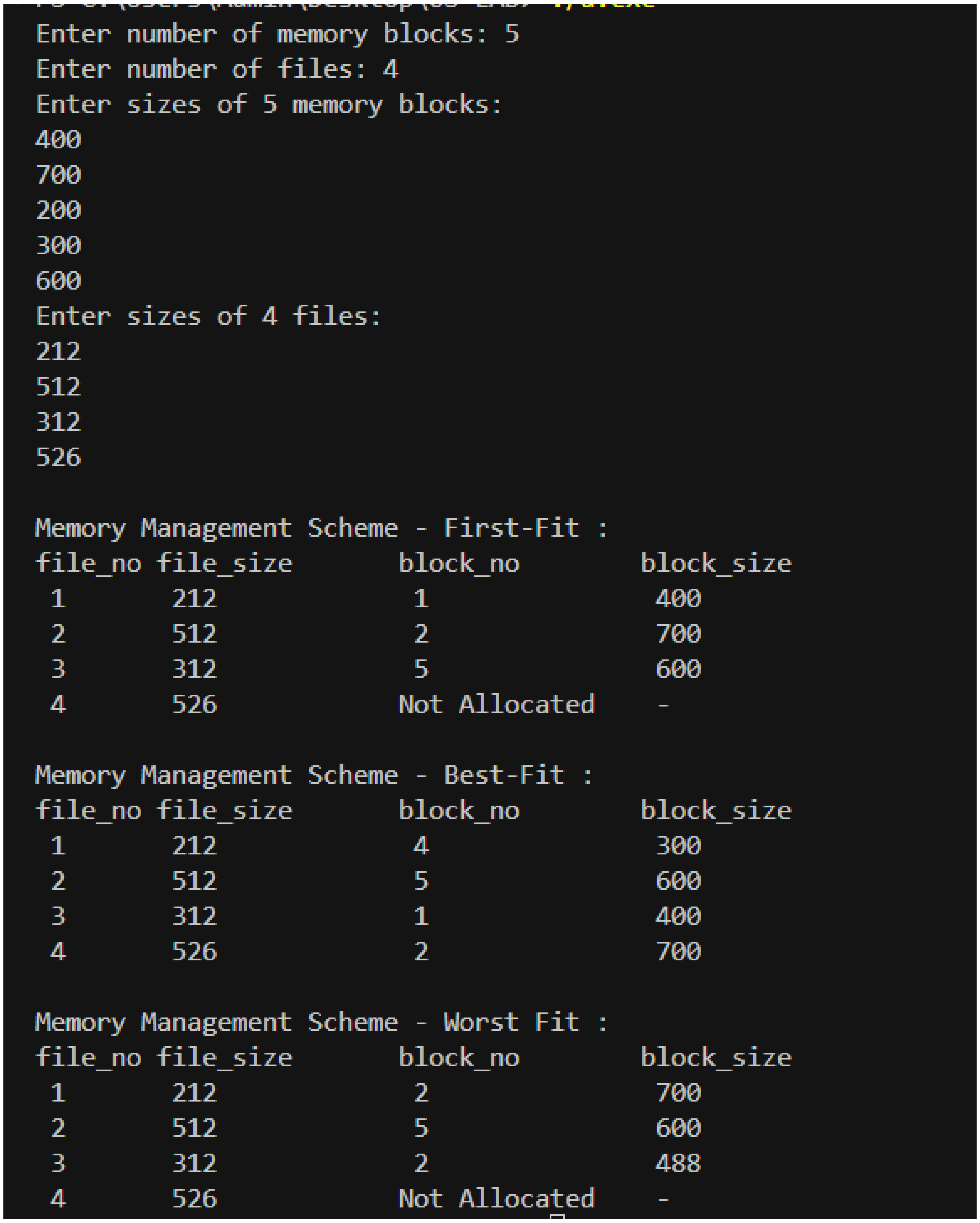
Write a C program to simulate the following contiguous memory allocation techniques. (Any one)
a) Worst-fit
b) Best-fit
c) First-fit

```
#include <stdio.h>
#define MAX 10

void firstFit(int blockSize[], int m, int processSize[], int n) {
    int allocation[MAX], usedBlockSize[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;
                usedBlockSize[i] = blockSize[j];
                blockSize[j] -= processSize[i];
                break;
                }
        }
    }
    printf("\nMemory Management Scheme - First-Fit :\n");
    printf("file_no\tfile_size\tblock_no\tblock_size\n");
    for (int i = 0; i < n; i++) {
        printf(" %d\t %d\t\t", i + 1, processSize[i]);
        if (allocation[i] != -1)
            printf(" %d\t\t %d\n", allocation[i] + 1, usedBlockSize[i]);
        else
            printf("Not Allocated\t -\n");
    }
}
void bestFit(int blockSize[], int m, int processSize[], int n) {
int allocation[MAX], usedBlockSize[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;
    usedBlockSize[i] = blockSize[bestIdx];
    blockSize[bestIdx] -= processSize[i];
}
}
printf("\nMemory Management Scheme - Best-Fit :\n");
printf("file_no\tfile_size\tblock_no\tblock_size\n");
for (int i = 0; i < n; i++) {
    printf(" %d\t %d\t\t", i + 1, processSize[i]);
if (allocation[i] != -1)
    printf(" %d\t\t %d\n", allocation[i] + 1, usedBlockSize[i]);
else
        printf("Not Allocated\t -\n");
}
}
void worstFit(int blockSize[], int m, int processSize[], int n) {
int allocation[MAX], usedBlockSize[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;
        usedBlockSize[i] = blockSize[worstIdx];
        blockSize[worstIdx] -= processSize[i];
    }
}
```

```
printf("\nMemory Management Scheme - Worst Fit :\n");
printf("file_no\tfile_size\tblock_no\tblock_size\n");
for (int i = 0; i < n; i++) {
    printf(" %d\t %d\t\t", i + 1, processSize[i]);
    if (allocation[i] != -1)
        printf(" %d\t\t %d\n", allocation[i] + 1, usedBlockSize[i]);
    else
        printf("Not Allocated\t -\n");
}
}
int main() {
    int m, n;
    int blockSize1[MAX], blockSize2[MAX], blockSize3[MAX], processSize[MAX];
    printf("Enter number of memory blocks: ");
    scanf("%d", &m);
    printf("Enter number of files: ");
    scanf("%d", &n);
    printf("Enter sizes of %d memory blocks:\n", m);
    for (int i = 0; i < m; i++) {
        scanf("%d", &blockSize1[i]);
    }
    printf("Enter sizes of %d files:\n", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &processSize[i]);
    }
    for (int i = 0; i < m; i++) {
        blockSize2[i] = blockSize1[i];
        blockSize3[i] = blockSize1[i];
    }
    firstFit(blockSize1, m, processSize, n);
    bestFit(blockSize2, m, processSize, n);
    worstFit(blockSize3, m, processSize, n);
    return 0;
}
```

Output:



```
Enter number of memory blocks: 5
Enter number of files: 4
Enter sizes of 5 memory blocks:
400
700
200
300
600
Enter sizes of 4 files:
212
512
312
526

Memory Management Scheme - First-Fit :
file_no file_size       block_no        block_size
 1       212             1               400
 2       512             2               700
 3       312             5               600
 4       526            Not Allocated    -

Memory Management Scheme - Best-Fit :
file_no file_size       block_no        block_size
 1       212             4               300
 2       512             5               600
 3       312             1               400
 4       526             2               700

Memory Management Scheme - Worst Fit :
file_no file_size       block_no        block_size
 1       212             2               700
 2       512             5               600
 3       312             2               488
 4       526            Not Allocated    -
```