Exp: 6

```
import java.io.*;

import java.util.*;


public class MemoryAllocationAlgo {


        static int job[];

        static int block[];

        static int js,bs;

        static Scanner input=new Scanner(System.in);

        static int Allocation[];

        public static void main(String args[])

        {

                MemoryAllocationAlgo MA=new MemoryAllocationAlgo();

                while(true)

                {

                System.out.println("Menu:");

                System.out.println("\n1.Read Data-Job No. & Size, Block No. & Size \n2.First Fit 
\n3.Best Fit \n4.WorstFit\n5.Exit");

          System.out.println("Enter Your Choice:");

                int ch=Integer.parseInt(input.nextLine());

                switch(ch)

                {

                        case 1: System.out.println("\n Enter total no. of jobs to allocate:");

                                        js=Integer.parseInt(input.nextLine());

                                        System.out.println("\n Enter total no. of Free blocks:");

                                        bs=Integer.parseInt(input.nextLine());

                                        job=new int[js];

                                        block=new int[bs];

                                  MA.ReadData(js,bs);

                                 break;
```

```
                                case 2:

                                        MA.FirstFit();

                                        break;

                                case 3:MA.BestFit();

                                        break;

                                case 4:MA.WorstFit();

                                        break;

                                case 5:System.exit(0);

                                        break;

                        }//end of swith

                }//enf of while


        }//end of main

        void ReadData(int n,int m)

        {

                for(int i=0;i<n;i++)

                {

                  System.out.println("Enter Size of Job "+i+" :");

                        job[i]=Integer.parseInt(input.nextLine());

                }

                      for(int i=0;i<m;i++)

                {

                            System.out.println("Enter Size of FREE Block "+i+" :");

                             block[i]=Integer.parseInt(input.nextLine());

                }


        }

        void FirstFit()

        {   int flag=0;

              Allocation=new int[js];

              for (int i = 0; i < Allocation.length; i++)
        Allocation[i] = -1;
            for (int i = 0; i < js; i++)
    {
        for (int j = 0; j < bs; j++)
        {   flag=0;
            if (block[j] >=job[i])
            {  //System.out.println("i="+i+" j="+j+" B="+block[j]+" J="+job[i]+" all="+Allocation[i]);
               for(int k=0;k<js;k++)
               {
                        if(Allocation[k]==j)
                                flag=1;
               }
               // allocate block j to p[i] process
               if(flag==0)
               {  Allocation[i] = j;
                  //System.out.println(j+" B="+block[j]+" J="+job[i]+" all="+Allocation[i]);
                  break;
               }
            }
        }
    }
            Display();
}

void Display()
{
            System.out.println("Job No.\tJobSize  \tBlock No\tFragment");
            for(int i=0;i<js;i++)
            {
                        System.out.print(" "+i+"\t  "+job[i]+"\t ");
              if(Allocation[i]!=-1)
```

```
          {
                System.out.print("\t"+Allocation[i]+"\t"+(block[Allocation[i]]-job[i]));
          }
          else
          {
                System.out.println(" Not allocated");
          }
          System.out.println();

        }
}

void BestFit()
{
        int flag=0;
        Allocation=new int[js];
        for (int i = 0; i < Allocation.length; i++)
    Allocation[i] = -1;
        for (int i = 0; i < js; i++)
      {  int BestInd=-1;
       for (int j = 0; j < bs; j++)
         {  flag=0;
           if (block[j] >=job[i])
            {
             for(int k=0;k<js; k++)
              {
               if(Allocation[k]==j)
                   { flag=1;
                 break;
                }
              }
             // allocate block j to p[i] process
            // if(flag==1)
            //{
                   //  break;
            //}
                if(BestInd==-1 && flag==0)
                 {
                      BestInd=j;
                 }
                else if(flag==0 && block[BestInd]>block[j])
                {
                     BestInd=j;
                }
               else
               {
                     continue;
               }
          }
        }
        if(BestInd!=-1)
        {
            Allocation[i]=BestInd;
        }
      }
            Display();

}



void WorstFit()
{
      int flag=0;
       Allocation=new int[js];
       for (int i = 0; i < Allocation.length; i++)
Allocation[i] = -1;
       for (int i = 0; i < js; i++)
  {  int WorstInd=-1;
     for (int j = 0; j < bs; j++)
       {  flag=0;
         if (block[j] >=job[i])
          {
           for(int k=0;k<js; k++)
            {
             if(Allocation[k]==j)
                  { flag=1;
                break;
              }
            }
       // allocate block j to p[i] process
      // if(flag==1)
       //{
              //  break;
       //}
          if(WorstInd==-1 && flag==0)
           {
               WorstInd=j;
           }
          else if(flag==0 && block[WorstInd]<block[j])
          {
              WorstInd=j;
          }
          else
```

```
                {
                    continue;
                }
            }
        }
        if(WorstInd!=-1)
        {
            Allocation[i]=WorstInd;
        }
    }
            Display();
 }

}
```

Output:

```
Menu:

1.Read Data-Job No. & Size, Block No. & Size
2.First Fit
3.Best Fit
4.WorstFit
5.Exit
Enter Your Choice:
1

 Enter total no. of jobs to allocate:
4

 Enter total no. of Free blocks:
5
Enter Size of Job 0 :
100
Enter Size of Job 1 :
500
Enter Size of Job 2 :
200
Enter Size of Job 3 :
300
Enter Size of FREE Block 0 :
600
Enter Size of FREE Block 1 :
212
Enter Size of FREE Block 2 :
```

```
417
Enter Size of FREE Block 3 :
112
Enter Size of FREE Block 4 :
426
Menu:

1.Read Data-Job No. & Size, Block No. & Size
2.First Fit
3.Best Fit
4.WorstFit
5.Exit
Enter Your Choice:
2
Job No.      JobSize      Block No     Fragment
 0       100        0      500
 1       500   Not allocated

 2       200        1      12
 3       300        2      117
Menu:

1.Read Data-Job No. & Size, Block No. & Size
2.First Fit
3.Best Fit
4.WorstFit
5.Exit
Enter Your Choice:
3
Job No.      JobSize      Block No     Fragment
 0       100        3      12
 1       500        0      100
 2       200        1      12
 3       300        2      117
Menu:

1.Read Data-Job No. & Size, Block No. & Size
2.First Fit
3.Best Fit
4.WorstFit
5.Exit
Enter Your Choice:
4
Job No.      JobSize      Block No     Fragment
 0       100        0      500
 1       500   Not allocated

 2       200        4      226
 3       300        2      117
```