**LAB-7**

**Memory Management problem**

1. Write a C program 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 b[], int nb, int f[], int nf);

void worstFit(int b[], int nb, int f[], int nf);

void bestFit(int b[], int nb, int f[], int nf);

int main() {

int b[max], f[max], nb, nf;

printf("Memory Management Schemes\n");

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 (int i = 0; i < nb; i++) {

printf("Block %d: ", i + 1);

scanf("%d", &b[i]);

}

printf("\nEnter the size of the files:\n");

for (int i = 0; i < nf; i++) {

printf("File %d: ", i + 1);

scanf("%d", &f[i]);

}

printf("\nMemory Management Scheme - First Fit");

firstFit(b, nb, f, nf);

printf("\n\nMemory Management Scheme - Worst Fit");

worstFit(b, nb, f, nf);

printf("\n\nMemory Management Scheme - Best Fit");

bestFit(b, nb, f, nf);

return 0;

}

void firstFit(int b[], int nb, int f[], int nf) {

int bf[max] = {0}, ff[max], frag[max];

for (int i = 0; i < nf; i++) {

ff[i] = -1;

for (int j = 0; j < nb; j++) {

if (!bf[j] && b[j] >= f[i]) {

ff[i] = j;

frag[i] = b[j] - f[i];

bf[j] = 1;

break;

}

}

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (int i = 0; i < nf; i++) {

if (ff[i] != -1)

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i+1, f[i], ff[i]+1, b[ff[i]], frag[i]);

else

printf("\n%d\t\t%d\t\tNot Allocated", i+1, f[i]);

}

}

void worstFit(int b[], int nb, int f[], int nf) {

int tempBlocks[max];

for (int i = 0; i < nb; i++) tempBlocks[i] = b[i]

int bf[max] = {0}, ff[max], frag[max];

for (int i = 0; i < nf; i++) {

int index = -1, maxDiff = -1;

for (int j = 0; j < nb; j++) {

if (!bf[j] && tempBlocks[j] >= f[i] && (tempBlocks[j] - f[i] > maxDiff)) {

index = j;

maxDiff = tempBlocks[j] - f[i];

}

}

if (index != -1) {

ff[i] = index;

frag[i] = tempBlocks[index] - f[i];

bf[index] = 1;

} else {

ff[i] = -1;

}

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (int i = 0; i < nf; i++) {

if (ff[i] != -1)

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i+1, f[i], ff[i]+1, b[ff[i]], frag[i]);

else

printf("\n%d\t\t%d\t\tNot Allocated", i+1, f[i]);

}

}

void bestFit(int b[], int nb, int f[], int nf) {

int tempBlocks[max];

for (int i = 0; i < nb; i++) tempBlocks[i] = b[i];

int bf[max] = {0}, ff[max], frag[max];

for (int i = 0; i < nf; i++) {

int index = -1, minDiff = 100000;

for (int j = 0; j < nb; j++) {

if (!bf[j] && tempBlocks[j] >= f[i] && (tempBlocks[j] - f[i] < minDiff)) {

index = j;

minDiff = tempBlocks[j] - f[i];

}

}

if (index != -1) {

ff[i] = index;

frag[i] = tempBlocks[index] - f[i];

bf[index] = 1;

} else {

ff[i] = -1;

}

}

printf("\nFile\_no:\tFile\_size:\tBlock\_no:\tBlock\_size:\tFragment");

for (int i = 0; i < nf; i++) {

if (ff[i] != -1)

printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i+1, f[i], ff[i]+1, b[ff[i]], frag[i]);

else

printf("\n%d\t\t%d\t\tNot Allocated", i+1, f[i]);

}

}

OUTPUT:

