**Program :**

import java.util.Scanner;

public class BestFit{

    static void bestFit(int blockSize[], int m, int processSize[], int n){

        int allocation[] = new int[n]; // Stores block id of the block allocated to a process

        for (int i = 0; i < allocation.length; i++) // Initially no block is assigned to any process

        allocation[i] = -1;

        System.out.println("\nProcess no.\tProcess Size\tBlock no.\tBlock size remaining");

        for (int i=0; i<n; i++){

            int bestIdx = -1;  // Find the best fit block for current process

            boolean allocated = false;

            for (int j=0; j<m; j++){

                if (blockSize[j] >= processSize[i]){

                    if (allocated == false){ //to assign block first time

                        bestIdx = j;

                        allocated=true;

                    }

                    else if (blockSize[bestIdx] > blockSize[j])

                        bestIdx = j;

                }

            }

            if (allocated == true){      //if allocated update allocation list and reduce corresp block size

                allocation[i] = bestIdx; // allocate block j to p[i] process

                blockSize[bestIdx] -= processSize[i]; // Reduce available memory in this block.

            }

            System.out.print(" " + (i+1) + "\t\t" + processSize[i] + "\t\t" );//printing output

            if (allocation[i] != -1)

                System.out.print(allocation[i] + 1 + "\t\t" + blockSize[allocation[i]]);

            else

                System.out.print("Not Allocated");

            System.out.println();

        }

    }

    public static void main(String[] args){

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the number of processes : ");

        int n = sc.nextInt();

        int processSize[] = new int[n];

        System.out.println("Enter the processes : ");

        for(int i=0;i<n;i++){

            System.out.print("Process No:"+ (i+1) + " =");

            processSize[i]=sc.nextInt();

        }

        System.out.print("\nEnter the number of memory blocks : ");

        int m = sc.nextInt();

        int blockSize[] = new int[m];

        System.out.println("Enter the memory blocks : ");

        for(int i=0;i<m;i++){

            System.out.print("Block No:"+ (i+1) + " =");

            blockSize[i]=sc.nextInt();

        }

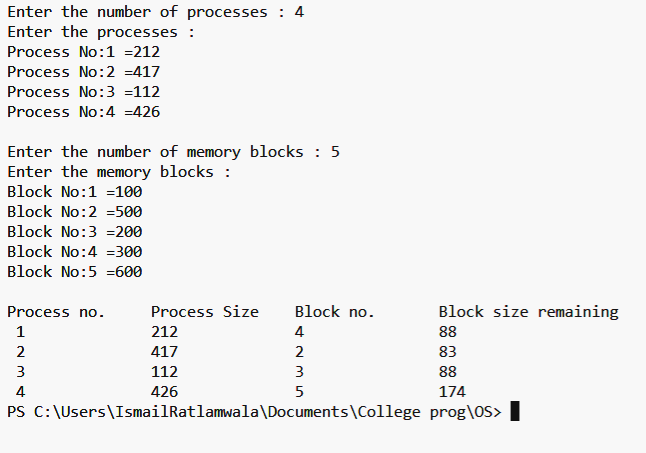
        bestFit(blockSize, m, processSize, n);

        sc.close();

    }

}

**Output :**

****

**Program :**

#include <bits/stdc++.h>

using namespace std;

int main()

{

    vector<pair<int,bool>> request;

    cout<<"Enter number of request : ";

    int n,inp;

    cin>>n;

    cout<<"Enter the requests :"<<endl;

    for(int i=0;i<n;i++) {

        cin>>inp;

        request.push\_back({inp,false});

    }

    cout<<"Enter the position of head : ";

    int head;

    cin>>head;

    int totalMov=0;

    cout<<"Movement of head : \n"<<head;

    for(int j=0;j<n;j++){

        int minDiff=INT\_MAX, indx;

        for(int i=0;i<n;i++){

            if(!request[i].second && (abs(head-request[i].first)<minDiff)){

                minDiff=abs(head-request[i].first);

                indx=i;

            }

        }

        cout<<" => "<<request[indx].first;

        head=request[indx].first;

        request[indx].second=true;

        totalMov += minDiff ;

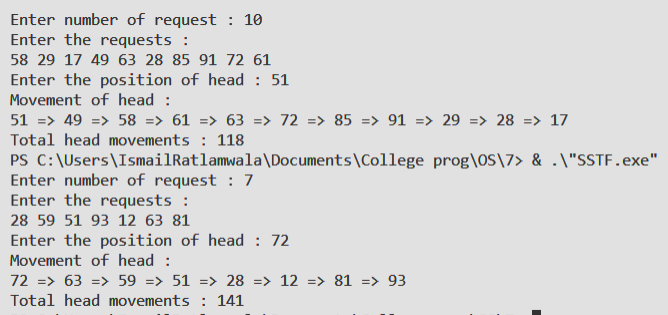
    }

    cout<<"\nTotal head movements : "<<totalMov<<endl;

    return 0;

}

**Output :**



**Program :**

#include <bits/stdc++.h>

using namespace std;

int main()

{

    vector<int> request;

    cout<<"Enter the size of disk : ";

    int size;

    cin>>size;

    size--;

    cout<<"Enter number of request : ";

    int n;

    cin>>n;

    cout<<"Enter the requests :"<<endl;

    for(int i=0;i<n;i++) {

        int inp;

        cin>>inp;

        request.push\_back(inp);

    }

    cout<<"Enter the position of head : ";

    int head;

    cin>>head;

    bool fromLeft=false;

    sort(request.begin(),request.end());

    int cutoffInd;

    for(cutoffInd=0; cutoffInd<n; cutoffInd++)

        if(request[cutoffInd]>head) break;

    cout<<"Movement of head : \n"<<head;

    if(fromLeft){

        for(int i=cutoffInd-1;i>=0;i--) cout<<" => "<<request[i];

        for(int i=cutoffInd; i<n; i++) cout<<" => "<<request[i];

        cout<<"\nTotal head movements : "<<request[n-1]+head;

    }

    else{

        for(int i=cutoffInd; i<n; i++) cout<<" => "<<request[i];

        cout<<" => "<<size;

        for(int i=cutoffInd-1;i>=0;i--) cout<<" => "<<request[i];

        cout<<"\nTotal head movements : "<<2\*size-head-request[0];

    }

    return 0;

}

**Output :**

