```
package ass6nextfit;
import java.util.Arrays;
import java.util.Scanner;

public class ass6nextfit {
  static void NextFit(int blockSize[], int m, int processSize[], int n, int remblockSize[]) {

   int allocation[] = new int[n], j = 0;
   Arrays.fill(allocation, -1);
   for (int i = 0; i < n; i++) {
    int count = 0;
    while (count < m) {
     count++;
     if (blockSize[j] >= processSize[i]) {
      allocation[i] = j;
      blockSize[j] -= processSize[i];
      remblockSize[i] = blockSize[j];
      break;
     }
     j = (j + 1) % m;
     count += 1;
    }
   }

   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 blockSize[] = new int[m];
   int remblockSize[] = 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;
```

```
    }
    NextFit(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
*/
```