1)

```
#include <iostream>
using namespace std;
int factorial(int n)
{
  if(n==0)
  {
     return 1;
  }
  else if(n==1)
  {
     return 1;
  }
  else
  {
     return (n*factorial(n-1));
  }
}
int main()
{
    int n,i,j;
    cout<<"Enter Number of lines : ";
    cin>>n;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-i;j++)
        {
            cout<<" ";
        }
        for(j=0;j<i+1;j++)
        {
            cout<<factorial(i)/(factorial(j)*factorial(i-j))<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

2)

```
#include <iostream>

#include <algorithm>

using namespace std;

int main()

{

    int arr[10], n,i;

    cin>>n;

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

    {

        cin>>arr[i];

    }

    cout<<"Entered list:";

    cout<<"[";

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

          {
            if(i<n-1)
            {
                  cout << arr[i]<<",";
            }
            else
            {
            cout << arr[i];
            }
        }
        cout<<"]"<<endl;


  next_permutation(arr,arr+n);
  cout<<"Output : ";
  cout<<"[";
        for (int i = 0; i < n; i++)
        {
            if(i<n-1)
            {
                  cout << arr[i]<<",";
            }
            else
            {
            cout << arr[i];
            }
        }
        cout<<"]";

        return 0;
}
```

3)

```
#include <iostream>
#include <regex>

using namespace std;
int main()
{
    string s,p;
    cout<<"Enter string: ";
    cin>>s;
    cout<<"Enter pattern : ";
    cin>>p;
    regex  b( p );

    if ( regex_match(s,b) )
    {
        cout << "TRUE";
    }
    else
    {
        cout<<"FALSE";
    }
    return 0;
}
```

4)

```
#include <iostream>
using namespace std;

bool isSubsetSum(int set[], int n, int sum)
{

        if (sum == 0)
                return true;
        if (n == 0)
                return false;


        if (set[n - 1] > sum)
                return isSubsetSum(set, n - 1, sum);


        return isSubsetSum(set, n - 1, sum)|| isSubsetSum(set, n - 1, sum - set[n -1]);
}


int main()
{
        int set[20];
        int i,n,sum;
        cout<<"Enter the size of array :";
        cin>>n;
        cout<<"Enter array elements :";
        for(i=0;i<n;i++)
        {
           cin>>set[i];
        }
        cout<<"Enter a number as sum :";
        cin>>sum;

        if (isSubsetSum(set, n, sum) == true)

                cout <<"True";

        else

                cout <<"False";

        return 0;

}
```

5)

```
#include <bits/stdc++.h>
#include <iostream>
#include <math.h>
using namespace std;
 int max(int a, int b)
{ return (a > b) ? a : b;
}

int cutRod(int price[], int index, int n)
{

    if (index == 0)
   {
        return n * price[0];
    }

    int notCut = cutRod(price,index - 1,n);
    int cut = INT_MIN;
    int rod_length = index + 1;
    if (rod_length <= n)
        cut = price[index]
               + cutRod(price,index,n - rod_length);

    return max(notCut, cut);
}
int main()
{
    int arr[] = { 1, 5, 8, 9, 10, 17, 17, 20 };
    int size = sizeof(arr) / sizeof(arr[0]);
    cout << "Maximum Obtainable Value is "<< cutRod(arr, size - 1, size);
    getchar();
    return 0;
}
```

6)

```
#include <iostream>
#include <vector>
#include <climits>
using namespace std;

void findMinAndMax(vector<int> const &nums, int low, int high, int &min, int
&max)
{
  if (low == high)
  {
        if (max < nums[low])
        {
            max = nums[low];
        }
  if (min > nums[high])
        {
            min = nums[high];
        }
return;
    }
  if (high - low == 1)
    {
        if (nums[low] < nums[high])
          {
            if (min > nums[low])
           {
                min = nums[low];
            }

            if (max < nums[high])
            {
                max = nums[high];
            }
}
        else {
            if (min > nums[high])
          {
                min = nums[high];
            }

            if (max < nums[low])
            {
                max = nums[low];
            }
        }
        return;
    }
  int mid = (low + high) / 2;
   findMinAndMax(nums, low, mid, min, max);
    findMinAndMax(nums, mid + 1, high, min, max);
}

int main()
{
    vector<int> nums = { 7, 2, 9, 3, 1, 6, 7, 8, 4 };
     int max = INT_MIN, min = INT_MAX;
    int n = nums.size();
    findMinAndMax(nums, 0, n - 1, min, max);

    cout << "The minimum array element is " << min << endl;
    cout << "The maximum array element is " << max << endl;
    return 0;
}
```

7)

```
#include <bits/stdc++.h>
using namespace std;
void printAnagrams(string arr[], int size)
{
        unordered_map<string, vector<string> > map;
        for (int i = 0; i < size; i++)
              {
```

```
        string word = arr[i];
        char letters[word.size() + 1];
        strcpy(letters, word.c_str());
        sort(letters, letters + word.size() + 1);
        string newWord = "";
        for (int i = 0; i < word.size() + 1; i++)
        {
                newWord += letters[i];
        }
        if (map.find(newWord) != map.end())
        {
                map[newWord].push_back(word);
        }
        else {
                vector<string> words;
                words.push_back(word);
                map[newWord] = words;
        }
}
unordered_map<string, vector<string> >::iterator it;
for (it = map.begin(); it != map.end(); it++) {
        vector<string> values = map[it->first];
        if (values.size() > 1) {
                cout << "[";
                for (int i = 0; i < values.size() - 1; i++) {
                        cout << values[i] << ", ";
                }
                cout << values[values.size() - 1];
                cout << "]";
        }
    }
}


int main()
{
    string arr[] = { "cat", "dog", "tac", "god", "act" };
    int size = sizeof(arr) / sizeof(arr[0]);


    printAnagrams(arr, size);


    return 0;
}
```

8)

```
#include <bits/stdc++.h>
using namespace std;
#define R 4
#define C 4

void spiralPrint(int m, int n, int a[R][C])
{
    int i, k = 0, l = 0;

    while (k < m && l < n) {
        for (i = l; i < n; ++i) {
            cout << a[k][i] << " ";
        }
        k++;
```

```
                for (i = k; i < m; ++i) {
                        cout << a[i][n - 1] << " ";
                }
                n--;

                if (k < m) {
                        for (i = n - 1; i >= l; --i) {
                                cout << a[m - 1][i] << " ";
                        }
                        m--;
                }

                if (l < n) {
                        for (i = m - 1; i >= k; --i) {
                                cout << a[i][l] << " ";
                        }
                        l++;
                }
        }
}

int main()
{
        int a[R][C] = {{1, 2, 3, 4},
                                {5, 6, 7, 8},
                                {9, 10, 11, 12},
                                {13, 14, 15, 16}};

        spiralPrint(R, C, a);
        return 0;
}
```

9)

```
#include <iostream>
using namespace std;
```

```
int main()
{       int arr[10],i,n,k;
        cout<<"Enter the array size :";
        cin>>n;
        cout<<"Enter array elements"<<endl;
        for(i=0;i<n;i++)
        {
           cin>>arr[i];
        }
        cout<<"enter target value:"<<endl;
        cin>>k;
        for (int i = 0; i < n; i++)
        {       if (arr[i] == k)
                {
                        cout<<"The position is :"<<i<<endl;
                        break;
                }
                else if (arr[i] > k)
                {
                    cout<<"The position is :"<<i<<endl;
                    break;
                }
                else if( k>arr[n-1])
                {
                    cout<<"The position is :"<<n<<endl;
                    break;
                }
         }
                return 0;
}
```

10)

```
string=input().split()
l=[]
lc=[]
for i in string:
   l.append(i)
for i in l:
   y=l.count(i)
   lc.append(y)
#print(lc,l)
ls=[]
for i in range(0,len(lc)):
   if(lc[i]>1):
      ls.append(l[i])
s=set(ls)
ss=list(s)
print(ss)
```

11)

```
#include <bits/stdc++.h>

using namespace std;


int countEqual(int A[], int B[], int N)

{



        int first = 0;

        int second = N - 1;

        int count = 0;



        while (first < N && second >= 0) {
```

```
        if (A[first] < B[second]) {
        first++;
        }

        else if (B[second] < A[first]) {

        second--;
        }

        else {

        count++;

        first++;

        second--;
        }
    }

    return count;
}

int main()
```

```
{
        int A[] = { 2, 4, 5, 8, 12, 13, 17,
                        18, 20, 22, 309, 999 };
        int B[] = { 109, 99, 68, 54, 22, 19,
                        17, 13, 11, 5, 3, 1 };
        int N = sizeof(A) / sizeof(int);
        cout << countEqual(A, B, N);

        return 0;
}
```

12)

```
#include <bits/stdc++.h>
using namespace std;

#define R 4
#define C 4
void counterClockspiralPrint(int Matrix[R][C])
{
        int size = R;
        int i = size, k = 0, flag = 0, j = 0;
        while (i > 0) {
                for (j = flag; j < i; j++) {
                cout << Matrix[j][k] << " ";
                }

                i = i - 1;

                j = j - 1;

                k = j;

                if (i > 0) {

                for (j = size - i; j < i + 1; j++)

                        cout << Matrix[k][j] << " ";

                for (j = k - 1; j > size - i - 2; j--)

                        cout << Matrix[j][k] << " ";

                }

                else

                break;

                j = j + 1;

                k = j;

                i = i - 1;

                if (i > 0) {

                for (j = i; j > size - i - 2; j--)

                        cout << Matrix[k][j] << " ";

                k = k + 1;

                i = i + 1;

                flag = flag + 1;

                }

                else

                break;
```

```
        }
}

int main()
{
        int Matrix[R][C] = { { 1, 2, 3, 4 },
                                        { 5, 6, 7, 8 },
                                        { 9, 10, 11, 12 },
                                        { 13, 14, 15, 16 } };
        counterClockspiralPrint(Matrix);
        return 0;
}
```

13)

```
#include <bits/stdc++.h>
using namespace std;

#define R 3
#define C 6
void formSpiralMatrix(int arr[], int mat[R][C])
{
        int top = 0,
                bottom = R - 1,
        left = 0,
        right = C - 1;

int index = 0;

while (1) {

        if (left > right)
        break;
        for (int i = left; i <= right; i++)
        mat[top][i] = arr[index++];
        top++;

        if (top > bottom)
        break;
        for (int i = top; i <= bottom; i++)
        mat[i][right] = arr[index++];
        right--;

        if (left > right)
        break;
        for (int i = right; i >= left; i--)
        mat[bottom][i] = arr[index++];
        bottom--;
```

```
                if (top > bottom)

                break;

                for (int i = bottom; i >= top; i--)

                mat[i][left] = arr[index++];

                left++;

        }

}

void printSpiralMatrix(int mat[R][C])

{


        for (int i = 0; i < R; i++) {

                for (int j = 0; j < C; j++)

                cout << mat[i][j] << " ";

                cout << '\n';

        }

}


int main()

{

        int arr[]

                = { 1, 2, 3, 4, 5, 6,

                7, 8, 9, 10, 11, 12,

                13, 14, 15, 16, 17, 18 };
```

```
	int mat[R][C];


	formSpiralMatrix(arr, mat);

	printSpiralMatrix(mat);


	return 0;

}
```

14)

```
#include <bits/stdc++.h>


using namespace std;

int maxSum(int p0, int p1, int a[], int pos, int n)

{

	if (pos == n) {

		if (p0 == p1)

		return p0;

		else

		return 0;

	}

	int ans = maxSum(p0, p1, a, pos + 1, n);

	ans = max(ans, maxSum(p0 + a[pos], p1, a, pos + 1, n));

	ans = max(ans, maxSum(p0, p1 + a[pos], a, pos + 1, n));
```

```
        return ans;
}
int main()
{
        int n = 4;
        int a[n] = { 1, 2, 3, 6 };
        cout << maxSum(0, 0, a, 0, n);
        return 0;
}
```

15)

```
#include "bits/stdc++.h"
using namespace std;
void CamelCase(vector<string>& words,
                string pattern)
{
        map<string, vector<string> > map;
        for (int i = 0; i < words.size(); i++) {
                string str = "";
                int l = words[i].length();
                for (int j = 0; j < l; j++) {

                if (words[i][j] >= 'A'
                    && words[i][j] <= 'Z') {
                    str += words[i][j];
                    map[str].push_back(words[i]);
                }
                }
        }

        bool wordFound = false;
        for (auto& it : map) {
                if (it.first == pattern) {
                wordFound = true;
                for (auto& itt : it.second) {
                        cout << itt << endl;
                }
                }
        }
        if (!wordFound) {
                cout << "No match found";
        }
}
int main()
{
        vector<string> words = {
                "Hi", "Hello", "HelloWorld",
                "HiTech", "HiGeek", "HiTechWorld",

                "HiTechCity", "HiTechLab"

        };

        string pattern = "HT";


        CamelCase(words, pattern);


        return 0;

}
```

16)

```
#include<iostream>

#include<algorithm>


using namespace std;

string longestCommonPrefix(string ar[], int n)

{

        if (n == 0)

                return "";


        if (n == 1)

                return ar[0];
```

```
    sort(ar, ar + n);
    int en = min(ar[0].size(),
                 ar[n - 1].size());
    string first = ar[0], last = ar[n - 1];
    int i = 0;
    while (i < en && first[i] == last[i])
        i++;

    string pre = first.substr(0, i);
    return pre;
}
int main()
{
    string ar[] = {"geeksforgeeks", "geeks",
                                "geek", "geezer"};
    int n = sizeof(ar) / sizeof(ar[0]);
    cout << "The longest common prefix is: "
        << longestCommonPrefix(ar, n);
    return 0;
}
```

17)

```
#include <bits/stdc++.h>
using namespace std;
```

```
void printPattern(int n)
{

        int i, j;
        for (i = 1; i <= n; i++) {
                for (j = 1; j < 2 * n; j++) {
                if (j == (n - i + 1)
                        || j == (n + i - 1)) {
                        cout << "* ";
                }
                else if ((i >= 4 && i <= n - 4)
                                && (j == n - i + 4
                                        || j == n + i - 4)) {

                        cout << "* ";
                }
                else if (i == n
                                || (i == n - 4
                                        && j >= n - (n - 2 * 4)
                                        && j <= n + n - 2 * 4)) {

                        cout << "* ";
                }
                else {
                        cout << " "
                                << " ";
                }
                }
                cout << "\n";
        }
}

int main()
{
        int N = 9;

        printPattern(N);
}
```

18)

```
#include <bits/stdc++.h>
using namespace std;
void fib(int f[], int N)
{
        f[1] = 1;
        f[2] = 1;

        for (int i = 3; i <= N; i++)
```

```
                f[i] = f[i - 1] + f[i - 2];
}

void fiboTriangle(int n)
{
        int N = n * (n + 1) / 2;
        int f[N + 1];
        fib(f, N);
        int fiboNum = 1;
        for (int i = 1; i <= n; i++) {
                for (int j = 1; j <= i; j++)
                cout << f[fiboNum++] << " ";

                cout << endl;
        }
}

int main()
{
        int n = 5;
        fiboTriangle(n);
        return 0;
}
```

19)

```
#include<bits/stdc++.h>

using namespace std;

void minMax(vector<int>&arr){

	int min_value = 0;

	int max_value = 0;

	int n = arr.size();

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

	int j = n - 1;

	for(int i = 0; i < n - 1; i++)

	{

		min_value += arr[i];

		max_value += arr[j];

		j -= 1;

	}

	cout<<min_value<<" "<<max_value<<endl;

}

int main(){


	vector<int>arr = {10, 9, 8, 7, 6, 5};

	vector<int>arr1 = {100, 200, 300, 400, 500};


	minMax(arr);
```

```
        minMax(arr1);

}
```

20) //python code

```
n=int(input("n value:"))
arr=list(map(int,input().split()))
x=int(input())
for i in range(n):
        if(arr[i]==x):
        print(i)
        continue
```

21)

```
#include <bits/stdc++.h>
using namespace std;
void AwesomeSort(vector<int> m, int n)
{
        vector<int> v1, v2, v3;

        int i;
        for (i = 0; i < n; i++) {
                if (m[i] % 10 == 0)
                v1.push_back(m[i]);

                else if (m[i] % 2 == 0)

                v2.push_back(m[i]);

                else

                v3.push_back(m[i]);

        }

        sort(v1.begin(), v1.end(), greater<int>());


        for (int i = 0; i < v1.size(); i++) {

                cout << v1[i] << " ";

        }

        for (int i = v2.size()-1; i >= 0; i--) {

                cout << v2[i] << " ";

        }

        for (int i = 0; i < v3.size(); i++) {

                cout << v3[i] << " ";

        }

}

int main()

{

        vector<int> arr{ 5, 10, 30, 7 };

        int N = arr.size();

        AwesomeSort(arr, N);
        return 0;
}
```

22)

```
#include <bits/stdc++.h>
using namespace std;
int minCollectingSpeed(vector<int>& piles,
                        int H)
{
        int ans = -1;

        int low = 1, high;
        high = *max_element(piles.begin(),
                                piles.end());
        while (low <= high)

        {
                int K = low + (high - low) / 2;

                int time = 0;
                for (int ai : piles) {

                time += (ai + K - 1) / K;
                }
                if (time <= H) {
                ans = K;
                high = K - 1;
                }
                else {
                low = K + 1;
                }
        }
        cout << ans;
 }
int main()
{
        vector<int> arr = { 3, 6, 7, 11 };
        int H = 8;
        minCollectingSpeed(arr, H);

        return 0;
}
```

23)

```
#include<bits/stdc++.h>
using namespace std;
```

```
int minDiff(int arr[], int n, int k)

{

        int result = INT_MAX;

        sort(arr, arr + n);

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

                result = min(result, arr[i+k-1] - arr[i]);


        return result;

}

int main()

{

        int arr[] = {10, 100, 300, 200, 1000, 20, 30};

        int n = sizeof(arr)/sizeof(arr[0]);

        int k = 3;


        cout << minDiff(arr, n, k) << endl;

        return 0;

}
```

24)

```
#include <bits/stdc++.h>

using namespace std;

int maxUniqueElements(int A[], int N)

{
        unordered_map<int, int> mp;
        for (int i = 0; i < N; i++) {
                mp[A[i]]++;
        }
        int cnt = 0;

        for (auto x : mp) {
                if (x.second % 2 == 0) {
                cnt++;
                }
        }
        int ans = mp.size();
        if (cnt % 2 == 1) {
                ans--;
        }
        return ans;
}
int main()
{
        int N = 5;
        int A[] = { 1, 2, 1, 3, 7 };
        cout << maxUniqueElements(A, N);
}
```

25)

```
#include<iostream>

using namespace std;

int min(int x, int y) { return (x < y)? x : y; }

int max(int x, int y) { return (x > y)? x : y; }

int findLength(int arr[], int n)

{

        int max_len = 1;

        for (int i=0; i<n-1; i++)

        {

                int mn = arr[i], mx = arr[i];

                for (int j=i+1; j<n; j++)

                {

                mn = min(mn, arr[j]);

                mx = max(mx, arr[j]);

                if ((mx - mn) == j-i)

                                max_len = max(max_len, mx-mn+1);

                }

        }

        return max_len; // Return result

}

int main()

{

        int arr[] = {1, 56, 58, 57, 90, 92, 94, 93, 91, 45};
```

```
    int n = sizeof(arr)/sizeof(arr[0]);

    cout << "Length of the longest contiguous subarray is "

         << findLength(arr, n);

    return 0;

}
```