Frist\_fit

#include <stdio.h>

#define MAX 10 // Maximum size for arrays

int main() {

int blockSize[MAX], processSize[MAX], allocation[MAX];

int blocks, processes;

// Input number of memory blocks

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

scanf("%d", &blocks);

// Input sizes of each memory block

printf("Enter sizes of %d memory blocks:\n", blocks);

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

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

// Input number of processes

printf("Enter number of processes: ");

scanf("%d", &processes);

// Input sizes of each process

printf("Enter sizes of %d processes:\n", processes);

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

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

// Step 1: Initialize allocation array to -1 (means not allocated)

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

allocation[i] = -1;

// Step 2: Apply First Fit Allocation

for (int i = 0; i < processes; i++) { // For each process

for (int j = 0; j < blocks; j++) { // Check all blocks

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

allocation[i] = j; // Allocate block j to process i

blockSize[j] -= processSize[i]; // Reduce block size

break; // Go to next process after successful allocation

}

}

}

// Step 3: Display results

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

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

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

if (allocation[i] != -1)

printf("%d\n", allocation[i] + 1); // Block numbers shown as 1-based index

else

printf("Not Allocated\n");

}

return 0;

}

START

1. Input number of memory blocks → blocks

2. Input sizes of all memory blocks → blockSize[]

3. Input number of processes → processes

4. Input sizes of all processes → processSize[]

5. Initialize allocation[] to -1 for all processes

6. For each process i:

a. For each memory block j:

i. If blockSize[j] ≥ processSize[i]:

→ allocation[i] = j

→ blockSize[j] = blockSize[j] - processSize[i]

→ Break inner loop (go to next process)

7. For each process:

a. Print process number, size, and allocated block number

b. If not allocated, print "Not Allocated"

END