18. (b) Test 18

Test Summary

- No. of Sections: 2
- No. of Questions: 3
- Total Duration: 45 min

Section 1 - Coding Proficiency

Section Summary

- No. of Questions: 2
- Duration: 30 min

Additional Instructions:

None

Q1. **Remove Duplicates**

Write a program to remove duplicate elements in an array
Ip: {1,1,1,1,2,2,2,3,3,4,5}

Op: {1,2,3,4,}

Input Format

Input contains array size and values

Output Format

Print the unique elemnts separated by dpace

Constraints

1<=array_size<=1000

Sample Input

```
10
12 45 10 23 45 10 55 67 6 9
```

Sample Output

```
12 45 10 23 55 67 6 9
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. **Matrix Rotation**

An image is representing m*n matrix of integers, where each integer represents a pixel value.Write an algorithm to rotate an image by 90 degree left or right according to the value of flag variable. If r=the flag value is 0, then rotate to the left and if flag value is 1 , then rotate to the right

Case 1:
Flag = 1
Input:
2 3 1
4 6 3
5 4 2
Output: 5 4 2
4 6 3
2 3 1
Case 2:
Flag = 0
Input:
2 1
3 4
Output:
4 1
3 2

Input Format

The input to the method consist of four arguments
img , a matrix of integers representing the pixels of the image
rows, an integer representing the no of rows(m)
columns, an integer representing the no of columns(n)
flag , an integer representing the rotation of the image

Output Format

Return a matrix of integers representing the pixels of the image rotated according to the value of the flag variable

Constraints

1 ≤ array_size ≤ 1000

Sample Input

```
3 3 1
2 3 1
4 6 3
5 4 2
```

Sample Output

```
5 4 2
4 6 3
2 3 1
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Section 2 - Essay Writing

Section Summary

- No. of Questions: 1
- Duration: 15 min

Additional Instructions:

None

Q1. **Essay Writing**

Government investments should be towards technologies or people standard of living

Directions

Write an essay for the given question

Keywords

Answer Key & Solution

Section 1 - Coding Proficiency

Q1

Test Case

Input

```
20
12 45 10 23 45 10 55 67 6 9 78 54 29 4 45 6 9 1
```

Output

```
12 45 10 23 55 67 6 9 78 54 29 4
```

Weightage - 5

Input

```
175
615 274 122 422 720 823 81 171 810 276 814 276 7
```

Output

```
615 274 122 422 720 823 81 171 810 276 814 77 55
```

Weightage - 10

Input

```
210 786 174 408 877 626 117 809 865 905 570 685
```

Output

```
786 174 408 877 626 117 809 865 905 570 685 785
```

Weightage - 10

Input

```
726 137 62 665 392 453 9 261 185 522 490 355 103
```

Output

```
137 62 665 392 453 9 261 185 522 490 355 103 430
```

Weightage - 10

Input

```
35 541 954 618 971 193 201 747 849 986 580 248 5
```

Output

```
541 954 618 971 193 201 747 849 986 580 248 568
```

Weightage - 5

Input

```
753 716 253 801 464 588 769 330 505 692 69 366 7
```

Output

```
716 253 801 464 588 769 330 505 692 69 366 786 1
```

Weightage - 10

Input

```
258 200 565 129 539 621 663 226 578 29 201 512 5
```

Output

```
200 565 129 539 621 663 226 578 29 201 512 588 5
```

Weightage - 10

Input

```
442 389 806 1 936 362 330 53 13 80 116 149 697 8
```

Output

```
389 806 1 936 362 330 53 13 80 116 149 697 884 2
```

Weightage - 10

Input

```
489 181 610 646 894 646 608 783 711 450 83 586 1
```

Output

```
181 610 646 894 608 783 711 450 83 586 172 851 9
```

Weightage - 10

Input

```
401 291 632 618 653 445 752 500 936 593 270 743
```

Output

```
291 632 618 653 445 752 500 936 593 270 743 548
```

Weightage - 10

Input

```
509 865 335 62 670 955 325 805 814 490 180 945 5
```

Output

```
865 335 62 670 955 325 805 814 490 180 945 501 2
```

Weightage - 10

Sample Input

```
10
12 45 10 23 45 10 55 67 6 9
```

Sample Output

```
12 45 10 23 55 67 6 9
```

Solution

```
#include<stdio.h>
int main()
{
    int arr[1000],index,search,updatepos = 0,size;
    scanf("%d",&size);
    for(index=0 ; index<size ; index++)
        scanf("%d",&arr[index]);
    for(index = 0 ; index < size  ; index++)
    {
        if(arr[index] != -1)
        {
            for(search = index+1 ; search < size ; search++)
            {
                    if(arr[search]  == arr[index])
                        arr[search] = -1;
            }
            arr[updatepos++] = arr[index];

            if(updatepos-1 != index)
```

```
                arr[index] = -1;
        }
    }

     for(index = 0 ; index < updatepos ; index++)
          printf("%d ",arr[index]);

    return 0;
}
```

```
#include<stdio.h>
int main()
{
    int arr[1000],index,search,updatepos = 0,size;
    scanf("%d",&size);
    for(index=0 ; index<size ; index++)
        scanf("%d",&arr[index]);
    for(index = 0 ; index < size  ; index++)
    {
        if(arr[index] != -1)
        {
            for(search = index+1 ; search < size ; search++)
            {
                    if(arr[search]  == arr[index])
                        arr[search] = -1;
            }
            arr[updatepos++] = arr[index];
            if(updatepos-1 != index)
                 arr[index] = -1;
        }
    }

     for(index = 0 ; index < updatepos ; index++)
          printf("%d ",arr[index]);

    return 0;
}
```

Q2

Test Case

Input

```
2 2 0
2 1
3 4
```

Output

```
1 4
2 3
```

Weightage - 5

Input

```
4 4 0
1 2 3 4
5 6 7 8
9 10 11 12
```

Output

```
4 8 12 16
3 7 11 15
2 6 10 14
1 5 9 13
```

Weightage - 5

Input

```
3 3 0
1 2 3
```

Output

```
3 6 9
2 5 8
```

```
4 5 6
```

```
1 4 7
```

Weightage - 10

Input

```
5 5 1
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
```

Output

```
21 16 11 6 1
22 17 12 7 2
23 18 13 8 3
24 19 14 9 4
```

Weightage - 10

Input

```
5 5 0
21 16 11 6 1
22 17 12 7 2
23 18 13 8 3
```

Output

```
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
```

Weightage - 10

Input

```
3 3 1
3 6 9
2 5 8
1 4 7
```

Output

```
1 2 3
4 5 6
7 8 9
```

Weightage - 10

Input

```
4 4 1
4 8 12 16
3 7 11 15
2 6 10 14
```

Output

```
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Weightage - 10

Input

```
3 3 1
5 4 2
4 6 3
2 3 1
```

Output

```
2 4 5
3 6 4
1 3 2
```

Weightage - 10

Input

```
10 10 1
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
```

Output

```
41 31 21 11 1 41 31 21 11 1
42 32 22 12 2 42 32 22 12 2
43 33 23 13 3 43 33 23 13 3
44 34 24 14 4 44 34 24 14 4
```

Weightage - 10

Input

```
10 10 0
41 31 21 11 1 41 31 21 11 1
42 32 22 12 2 42 32 22 12 2
43 33 23 13 3 43 33 23 13 3
```

Output

```
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
```

Weightage - 10

Input

```
10  10  1
1  1  1  1  1  1  1  1  1  1
2  2  2  2  2  2  2  2  2  2
3  3  3  3  3  3  3  3  3  3
```

Output

```
5  4  3  2  1  5  4  3  2  1
5  4  3  2  1  5  4  3  2  1
5  4  3  2  1  5  4  3  2  1
5  4  3  2  1  5  4  3  2  1
```

Weightage - 10

Sample Input

```
3  3  1
2  3  1
4  6  3
5  4  2
```

Sample Output

```
5  4  2
4  6  3
2  3  1
```

Solution

Header

```
#include<stdio.h>
#include<malloc.h>



int ** rotatePixelImage( int **arr , int m , int n , int flag)
{
    int ctr,row,col,temp,start,end;

for( ctr = 0 ; ctr < m-1 ; ctr++)
{
 for(row = ctr+1 , col = ctr+1 ; row < m && col < m ; row++ , col++)
 {
    temp = arr[row][ctr];
    arr[row][ctr] = arr[ctr][col];
    arr[ctr][col] = temp;
 }
}
if( flag == 1)
{
for(col = 0 ; col < m ; col++)
{
    // colwise reverse
    for(start = 0 , end = m-1 ; start < end ;start++ , end--)
    {
        temp = arr[col][start];
        arr[col][start] = arr[col][end];
        arr[col][end] = temp;
    }
}

}
else if( flag == 0)
{
    // rowwise reverse
for(row = 0 ; row < m ; row++)
{
    for(start = 0 , end = m-1 ; start < end ; start++ , end--)
    {
        temp = arr[start][row];
        arr[start][row] = arr[end][row];
        arr[end][row] = temp;
    }
}
```

```
}


}
//printf("\n\n");


return arr;
}
```

Footer

```
int main()
{

int s , m , n , row , col , temp, ctr , flag , start , end;
int **arr;
scanf("%d%d%d" , &m , &n , &flag);
arr = ( int **)malloc( sizeof(int*) * m);
for(row = 0 ; row < m ; row++)
{
    arr[row]=(int*)malloc(n*sizeof(int));
    for(col = 0 ; col < n; col++)
         scanf("%d",&arr[row][col]);
}
arr = rotatePixelImage(arr,m,n,flag);

for(row = 0 ; row < m ; row++ , printf("\n"))
{
    for(col = 0; col < n ; col++)
        printf("%d " , arr[row][col]);
}
}



#include<stdio.h>
int main()
{

int s , m , n , row , col , temp, ctr , flag , start , end;
int arr[1000][1000];
scanf("%d%d%d" , &m , &n , &flag);
for(row = 0 ; row < m ; row++)
{
    for(col = 0 ; col < n; col++)
         scanf("%d",&arr[row][col]);
}
for( ctr = 0 ; ctr < m-1 ; ctr++)
{
 for(row = ctr+1 , col = ctr+1 ; row < m && col < m ; row++ , col++)
 {
    temp = arr[row][ctr];
    arr[row][ctr] = arr[ctr][col];
    arr[ctr][col] = temp;
 }
}
if( flag == 1)
{
for(col = 0 ; col < m ; col++)
{
    // colwise reverse
    for(start = 0 , end = m-1 ; start < end ;start++ , end--)
    {
        temp = arr[col][start];
```

```
            arr[col][start] = arr[col][end];
            arr[col][end] = temp;
        }
    }

    }
    else if( flag == 0)
    {
        // rowwise reverse
    for(row = 0 ; row < m ; row++)
    {
        for(start = 0 , end = m-1 ; start < end ; start++ , end--)
        {
            temp = arr[start][row];
            arr[start][row] = arr[end][row];
            arr[end][row] = temp;
        }
    }

    }
    //printf("\n\n");
    for(row = 0 ; row < m ; row++ , printf("\n"))
    {
        for(col = 0; col < n ; col++)
            printf("%d " , arr[row][col]);
    }

    return 0;
    }
```

Section 2 - Essay Writing

Q1

Sample Essay

No Essay

Keywords

technology, people, living, standard, government, investment,