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

**a) Worst-fit**

**b) Best-fit**

**c) First-fit**

Program:

#include <stdio.h>

#include <string.h>

#define MAX\_BLOCKS 10

#define MAX\_PROCESSES 10

void resetAllocation(int allocation[], int blockSize[], int originalBlockSize[], int m) {

for (int i = 0; i < MAX\_PROCESSES; i++)

allocation[i] = -1;

for (int i = 0; i < m; i++)

blockSize[i] = originalBlockSize[i];

}

void firstFit(int blockSize[], int m, int processSize[], int n, int allocation[]) {

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

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

if (blockSize[j] >= processSize[i]) {

allocation[i] = j;

blockSize[j] -= processSize[i];

break;

}

}

}

}

void bestFit(int blockSize[], int m, int processSize[], int n, int allocation[]) {

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

int bestIdx = -1;

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

if (blockSize[j] >= processSize[i]) {

if (bestIdx == -1 || blockSize[j] < blockSize[bestIdx]) {

bestIdx = j;

}

}

}

if (bestIdx != -1) {

allocation[i] = bestIdx;

blockSize[bestIdx] -= processSize[i];

}

}

}

void worstFit(int blockSize[], int m, int processSize[], int n, int allocation[]) {

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

int worstIdx = -1;

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

if (blockSize[j] >= processSize[i]) {

if (worstIdx == -1 || blockSize[j] > blockSize[worstIdx]) {

worstIdx = j;

}

}

}

if (worstIdx != -1) {

allocation[i] = worstIdx;

blockSize[worstIdx] -= processSize[i];

}

}

}

void printResult(int processSize[], int allocation[], int n) {

printf("\nProcess No.\tProcess Size\tBlock No.\n");

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

printf("%d\t\t%d\t\t", i + 1, processSize[i]);

if (allocation[i] != -1)

printf("%d\n", allocation[i] + 1);

else

printf("Not Allocated\n");

}

}

int main() {

int blockSize[MAX\_BLOCKS], originalBlockSize[MAX\_BLOCKS];

int processSize[MAX\_PROCESSES], allocation[MAX\_PROCESSES];

int m, n, choice;

printf("Enter number of memory blocks: ");

scanf("%d", &m);

printf("Enter size of each memory block:\n");

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

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

originalBlockSize[i] = blockSize[i];

}

printf("Enter number of processes: ");

scanf("%d", &n);

printf("Enter size of each process:\n");

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

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

}

do {

printf("\nMemory Allocation Techniques:\n");

printf("1. First Fit\n");

printf("2. Best Fit\n");

printf("3. Worst Fit\n");

printf("4. Exit\n");

printf("Enter your choice (1-4): ");

scanf("%d", &choice);

resetAllocation(allocation, blockSize, originalBlockSize, m);

switch (choice) {

case 1:

firstFit(blockSize, m, processSize, n, allocation);

printf("\n--- First Fit Allocation ---\n");

printResult(processSize, allocation, n);

break;

case 2:

bestFit(blockSize, m, processSize, n, allocation);

printf("\n--- Best Fit Allocation ---\n");

printResult(processSize, allocation, n);

break;

case 3:

worstFit(blockSize, m, processSize, n, allocation);

printf("\n--- Worst Fit Allocation ---\n");

printResult(processSize, allocation, n);

break;

case 4:

printf("Exiting...\n");

break;

default:

printf("Invalid choice. Please select 1-4.\n");

}

} while (choice != 4);

return 0;

}

Output:



