```java
package ass6bestfit;
import java.util.Scanner;

public class ass6bestfit {
 static void bestFit(int blockSize[], int m, int processSize[], int n, int remblockSize[]) {
  int allocation[] = new int[n];
  for (int i = 0; i < allocation.length; 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)
      bestIdx = j;
     else if (blockSize[bestIdx] > blockSize[j])
      bestIdx = j;
    }
   }
   if (bestIdx != -1) {
    allocation[i] = bestIdx;
    blockSize[bestIdx] -= processSize[i];
    remblockSize[i] = blockSize[bestIdx];
   }
  }

  System.out.println("\nProcess No.\tProcess Size\tBlock no.\tRemaninig Block Size");
  for (int i = 0; i < n; i++) {
   System.out.print(" " + (i + 1) + "\t\t" + processSize[i] + "\t\t");
   if (allocation[i] != -1) {
    System.out.print((allocation[i] + 1) + "\t\t" + remblockSize[i]);
   } else {
    System.out.print("Not Allocated" + "\t" + remblockSize[i]);
   }
   System.out.println();
  }
 }

 public static void main(String[] args) {
  int m, n, num;
  Scanner in = new Scanner(System.in);
  System.out.print("Enter how many number of blocks you want to enter:");
  m = in.nextInt();
  int remblockSize[] = new int[m];
  int blockSize[] = new int[m];
  for (int i = 0; i < m; i++) {
   System.out.print("Enter Data " + (i + 1) + ":");
   num = in.nextInt();
   blockSize[i] = num;
  }
  System.out.print("Enter how many number of process you want to enter:");
  n = in.nextInt();
  int processSize[] = new int[n];
  for (int i = 0; i < n; i++) {
   System.out.print("Enter Data " + (i + 1) + ":");
```

```
    num = in.nextInt();
    processSize[i] = num;
   }
  bestFit(blockSize, m, processSize, n, remblockSize);

   in.close();
  }

}
/*
 * Enter how many number of blocks you want to enter:5
Enter Data 1:100
Enter Data 2:500
Enter Data 3:200
Enter Data 4:300
Enter Data 5:600
Enter how many number of process you want to enter:4
Enter Data 1:212
Enter Data 2:417
Enter Data 3:112
Enter Data 4:426

Process No. Process Size Block no. Remaninig Block Size
 1  212  4  88
 2  417  2  83
 3  112  3  88
 4  426  5  174

 First fit = Allocate the first hole that is big enough
 second fit = same as first page but start search always from last allocation hole
 best fit = allocate the smallest bone that is big enough
 worst fit = allocate the largest hole
*/
```