SURYA Gold
Date ______ Page ______

4/7/2024

Write a C program to simulate the following contiguous memory allocation techniques
a) worst-fit
b) Best-fit
c) First-fit

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

~~void worstfit(int blocksize[], int~~
void firstFit (int nb, int nf, int b[], int f[])
{
        int ff[MAX] = {0};
        int allocated [MAX] = {0};
        for (int i=0; i<nf ; i++) {
            ff[i] = -1;
            for (int j=0; j<nb; j++) {
                if (allocated[j] == 0 && b[j] >= f[i]) {
                    ff[i] = j;
                    allocated[j] = 1;
                    break;
                }
            }
    }

    printf("\nFile_no :\tFile_size :\t Block_no :\t Block
            _size:");
    for(int i=0; i<nf; i++) {
        if (ff[i] != -1)
            printf("\n%d\t\t%d\t\t%d\t\t%d", i+1,
                   f[i], ff[i]+1, b[ff[i]]);
    else
```

```
        printf("\n%d\t\t%d\t\t-\t\t-", i+1, f[i]);
    }
}

void bestFit (int nb, int nf, int b[], int f[])
{
    int ff[MAX] = {0};
    int allocated[MAX] = {0};
    for(int i=0; i<nf; i++){
        int best = -1;
        ff[i] = -1;
        for (int j=0; j<nb; j++){
            if(allocated[j] == 0 && b[j] >= f[i]) {
                if (best == -1 || b[j] < b[best])
                    best = j;
            }
        }
        if (best != -1) {
            ff[i] = best;
            allocated[best] = 1;
        }
    }

    printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:");
    for(int i=0; i<nf; i++){
        if (ff[i] != -1)
            printf("\n%d\t\t%d\t\t%d\t\t%d", i+1,
                   f[i], ff[i]+1, b[ff[i]]);
        else
            printf("\n%d\t\t%d\t\t-\t\t-", i+1, f[i]);
    }
}

void worstFit(int nb, int nf, int b[], int f[]){
    int ff[MAX] = {0};
int allocated[MAX] = {0};
for(int i=0; i<nf; i++){
    int worst = -1;
    ff[i] = -1;
    for(int j=0; j<nb; j++){
        if(allocated[j] == 0 && b[j] >= f[i]){
            if(worst == -1 || b[j] > b[worst])
                worst = j;
        }
    }
    if(worst != -1){
        ff[i] = worst;
        allocated[worst] = 1;
    }
}
printf("\nFile_no:\tFile_size:\tBlock_no:\t Block_size:");
for(int i=0; i<nf; i++){
    if(ff[i] != -1)
        printf("\n%d\t\t%d\t\t%d\t\t%d",
               i+1, f[i], ff[i]+1, b[ff[i]]);
    else
        printf("\n%d\t\t%d\t\t-\t\t-", i+1, f[i]);
}
}

int main()
{
    int nb, nf, choice;
    printf("Memory Management Scheme");
    printf("\nEnter the number of blocks:");
    scanf("%d", &nb);
    printf("Enter the number of files:");
    scanf("%d", &nf);
```

```
int b[nb], f[nf];
printf("\nEnter the size of the blocks:\n");
for(int i=0; i<nb; i++){
    printf("Block %d:", i+1);
    scanf("%d", &b[i]);
}

printf("Enter the size of the files:\n");
for(int i=0; i<nf; i++){
    printf("File %d:", i+1);
    scanf("%d", &f[i]);
}
while(1){
  printf("\n1. First Fit \n 2. Best Fit \n 3. Worst Fit
          \n 4. Exit \n");
  printf("Enter your choice:");
  scanf("%d", &choice);
  switch(choice){
     case 1:
          printf("\n\t Memory Management scheme - First Fit
                \n");
          firstFit(nb, nf, b, f);
          break;
     case 2:
          printf("\n\t Memory Management scheme - Best Fit\n");
          bestFit(nb, nf, b, f);
          break;
     case 3:
          printf("\n\t Memory Management scheme - Worst Fit\n");
          worstfit(nb, nf, b, f);
          break;
     case 4:
          printf("\n Exiting....\n");
```

```
        exit(0);
        break;
    default:
        printf("\n Invalid choice.\n");
        break;
    }
  }
  return 0;
}
```

Output:-

```
Memory Management Scheme
Enter the number of blocks : 5
Enter the number of files : 4
Enter the size of the blocks :
Block 1 : 100
Block 2 : 500
Block 3 : 200
Block 4 : 300
Block 5 : 600
Enter the size of the files :
File 1 : 123
File 2 : 323
File 3 : 523
File 4 : 50
1. First Fit
2. Best Fit
3. Worst Fit
4. Exit
Enter your choice : 1
    Memory Management Scheme - First Fit
File_no:     File_size:      Block_no:     Block_size
  1            123              2             500
```

```
2          323              5            600
3          523              -            ~~600~~ -
4          50               1            100

1. First Fit
2. Best Fit
3. Worst Fit
4. Exit
Enter your choice: 2
      Memory Management scheme - Best Fit
file_no:    File_size:      Block_no:     Block_size:
 1           123              3             200
 2           323              2             500
 3           523              5             600
 4           50               1             100

1. First Fit
2. Best Fit
3. Worst Fit
4. Exit
Enter your choice : 3
      Memory Management scheme - Worst Fit
File_no:    File_size:      Block_no:     Block_size:
 1           123              5             600
 2           323              2             500
 3           523              -             -
 4           50               4             300

1. First Fit
2. Best Fit
3. Worst Fit
4. Exit
Enter your choice: 4
Exiting .....
```

[illegible]
4/3/24