2024/07/04
Thursday

Week-8

write a cprogram to simulate the following contiguous memory allocation techniques

a) worst fit
b) Best fit
c) First fit.

```
#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("\nmemory management scheme-First Fit\n");
printf("File no:\tFile size:\tBlock no:\tBlock size:\tFragment");
for(i=1;i<=nf;i++){
    printf("\n%d\t%d\t",i,f[i]);
    if(ff[i]!=0){
        printf("%d\t%d\t%d",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 = 1000;
  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 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%d\t\t%d\n", i, f[i]);
    if(ff[i] != 0) {
      printf("%d\t\t%d\t\t%d\t\t%d\n", i, 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\nMemory management scheme-Worst fit\n");
    printf("File no:\tFile size:\tBlock no:\tBlock size:\tFragment\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");
        }
    }
}

int main(){
    int b[maxs],f[maxs],nb,nf;
    printf("\nEnter the number of blocks:");
    scanf("%d",&nb);
    printf("\nEnter the size of the blocks:\n");
    for(int i=1;i<=nb;i++){
        printf("Block%d:",i);
        scanf("%d",&b[i]);
    }
```

```
printf("Enter the size of the files:\n");
for(int i=1;i<=nf;i++){
  printf("file no:%d\n",i);
  scanf("%d",&ff[i]);
}

int b1[max],b2[max],b3[max];
for(int i=1;i<=nb;i++){
  b1[i]=b[i];
  b2[i]=b[i];
  b3[i]=b[i];
}

firstfit(nb,nf,b1,ff);
bestfit(nb,nf,b2,ff);
worstfit(nb,nf,b3,ff);

return 0;
}
```

Output:

```
Enter the no. of blocks:6
Enter the no. of files:4
Enter the size of blocks.
b1-200 b2-400 b3-600 b4-500 b5-300

Enter the size of the files:
F1->357, F2-210, F3-468, F4-491
```

First fit.

File no	File-Size:	Block-no:	Blocksize	Fragment
1	357	2	400	43
2	210	3	600	390
3	468	4	500	32
4	491	Not Allocated		

Memory management scheme: Best fit

	File size	Block no.	Block size	Fragment
1	357	2	400	43
2	210	6	250	40
3	468	4	500	32
4	491	3	600	109

Worst fit:

	File size	Block no:	Block size	Fragment
1	357	3	600	243
2	210	4	500	290
3	468	Not Allocated		
4	491	Not Allocated		

Sum
4/7/24