

```

1
2  /**
3   A PROGRAM TO INPUT THE SIZE OF A DOUBLE DIMENSIONAL ARRAY (m * n ).
4   i.e, m & n > 2 but m & n < 20.
5   ARRANGE THE ARRAY ELEMENTS IN ACENDING ORDER AND PRINT THE BOUNDARY
6   ELEMENTS ONLY
7   IN A DOUBLE DIMENSIONAL FORM.
8   */
9   //SUCHIT TE XII A
10  import java.util.*;
11  public class DDABoundary
12  {
13      public static void main()
14      {
15          Scanner sc = new Scanner (System.in);
16          System.out.println("A PROGRAM TO FIND THE SUM AND PRINT THE BO
17          UNDARY ELEMENTS OF A DOUBLE DIMENSIONAL ARRAY");
18          System.out.println("PLEASE ENTER THE SIZE OF THE DDA SUCH THAT
19          THE no.of ROWS AND no.of COLUMNS SHOULD BE");
20          System.out.println("GREATER THAN 2 BUT LESSER THAN 20");
21          System.out.println("ENTER ROW SIZE : ");
22          int row = sc.nextInt();
23          System.out.println("ENTER COLUMN SIZE : ");
24          int column = sc.nextInt();
25          if((row<=2 || row>=20) || (column<=2 || column>=20))
26          {
27              System.out.println("INVALID INPUT!");
28              System.exit(0);
29          }
30          int arr[][]=new int[row][column];
31          int temp[]=new int[row*column];
32          int ctrl = 0;
33          //INPUT FROM USER
34          System.out.println("ENTER THE ELEMENTS OF THE ARRAY");
35          for(int i=0;i<row;i++)
36          {
37              System.out.println("FOR ROW "+(i+1));
38              for(int j=0;j<column;j++)
39              {
40                  arr[i][j]=sc.nextInt();
41                  temp[ctrl]=arr[i][j];    // STORING DDA IN SINGLE DIMEN
42                  ctrl++;
43              }
44          }
45          //SORTING IN ASCENDING ORDER.
46          for(int i=0;i<temp.length-1;i++)
47          {
48              for(int j=(1+i);j<temp.length;j++)
49              {
50                  if(temp[j]<temp[i])
51                  {
52                      int hold=temp[i];
53                      temp[i]=temp[j];

```

```
51         temp[j]=hold;
52     }
53 }
54 }
55 //PRINT ORIGINAL ARRAY
56 System.out.println("ORIGINAL MATRIX");
57 for(int i=0;i<row;i++)
58 {
59     for(int j=0;j<column;j++)
60     {
61         System.out.print(arr[i][j]+" ");
62     }
63     System.out.println();
64 }
65 //TO REPLACE OLD DDA ELEMENTS IN ASCENDING ORDER
66 int ctr2=0;
67 for(int i=0;i<row;i++)
68 {
69     for(int j=0;j<column;j++)
70     {
71         arr[i][j]=temp[ctr2];
72         ctr2++;
73     }
74     System.out.println();
75 }
76 //PRINT ARRAY WITH ELEMENTS IN ASCENDING ORDER
77 System.out.println("NEW MATRIX");
78 for(int i=0;i<row;i++)
79 {
80     for(int j=0;j<column;j++)
81     {
82         System.out.print(arr[i][j]+" ");
83         System.out.print("\t");
84     }
85     System.out.println();
86 }
87 int finale[][]=new int[row][column];
88 int sumBoundary=0;
89 //TO PRINT AND FIND THE SUM OF BOUNDARY ELEMENTS
90 System.out.println("BOUNDARY ELEMENTS");
91 for(int i=0;i<row;i++)
92 {
93     for(int j=0;j<column;j++)
94     {
95         if(i==0 || i==(row-1) || j==0 || j==(column-1))
96         {
97             System.out.print(arr[i][j]+" ");
98             System.out.print("\t");
99             sumBoundary=sumBoundary+arr[i][j];
100         }
101         else
102         {
103             System.out.print(" ");
104         }
```

```
105         }
106         System.out.println();
107     }
108     System.out.println("THE SUM OF THE BOUNDARY ELEMENTS ARE : "+s
umBoundary);
109     }
110 }
111
```