```c
#include <stdio.h>
#define MAX 100
void firstFit(int blockSize[], int m, int processSize[], int n) {
    int allocation[MAX];
    for (int i = 0; i < n; i++)
        allocation[i] = -1;
    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;
            }
        }
    }
    printf("\nFirst-Fit Allocation:\n");
    printf("Process 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");
    }
}
void bestFit(int blockSize[], int m, int processSize[], int n) {
    int allocation[MAX];
    for (int i = 0; i < n; i++)
        allocation[i] = -1;
    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];
        }
```

```
        }
        if (bestIdx != -1) {
            allocation[i] = bestIdx;
            blockSize[bestIdx] -= processSize[i];
        }
    }
    printf("\nBest-Fit Allocation:\n");
    printf("Process 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");
    }
}
void worstFit(int blockSize[], int m, int processSize[], int n) {
    int allocation[MAX];
    for (int i = 0; i < n; i++)
        allocation[i] = -1;
    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];
        }
    }
    printf("\nWorst-Fit Allocation:\n");
    printf("Process 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("\nWorst-Fit Allocation:\n");
    printf("Process 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], processSize[MAX], m, n;
    printf("Enter number of memory blocks: ");
    scanf("%d", &m);
    printf("Enter size of each block:\n");
    for (int i = 0; i < m; i++) {
        printf("Block %d: ", i + 1);
        scanf("%d", &blockSize[i]);
    }
    printf("\nEnter number of processes: ");
    scanf("%d", &n);
    printf("Enter size of each process:\n");
    for (int i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        scanf("%d", &processSize[i]);
    }
    int blockSize1[MAX], blockSize2[MAX], blockSize3[MAX];
    for (int i = 0; i < m; i++) {
        blockSize1[i] = blockSize[i];
        blockSize2[i] = blockSize[i];
        blockSize3[i] = blockSize[i];
    }
    firstFit(blockSize1, m, processSize, n);
    bestFit(blockSize2, m, processSize, n);
    worstFit(blockSize3, m, processSize, n);
    return 0;
}
```

```
Enter number of memory blocks: 5
Enter size of each block:
Block 1: 400
Block 2: 700
Block 3: 200
Block 4: 300
Block 5: 600

Enter number of processes: 4
Enter size of each process:
Process 1: 212
Process 2: 517
Process 3: 312
Process 4: 526

First-Fit Allocation:
Process No.     Process Size    Block No.
1               212             1
2               517             2
3               312             5
4               526             Not Allocated

Best-Fit Allocation:
Process No.     Process Size    Block No.
1               212             4
2               517             5
3               312             1
4               526             2

Worst-Fit Allocation:
Process No.     Process Size    Block No.
1               212             2
2               517             5
3               312             2
4               526             Not Allocated

Process returned 0 (0x0)   execution time : 28.651 s
Press any key to continue.
```