

Write a c program to stimulate the following contiguous memory allocation techniques.

A) worst-fit

B) best-fit

C) First-fit

```
#include <stdio.h>
#define MAX 100

void allocate(int blocks[], int n, int procs[], int m, char *type) {
    int alloc[m], i, j, idx;
    for (i = 0; i < m; i++) alloc[i] = -1;

    for (i = 0; i < m; i++) {
        idx = -1;
        for (j = 0; j < n; j++) {
            if (blocks[j] >=
procs[i]) {
                if ((type[0] == 'F' && idx == -1) ||
                    (type[0] == 'B' && (idx == -1 || blocks[j] < blocks[idx])) ||
                    (type[0] == 'W' && (idx == -1 || blocks[j] > blocks[idx])))
                    idx = j;
            }
        }
        if (idx != -1) {
            alloc[i] = idx;
            blocks[idx] -= procs[i];
        }
    }

    printf("\n%s Fit:\n", type);
    for (i = 0; i < m; i++)
        printf("Process %d -> Block %d\n", i + 1, alloc[i] == -1 ? -1 : alloc[i] + 1);
}

int main() {
    int n, m, i, blocks[MAX], procs[MAX], b1[MAX], b2[MAX];

    printf("Enter number of blocks: "); scanf("%d", &n);
    printf("Enter block sizes: "); for (i = 0; i < n; i++) scanf("%d", &blocks[i]);

    printf("Enter number of processes: "); scanf("%d", &m);
    printf("Enter process sizes: "); for (i = 0; i < m; i++) scanf("%d", &procs[i]);

    for (i = 0; i < n; i++) b1[i] = b2[i] = blocks[i];
```

```
allocate(blocks, n, procs, m, "First");
allocate(b1, n, procs, m, "Best");
allocate(b2, n, procs, m, "Worst");

return 0;
}
```

OUTPUT:

```
Enter number of blocks: 5
Enter block sizes: 100 500 200 300 600
Enter number of processes: 4
Enter process sizes: 212 417 112 426

First Fit:
Process 1 -> Block 2
Process 2 -> Block 5
Process 3 -> Block 2
Process 4 -> Block -1

Best Fit:
Process 1 -> Block 4
Process 2 -> Block 2
Process 3 -> Block 3
Process 4 -> Block 5

Worst Fit:
Process 1 -> Block 5
Process 2 -> Block 2
Process 3 -> Block 5
Process 4 -> Block -1
```