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

a) Worst-fit

b) best-fit

c) First-fit

Page No.
Date 04 07 24

```
#include<stdio.h>
#define MAX 25

void firstfit(int nb, int nf, int b[], int f[]) {
    int frag[MAX], bf[MAX] = {0}, ff[MAX] = {0};
    int i, j, temp;

    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;
                    frag[i] = temp;
                    bf[j] = 1;
                    break;
                }
            }
        }
    }
    printf("\n Memory Management Scheme - First fit \n");
    printf(" File no:\t File size:\t Block no:\t Block size:\t Fragment\n");
    for(i=0; i<=nf; i++) {
        printf("%d\t\t%d\t\t", i, f[i]);
        if(ff[i] != 0) {
            printf("%d\t\t%d\t\t%d\n", ff[i], b[ff[i]], frag[i]);
        } else {
            printf("Not allocated\n");
        }
    }
}
```

```
void bestfit(int nb, int nf, int b[], int f[]) {
    int frag[MAX], bf[MAX] = {0}, ff[MAX] = {0};
    int i, j, temp, lowest = 10000;

    for(i=1; i<=nf; i++) {
        for (j=1; j<=nb; j++) {
            if (bf[j] != 1) {
                temp = b[j] - f[i];
                if (temp>=0 && lowest > temp) {
                    ff[i] = j;
                    lowest = temp;
                }
            }
        }
        frag[i] = lowest;
        bf[ff[i]] = 1;
        lowest = 10000;
    }
    printf("\n In Memory Management Scheme - Best fit \n");
    printf("File no\t File Size \t Block no \t Block Size \t Fragment \n");
    for (i=1; i<=nf; i++) {
        printf("%d\t\t %d\t\t %d\t\t", i, f[i]);
        if (ff[i] != 0) {
            printf("%d\t\t %d\t\t %d\n", ff[i], b[ff[i]], frag[i]);
        } else {
            printf("Not Allocated\n");
        }
    }
}

void worstfit(int nb, int nf, int b[], int f[]) {
    int frag[MAX], bf[MAX] = {0}, ff[MAX] = {0};
    int i, j, temp, highest = 0;

    for(i=1; i<=nf; i++) {
        for(j=1; j<=nb; j++) {
```

```
if(bf[j] != 1) {
    temp = b[j] - f[i];
    if(temp >= 0 && highest < temp) {
        ff[i] = j;
        highest = temp;
    }
}
frag[i] = highest;
bf[ff[i]] = 1;
highest = 0;
}
printf("\n Memory management Scheme - Worst Fit \n");
printf("File no:\t File size:\t Block no:\t Block size:\t Fragment\n");
for(i=1; i<=nf; i++) {
    printf("%d\t\t%d\t\t", i, f[i]);
    if(ff[i] != 0) {
        printf("%d\t\t%d\t\t%d\n", ff[i], b[ff[i]], frag[i]);
    } else {
        printf("Not Allocated\n");
    }
}
}
void main() {
int b[MAX], f[MAX], nb, nf;
printf("\nEnter the no. of blocks:");
scanf("%d", &nb);
printf("Enter the no. of files:");
scanf("%d", &nf);
printf("enter the size of blocks:-\n");
for(int i=1; i<=nb; i++) {
    printf("Block %d:", i);
    scanf("%d", &b[i]);
}
```

FirstFit(nb, nf, b, f);
BestFit(nb, nf, b, f);
WorstFit(nb, nf, b, f);

3

Memory Management Scheme - First fit

File_no:	File_size:	Block_no:	Block_size	Fragment
1	212	2	500	288
2	417	5	600	183
3	112	3	200	88
4	426	Not allocated		

Memory management Scheme - Best fit

File no	File size	Block no	Block size	Fragment
1	212	4	300	88
2	417	2	500	83
3	112	3	200	88
4	426	5	600	174

Memory management Scheme - Worst fit

File no	File size	Block no	Block size	Fragment
1	212	5	600	388
2	417	2	500	83
3	112	4	300	188
4	426	Not Allocated		

4/7/24