5

File 4 : 30

File 5 : 40

OUTPUT

FILE	FILE SIZE	BLOCK NO	BLOCK SIZE	FRAGMENT
1	5	1	10	5
2	20	2	20	0
3	5	3	30	25
4	30	4	40	10
5	40	0	1023096711	10



2. Best-fit

```
#include<stdio.h>
#define max 25
void main()
{
        int frag[max],b[max],f[max],i,j,nb,nf,temp,lowest=10000;
        static int bf[max],ff[max];
        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(lowest>temp)
                {
                ff[i]=j;
                lowest=temp;
                }
                }
                }
                frag[i]=lowest;
                bf[ff[i]]=1;
                lowest=10000;
                }
                printf("\nFile No\tFile Size \tBlock No\tBlock Size\tFragment");
                for(i=1;i<=nf && ff[i]!=0;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();
}
```

INPUT

Enter the number of blocks : 5

Enter the number of files : 4

Enter the size of blocks:

Block 1: 5

Block 2: 5

Block 3: 20

Block 4: 30

Block 5: 40

Enter the size of the files:

File 1 : 10

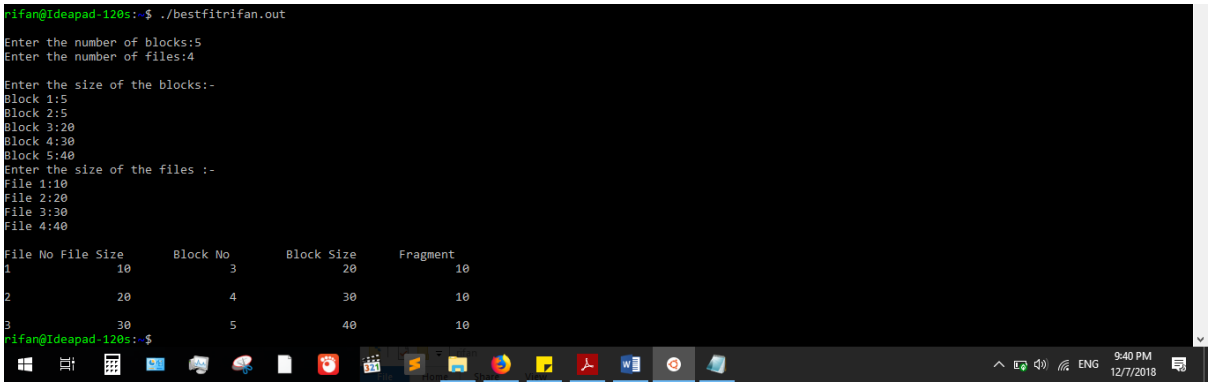
File 2 : 20

File 3 : 30

File 4 : 40

OUTPUT

FILE	FILE SIZE	BLOCK NO	BLOCK SIZE	FRAGMENT
1	10	3	20	10
2	29	4	30	10
3	30	5	40	10



3. First-fit

```
#include<stdio.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];
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) //if bf[j] is not allocated
{
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]);
```

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

INPUT

Enter the number of blocks : 3

Enter the number of files : 4

Enter the size of blocks:

Block 1: 10

Block 2: 20

Block 3: 30

Enter the size of the files:

File 1 : 15

File 2 : 15

File 3 : 20

File 4 : 10

OUTPUT

FILE	FILE SIZE	BLOCK NO	BLOCK SIZE	FRAGMENT
1	15	3	30	15
2	15	2	20	5
3	20	0	-1923401849	0
4	10	0	-1923401849	0