Program: Best Fit Memory Management

```
// C++ implementation of Best - Fit algorithm
#include<iostream>
using namespace std;
// Method to allocate memory to blocks as per Best fit algorithm
void bestFit(int blockSize[], int m, int processSize[], int n)
{
// Stores block id of the block allocated to a process
int allocation[n];
// Initially no block is assigned to any process
for (int i = 0; i < n; i++)
allocation[i] = -1;
// pick each process and find suitable blocks
// according to its size ad assign to it
for (int i = 0; i < n; i++)
{
// Find the best fit block for current process
int bestIdx = -1;
for (int j = 0; j < m; j++)
{
if (blockSize[j] >= processSize[i])
{
if (bestIdx == -1)
bestIdx = j;
else if (blockSize[bestIdx] > blockSize[j])
bestIdx = j;
```

```
}
}
// If we could find a block for current process
if (bestIdx != -1)
{
// allocate block j to p[i] process
allocation[i] = bestIdx;
// Reduce available memory in this block.
blockSize[bestIdx] -= processSize[i];
}
}
cout << "\nProcess No.\tProcess Size\tBlock no.\n";
for (int i = 0; i < n; i++)
{
cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
if (allocation[i] != -1)
cout << allocation[i] + 1;
else
cout << "Not Allocated";
cout << endl;
}
}
// Driver Method
int main()
{
int blockSize[] = {100, 500, 200, 300, 600};
int processSize[] = {212, 417, 112, 426};
```

```
int m = sizeof(blockSize) / sizeof(blockSize[0]);

int n = sizeof(processSize) / sizeof(processSize[0]);

bestFit(blockSize, m, processSize, n);

return 0 ;

}
```

OUTPUT:

Process No. Process Size Block no.

1 212 4

2 417 2

3 112 3

4 426 5