1.

```
#include <stdio.h>
int main()
{
   int n;
   scanf("%d",&n);
   int a[n];
   for(int i=0;i<n;i++){
      scanf("%d",&a[i]);
   }
   for(int i=(n-1);i>=0;i--){
      printf("%d ",a[i]);
   }
}
```

2.

```
#include <stdio.h>
int main()
{
   int a[5] = {1,2,3,4,5};
   int s =0;
   for(int i = 0;i<5;i++){
      s= s+a[i];
   }
   printf("%d ",s);
}
```

3.

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

```
    int a[5]={1,2,3,4,5};
    int b[5];
    for(int i = 0;i<5;i++){
        b[i]=a[i];
    }
    for(int i=0;i<5;i++)
        printf("%d ",b[i]);
}
```

4.

```
#include <stdio.h>
int main()
{
    int a[8]={1,1,7,1,7,1,1,1};
    int c=0,b[8],k=0,l=0;
    for(int i=0;i<8;i++){

        for(int j = i+1;j<8;j++){
                if(a[i]==a[j])
                {
                    c=c+1;
                    break;
                }
        }
    }
        printf("%d ",c);
        return 0;
}
```

5.

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

```
    int a[8]={1,1,7,1,7,1,1,1},c;
    for(int i=0;i<8;i++){

        for(int j = i+1;j<8;j++){
                if(a[i]>a[j])
                {
                    c=a[i];
                    a[i]=a[j];
                    a[j]=c;
                }
        }
    }
        printf("%d %d",a[0],a[7]);
        return 0;
}
```

6.

```
#include <stdio.h>
int main()
{
    int a[8]={1,1,7,1,7,4,1,2},e[8],o[8],k,l;
    for(int i=0;i<8;i++){

        if(a[i]%2==0){
            e[k]=a[i];++k;
        }
        else{
        o[l]=a[i];++l;
        }
    }
        printf("odd integers ");
        for(int i =0;i<l;i++){
```

```
        printf("%d ",o[i]);


    }
    printf("\neven integers ");
    for(int i =0;i<k;i++){
        printf("%d ",e[i]);


    }
    return 0;
}
```

7.

```
#include <stdio.h>
int main()
{
   int a[8]={1,1,7,1,7,4,1,2},k;
   printf("old array ");
   for(int i = 0;i<8;i++)
   printf("%d ",a[i]);
   printf("\nenter the position and the new elemnt ");
   scanf("%d",&k);
   scanf("%d",&a[k]);
   printf("\nnew array with new element\n");
   for(int i = 0;i<8;i++)
   printf("%d ",a[i]);


    return 0;
}
```

8.

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

```
    int a[8]={1,1,7,1,7,4,1,2},k;
    printf("old array ");
    for(int i = 0;i<8;i++)
    printf("%d ",a[i]);
    printf("\nenter the position for deletion ");
    scanf("%d",&k);
    a[k]=0;
    printf("element deleted...");
    printf("\nnew array without element\n");
    for(int i = 0;i<8;i++)
    printf("%d ",a[i]);

        return 0;
}
```

9.

```
#include <stdio.h>
int main()
{
    int a[8]={1,1,7,1,7,4,1,2},c;
        for(int i=0;i<8;i++){

        for(int j = i+1;j<8;j++){
                if(a[i]>a[j])
                {
                    c=a[i];
                    a[i]=a[j];
                    a[j]=c;
                }
        }
    }
    printf("%d ",a[6]);
```

```
    return 0;
}
```

10.

```
#include <stdio.h>
int getMedian(int ar1[], int ar2[], int n)
{
   int i = 0;
   int j = 0;
   int count;
   int m1 = -1, m2 = -1;
   for (count = 0; count <= n; count++)
   {

      if (i == n)
      {
         m1 = m2;
         m2 = ar2[0];
         break;
      }
      else if (j == n)
      {
         m1 = m2;
         m2 = ar1[0];
         break;
      }

      if (ar1[i] <= ar2[j])
      {
         m1 = m2;
         m2 = ar1[i];
```

```
            i++;
        }
        else
        {
            m1 = m2;
            m2 = ar2[j];
            j++;
        }
    }

    return (m1 + m2)/2;
}



int main()
{
    int ar1[] = {1, 12, 15, 26, 38};
    int ar2[] = {2, 13, 17, 30, 45};

    int n1 = sizeof(ar1)/sizeof(ar1[0]);
    int n2 = sizeof(ar2)/sizeof(ar2[0]);
    if (n1 == n2)
        printf("Median is %d", getMedian(ar1, ar2, n1));
    else
        printf("Doesn't work for arrays of unequal size");
    getchar();
    return 0;
}
```

11.

```
#include <stdio.h>
#define N 4
```

```
void multiply(int mat1[][N], int mat2[][N], int res[][N])
{
    int i, j, k;
    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            res[i][j] = 0;
            for (k = 0; k < N; k++)
                res[i][j] += mat1[i][k] * mat2[k][j];
        }
    }
}

int main()
{
    int mat1[N][N] = { { 1, 1, 1, 1 },
                       { 2, 2, 2, 2 },
                       { 3, 3, 3, 3 },
                       { 4, 4, 4, 4 } };

    int mat2[N][N] = { { 1, 1, 1, 1 },
                       { 2, 2, 2, 2 },
                       { 3, 3, 3, 3 },
                       { 4, 4, 4, 4 } };

    int res[N][N];
    int i, j;
    multiply(mat1, mat2, res);

    printf("Result matrix is \n");
```

```
    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++)
            printf("%d ", res[i][j]);
        printf("\n");
    }

    return 0;
}
```

12.

```
#include <stdio.h>
int main() {
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns: ");
    scanf("%d %d", &r, &c);


    printf("\nEnter matrix elements:\n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("Enter element a%d%d: ", i + 1, j + 1);
            scanf("%d", &a[i][j]);
        }


    printf("\nEntered matrix: \n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("%d  ", a[i][j]);
            if (j == c - 1)
                printf("\n");
```

```
    }


  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      transpose[j][i] = a[i][j];
    }


  printf("\nTranspose of the matrix:\n");
  for (i = 0; i < c; ++i)
    for (j = 0; j < r; ++j) {
      printf("%d  ", transpose[i][j]);
      if (j == r - 1)
        printf("\n");
    }
  return 0;
}
```

13.

```
#include <stdio.h>

void main()

  {
   int i,j,arr1[50][50],sum=0,n,m=0;

     printf("\n\nFind sum of left diagonals of a matrix :\n");


          printf("Input the size of the square matrix : ");
   scanf("%d", &n);
```

```
 m=n;
      printf("Input elements in the first matrix :\n");
for(i=0;i<n;i++)
{
   for(j=0;j<n;j++)
   {
            printf("element - [%d],[%d] : ",i,j);
            scanf("%d",&arr1[i][j]);
   }
}
      printf("The matrix is :\n");
      for(i=0;i<n;i++)
      {
        for(j=0;j<n ;j++)
          printf("% 4d",arr1[i][j]);
         printf("\n");
      }

      for(i=0;i<n;i++)
      {
  m=m-1;
       for(j=0;j<n ;j++)
   {
     if (j==m)
      {
        sum= sum+arr1[i][j];
      }

   }
      }
printf("Addition of the  left Diagonal elements is :%d\n",sum);
```

```
}
```

14.

```
#include <stdio.h>

int main (void)
{
	int a[10][10];
	int i = 0, j = 0, row = 0, col = 0;

	printf ("Enter the order of the matrix (mxn):\n");
	printf ("where m = number of rows; and\n");
	printf ("      n = number of columns\n");
	scanf ("%d %d", &row, &col);

	int flag = 0;

	printf ("Enter the elements of the matrix\n");
	for (i = 0; i < row; i++)
	{
		for (j = 0; j < col; j++)
		{
			scanf ("%d", &a[i][j]);
		}
	}

	for (i = 0; i < row; i++)
	{
		for (j = 0; j < col; j++)
		{
			if (i == j && a[i][j] != 1)
			{
```

```
                    flag = -1;
                    break;
                }
                else if (i != j && a[i][j] != 0)
                {
                    flag = -1;
                    break;
                }
            }
        }

        if (flag == 0)
        {
            printf ("It is a IDENTITY MATRIX\n");
        }
        else
        {
            printf ("It is NOT an identity matrix\n");
        }

        return 0;
}
```

15.

```
#include <stdio.h>



int search(int mat[4][4], int n, int x)
{
        if (n == 0)
            return -1;
        int smallest = mat[0][0], largest = mat[n - 1][n - 1];
```

```
        if (x < smallest || x > largest)
                return -1;



        int i = 0, j = n - 1;
        while (i < n && j >= 0)
        {
                if (mat[i][j] == x)
                {
                        printf("\n Found at %d, %d", i, j);
                        return 1;
                }
                if (mat[i][j] > x)
                        j--;
                else
                        i++;
        }

        printf("n Element not found");
        return 0; // if ( i==n || j== -1 )
}



int main()
{
        int mat[4][4] = {
                { 10, 20, 30, 40 },
                { 15, 25, 35, 45 },
                { 27, 29, 37, 48 },
                { 32, 33, 39, 50 },
        };
```

```
        search(mat, 4, 29);
        return 0;
}
```