WEEK 9

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

- a) Worst-fit
- b) Best-fit
- c) First-fit

CODE:

a. 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;
  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 \le nb; i++)
  {
       printf("Block %d:", i);
       scanf("%d", &b[i]);
```

```
}
printf("Enter the size of the files:\n");
for (i = 1; i \le nf; i++)
{
     printf("File %d:", i);
     scanf("%d", &f[i]);
}
for (i = 1; i \le nf; i++)
     temp = -1; // Reset temp to -1 for each new file
     for (j = 1; j \le nb; j++)
     if (bf[j] != 1)
             if (b[j] >= f[i])
             {
             ff[i] = j;
             temp = b[j] - f[i];
             break;
             }
     }
     frag[i] = temp;
     if (temp != -1)
     bf[ff[i]] = 1;
     }
```

```
}
  printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
  for (i = 1; i \le nf; i++)
       printf("\n\%d\t\t\%d\t\t\%d\t\t\%d", i, f[i], ff[i], b[ff[i]], frag[i]);
  }
  getch();
}
a )Best fit
#include <stdio.h>
#include <conio.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 \le nb; i++)
  {
       printf("Block %d:", i);
       scanf("%d", &b[i]);
```

```
}
printf("Enter the size of the files:\n");
for (i = 1; i \le nf; i++)
{
     printf("File %d:", i);
     scanf("%d", &f[i]);
}
for (i = 1; i \le nf; i++)
     lowest = 10000; // Reset lowest to a high value for each new file
     for (j = 1; j \le nb; j++)
     if (bf[j] != 1)
             temp = b[j] - f[i];
             if (temp \ge 0 \&\& lowest > temp)
        {
             ff[i] = j;
             lowest = temp;
             }
     }
     frag[i] = lowest;
     bf[ff[i]] = 1;
}
printf("\nFile No\tFile Size\tBlock No\tBlock Size\tFragment");
for (i = 1; i \le 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]);
}
```

a. 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;
  int bf[max], ff[max]; // Initialized these arrays to 0
  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 \le nb; i++)
  {
       printf("Block %d:", i);
       scanf("%d", &b[i]);
  }
  printf("Enter the size of the files:\n");
  for (i = 1; i \le nf; i++)
```

```
{
     printf("File %d:", i);
     scanf("%d", &f[i]);
}
for (i = 1; i \le nf; i++)
{
     highest = 0; // Reset highest to 0 for each new file
     for (j = 1; j \le 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;
}
printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragement");
for (i = 1; i \le nf; i++)
```

OBSERVATION:

```
Week 9

White a fregram to emulate continue mercy allocation techniques

allocation techniques

allocation techniques

b court fit

Thust fit

o) H include coldie ho

H define max 10

roid main()

int freg(max), f(max), f(max), i, ij, nb, nf, timp, high = 0, bf(max), f(max),

frentf(! Intro number of blocks land files");

fourtf(! Intro number of blocks land files");

fourtf(! Intro supe of blocks);

fourtf(! Intro supe of blocks);

fourtf(! int, i+t)

Scanf(! I d & b(i)),

for (i=1; inf, i+t)

high = 0

for (ij=1; inf, i+t)

high = 0

for (ij=1; inf, i+t)

figh(bf(ij)! = 0

tomp = b(ij)-f(i);

if (temp >0)

file I j;
```

fruitf (File no It File sig It Block no It Block signification of the form of the signification of the significant of the si

find (i) = lowest,

furth ("File no It File suge It Block no It Block of time

for (d = 1; ic = nf; i + t)

fourth ("I'd It It I d It It I d It It I de It

i, if (i), if (i), if (i), biff (i) fing (ii)

ord main ()

Int force (max);

fourth ("Inter no. of blocks and files"),

sean ("I'd", bid", & nb. Enf);

for (f = 1; iz = nb. i + t)

for (f = 1; iz = nf; i + t)

for (f = 1; j = nf; i + t)

for (f = 1; j = nf; f + t)

if (b(j) = f(i))

if (b(j) = f(i))

If (b(j) = f(i))

Jenak,

3

3

```
If literal to the size it Black no it Black size if literal literal to the size it Black no it Black size if local size is a size if literal l
```

OUTPUT:

```
Memory Management Scheme - First Fit
Enter the number of blocks:3
Enter the number of files:2

Enter the size of the blocks:
Block 1:5
Block 2:2
Block 3:7
Enter the size of the files:
File 1:1
File 2:4

File_no: File_size: Block_no: Block_size: Fragment
1 1 1 5 4
2 4 3 7 3
```

```
Enter the number of blocks:3
Enter the number of files:2

Enter the size of the blocks:
Block 1:5
Block 2:2
Block 3:7
Enter the size of the files:
File 1:1
File 2:4

File No File Size Block No Block Size Fragment
1 1 2 2 1
2 4 1 5
```