

main.c

Share

Run

```
1 #include <stdio.h>
2 #define MAX 100
3 void bestFit(int blockSize[], int m, int processSize[], int n) {
4     int allocation[n];
5     for (int i = 0; i < n; i++) allocation[i] = -1;
6     for (int i = 0; i < n; i++) {
7         int bestIdx = -1;
8         for (int j = 0; j < m; j++) {
9             if (blockSize[j] >= processSize[i]) {
10                 if (bestIdx == -1 || blockSize[bestIdx] >
11                     blockSize[j]) {
12                     bestIdx = j;
13                 }
14             }
15             if (bestIdx != -1) {
16                 allocation[i] = bestIdx;
17                 blockSize[bestIdx] -= processSize[i];
18             }
19         }
20         printf("Process No.\tBlock No.\n");
21         for (int i = 0; i < n; i++) {
22             printf(" %d\t\t", i + 1);
23             if (allocation[i] != -1) {
24                 printf("%d\n", allocation[i] + 1);
25             } else {
```

Output

Process No. Block No.  
1 4  
2 2  
3 3  
4 5  
  
=== Code Execution Successful ===



main.c



Share

Run

Output

Clear

```
1 #include <stdio.h>
2 void worstFit(int blockSize[], int m, int processSize[], int n) {
3     int allocation[n];
4     for (int i = 0; i < n; i++) allocation[i] = -1;
5     for (int i = 0; i < n; i++) {
6         int worstIdx = -1;
7         for (int j = 0; j < m; j++) {
8             if (blockSize[j] >= processSize[i]) {
9                 if (worstIdx == -1 || blockSize[worstIdx] <
10                     blockSize[j])
11                     worstIdx = j;
12             }
13         }
14         if (worstIdx != -1) {
15             allocation[i] = worstIdx;
16             blockSize[worstIdx] -= processSize[i];
17         }
18         printf("Process No.\tBlock No.\n");
19         for (int i = 0; i < n; i++)
20             printf(" %d\t\t%d\n", i + 1, allocation[i] + 1);
21     }
22 int main() {
23     int blockSize[] = {100, 500, 200, 300, 600};
24     int processSize[] = {212, 417, 112, 426};
25     int m = sizeof(blockSize) / sizeof(blockSize[0]);
26     int n = sizeof(processSize) / sizeof(processSize[0]);
27     worstFit(blockSize, m, processSize, n);
28 }
```

Process No. Block No.

1	5
2	2
3	5
4	0

=== Code Execution Successful ===