```
/*

input = 10

0 0 0 0 0 0 0 0 0 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 X X X X X X X X 0
0 0 0 0 0 0 0 0 0 0

*/
#include <iostream>
using namespace std;

int main() {
int i, k, j, n;
   cin>>n;

for(i = 1; i<=n; i++){
  if(i == 1||i==n){
    for(j=1; j<=n; j++){
       cout<<"0"<<" ";
    }
    cout<<endl;
  }

  else{
     cout<<"0"<<" ";
     for(k=1; k<=n-2; k++){
        cout<<"X"<<" ";
     }
     cout<<"0"<<" "<<endl;
  }
}
return 0;
}
```

```
/*
i/p:
5 12 24 36 48 60

o/p:
12
12 24
12 24 36
12 24 36 48
12 24 36 48 60
24
24 36
24 36 48
24 36 48 60
36
36 48
36 48 60
48
48 60
60
*/

#include <iostream>
using namespace std;

int main() {
      int n, i, a[100], j, k;
      cin>>n;

      for(i = 0; i<n; i++){
          cin>>a[i];
      }

      for(i = 0; i<n; i++){
          for(j = i; j<n; j++){
              for(k = i; k<=j; k++){
                  cout<<a[k]<<" ";
              }
              cout<<endl;
          }
      }
      return 0;
}
```

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

// Function to check if the given array is bitonic
int checkBitonic(int arr[], int n)
{
    int i, j;

    // Check for increasing sequence
    for (i = 1; i < n; i++) {
        if (arr[i] > arr[i - 1])
            continue;

        if (arr[i] <= arr[i - 1])
            break;
    }

    if (i == n)
        return 1;

    // Check for decreasing sequence
    for (j = i + 1; j < n; j++) {
        if (arr[j] < arr[j - 1])
            continue;

        if (arr[j] >= arr[j - 1])
            break;
    }

    i = j;

    if (i != n)
        return 0;

    return 1;
}


int main()
{
    int arr[] = {12,13,14,20,19,5,1};

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

    (checkBitonic(arr, n) == 1) ? cout << "YES"
                                : cout << "NO";

    return 0;
}
```

```
/*
Luke Skywalker gave Chewbacca an integer number x.
Chewbacca isn't good at numbers but he loves inverting digits in them.
Inverting digit t means replacing it with digit 9-t.

Help Chewbacca to transform the initial number x to the minimum possible
positive number by inverting some (possibly, zero) digits.
The decimal representation of the final number shouldn't start with a
zero.
*/
#include <iostream>
using namespace std;

int main() {
    char a[50];
    cin>>a;

    int i=0;
    if(a[i] == '9'){
        i++;
    }

    for( ; a[i]!='\0'; i++){
        int digit = a[i] - '0';

        if(digit >= 5){
            digit = 9 - digit;
            a[i] = digit + '0';
        }
    }

    cout<<a<<endl;
      return 0;
}
```

```
/*
8
10 -3 -4 7 6 5 -4 -1

The maximum non-circular sub array sum is: 21
The maximum circular sub array sum is: 23
*/
#include <iostream>
using namespace std;

int kedane(int a[], int n);
int circularMax(int a[], int n);

int main() {
      int n;
      cin>>n;

      int i, arr[100];
    for(i = 0; i<n; i++){
        cin>>arr[i];
    }

    cout<<"The maximum non-circular sub array sum is: "<<kedane(arr,
n)<<endl;
    cout<<"The maximum circular sub array sum is: "<<circularMax(arr, n);
      return 0;
}

int kedane(int a[], int n){
    int i, tempMax = 0;

    int maxSum = a[0], localMax = a[0];
    for(i = 1; i<n; i++){
        localMax = a[i];
        maxSum = localMax + maxSum;
        maxSum = max(localMax, maxSum);
        tempMax = max(tempMax, maxSum);
    }

    return tempMax;
}

int circularMax(int a[], int n){
    int cand1 = kedane(a,n), i, cumSum = 0;

    for(i = 0; i<n; i++){
        cumSum += a[i];
        a[i]  = -a[i];
    }

    int cand2 = cumSum + kedane(a,n);

    int circMax = 0;
    circMax = max(cand1, cand2);
```

```
    return(circMax);
}
```

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

int main() {
      /*
          ABCDEEDCBA
          ABCDDCBA
          ABCCBA
          ABBA
          AA
      */

      //take the number of rows
      int n;
      cin>>n;

      //row number
      int i = 0;
      while(i<n){
          char ch = 'A';
       //print for ch++
          for(int j = 0; j<n-i; j++){
              cout<<ch;
              ch++;
          }
          ch--;
       //print for ch--
          for(int j = 0; j<n-i; j++){
              cout<<ch;
              ch--;
          }
       //endl
          cout<<endl;
       //update row number
          i++;
      }
      return 0;
}
```

```
//binary to decimal (naive)

#include <iostream>
using namespace std;

int main() {
      int num;
      cin>>num;

      int temp = num;
      int last = 0, dec = 0, base = 1;

      while(num){
          last = num % 10;
          dec += last  * base;
          num = num/10;
          base = base*2;
      }
      cout<<dec;
      return 0;
}
```

```
//bitonic subarrays

#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main(){
    int times;
    cin>>times;

    while(times--){
        int n;
        cin>>n;

        vector<int> arr(n);
        vector<int> inc(n);
        vector<int> dec(n);


        for(int i = 0; i<n; i++){
            cin>>arr[i];
        }

        /*
          create a inc[] array from left to right that
          has the length of the increasing subarray,
          in arr[i], uptil that index.

          arr[] = [12 4 78 90 45 23]
          inc[] = [1 1 2 3 1]


          Similarly create another array dec[] from
          right to left, that has the length of the decreasing
          subarrays, by calculating the length of increasing subarrays
          from the last index.

          arr[] = {12, 4, 78, 90, 45, 23}
          dec[] is {2, 1, 1, 3, 2, 1}
        */

        inc[0] = 1;
        dec[n-1] = 1;

        for(int i = 1; i<n; i++){
            if(arr[i]>arr[i-1]){
                inc[i] = inc[i-1] + 1;
            }
            else{
                inc[i] = 1;
            }
        }
```

```
        for(int i = n-2; i>=0; i--){
            if(arr[i]>=arr[i+1]){
                dec[i] = dec[i+1] + 1;
            }
            else{
                dec[i] = 1;
            }
        }

        //max value of inc[i] + dec[i] - 1  is the length of the bitonic
subarray

        int max = inc[0] + dec[0] - 1;
        for(int i = 1; i<n; i++){
            if(inc[i] + dec[i] - 1 > max){
                max = inc[i] + dec[i] - 1;
            }
        }

        cout<<max<<endl;
    }
}
```

```
//count the set bits

#include <iostream>
using namespace std;

int main() {
	//get the number
	int num;
	cin>>num;

	int counter = 0;
	//make a loop that ands the number with 1, and it is updated with
every iteration such that
	//the number is right shifted.

	/*  000001010
	   &000000001....will give back 0, and this is added to a counter.

	    then right shifted

	    000000101
	   &000000001.....will give back 1, which will be added to the
counter, thus counter = 1.

	   similarly, counter = 2, at then end.

	   counter = counter + (num & 1)
	*/

	while(num>0){
	    //update counter
	    counter = counter + (num&1);

	    //update iteration
	    num = num>>1;
	}
	cout<<counter;
	return 0;
}
```

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

void decToOct(int num)
{     int arr[50], i = 0;

      while (num != 0) {
         arr[i] = num % 8;
         num /= 8;
         i++;
    }

    for (int j = i - 1; j >= 0; j--)
        cout << arr[j];
}

int main()
{
    int n;
      cin >>n;

    decToOct(n);

    return 0;
}
```

```
/*
Due to growing Traffic Problem Kejriwal wants to devise a new scheme.
The scheme is as follows, each car will be allowed to run on Sunday,
if the sum of digits which are even is divisible by 4 or sum of digits
which are odd in that number is divisible by 3.

However to check every car for the above criteria can't be done by the
Delhi Police.
You need to help Delhi Police by finding out if a car numbered N will be
allowed to run on Sunday?
*/

#include<iostream>
using namespace std;

int main() {
      int freq;
      cin>>freq;
      int last = 0, sumEven = 0, sumOdd = 0;

      for(int i = 0; i < freq; i++){
            int numPlate;
            cin>>numPlate;

        while(numPlate){
            last = numPlate % 10;
            if(last % 2 == 0){
                sumEven += last;
            }else{
                sumOdd += last;
            }
            numPlate = numPlate / 10;
        }
        //  cout<<sumOdd<<endl<<sumEven<<endl;
        if((sumOdd % 3 == 0)||(sumEven % 4 == 0)){
            cout<<"Yes"<<endl;
        }else{
            cout<<"No"<<endl;
        }
        sumEven = 0;
        sumOdd = 0;
      }
      return 0;
}
```

```
//print the edge elements of a 2-D Array
/*

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

1 2 3 4 8 12 16 15 14 13 9 5

*/
#include <iostream>
using namespace std;

void spiralPrint(int a[100][100], int R, int C){
    int i, j = 0;

    while(j<C){
        i = 0;
        cout<<a[i][j]<<" ";
        j++;
    }
    j--;
    i++;

    while(i<R){
        cout<<a[i][j]<<" ";
        i++;
    }
    i--;
    j--;

    while(j>=0){
        cout<<a[i][j]<<" ";
        j--;
    }
    j++;
    i--;

    // cout<<endl<<i<<endl<<j<<endl;

    while(i>0){
        cout<<a[i][j]<<" ";
        i--;
    }
}

int main() {
     int arr[100][100], r, c;

     cin>>r;
     cin>>c;
```

```
    for(int i = 0; i<r; i++){
        for(int j = 0; j<c; j++){
           cin>>arr[i][j];
        }
    }

    spiralPrint(arr, r, c);

    return 0;
}
```

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

int main() {
      int minF, maxF, skip;
      cin>>minF>>maxF>>skip;
      int i = minF;

      //C = (5/9)(F - 32)

      while(i <= maxF){
          int celsius = (5*(i - 32))/9;
          cout<<i<<"  "<<celsius<<endl;
          i = i + skip;
      }
      return 0;
}
```

```
//Fibonnaci
#include <iostream>
using namespace std;
int main() {
    int a=0, b=1, c = a+b, n;

    cout<<"Enter the number of terms up to which you want to compute the
series: ";
    cin>>n;

    cout<<"0 1 ";

    int i = 2;
    while(i<n){
        cout<<c<<" ";

        a = b;
        b = c;
        c = a+b;

        i++;
    }
}
```

```
/*1
1 2
3 5 8
13 21 34 55
89 144 233 377 610 */


#include <iostream>
using namespace std;

int main() {
      int n, i, sum = 0;
      cin>>n;
      int cuml = n;

      //getting cumulative sum
      while(cuml>0){
          sum += cuml;
          cuml--;
      }

      //fibonaaci array generation
      int arr[200];
      arr[0] = 0;
      arr[1] = 1;
      for(i = 2; i <= sum; i++){
          arr[i] = arr[i-1] + arr[i-2];
      }

      int num = 1;
    for (int i = 1; i <= n; i++)
    {
        for (int j = 1;j <= i;j++)
            cout << arr[num++] << " ";

        cout << endl;
    }

//    for(i = 0; i<sum; i++){
//        cout<<arr[i]<<" ";
//    }
      return 0;
}
```

```
/*
we compare two numbers XY (Y appended at the end of X) and YX (X appended
at the end of Y).
If XY is larger, then X should come before Y in output, else Y should
come before.
For example, let X and Y be 542 and 60. To compare X and Y, we compare
54260 and 60542.
Since 60542 is greater than 54260, we put Y first.
*/

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

bool mycmp(string a,string b){
    string ab=a.append(b);
    string ba=b.append(a);
    return ab.compare(ba)>0 ? 1 : 0;
}

int main() {
    int times;
    cin>>times;

    while(times--){
        int n;
        cin>>n;

        string str[n];
        for(int i=0;i<n;i++){
            cin>>str[i];
        }

        sort(str,str+n,mycmp);

        for(int i=0;i<n;i++){
            cout<<str[i];
        }

        cout<<endl;
    }
      return 0;
}
```

```
//frequency counter

#include <iostream>
using namespace std;

int main() {
      long long int num;
      cin>>num;

      int check;
      cin>>check;

      long long int temp = num;
       int last = 0,
          count = 0;

      while(temp){
          last = temp % 10;
          if(last == check){
              count++;
          }
          temp = temp/10;
      }
      cout<<count;
      return 0;
}
```

```
/*

*          ***********
*          *
*          *
*          *
*          *
*          *
*          *
*          *
*          *
*          *
**********************
           *          *
           *          *
           *          *
           *          *
           *          *
           *          *
           *          *
           *          *
           *          *
***********           *

*/

#include <iostream>
using namespace std;

int main() {
      int n,i = 1,j;
      cin>>n;

      //for first row
         cout<<"*";

         for(j = 2; j<=n/2; j++){
          cout<<" ";
       }

         for(j = n/2 + 1; j<=n; j++){
             cout<<"*";
         }

         cout<<endl;

      //middle - 1
      for(i = 2; i <= n/2; i++){
         cout<<"*";

         for(j = 2; j<=n/2; j++){
          cout<<" ";
       }
```

```
        cout<<"*";

        for(j = n/2 + 2; j<=n; j++){
            cout<<" ";
        }
        cout<<endl;
      }

      //full one
      for(j = 1; j<=n; j++){
          cout<<"*";
      }cout<<endl;

      //middle - 2
      for(i = (n/2 + 2); i<= n-1; i++){
          for(j = 1; j<=n/2; j++){
              cout<<" ";
          }

          cout<<"*";

          for(j = n/2 + 2; j<=n-1; j++){
           cout<<" ";
        }

        cout<<"*";
        cout<<endl;
      }

      //last row
          for(j = 1; j<=(n/2 + 1); j++){
              cout<<"*";
          }

          for(j = n/2 + 2; j<=n-1; j++){
           cout<<" ";
        }

        cout<<"*";
        cout<<endl;

      return 0;
}
```

```
#include<iostream>
using namespace std;
int main()
{
    int t;
    cin>>t;
    int c1,c2,c3,c4,n,m;

    int rick[1005],cab[1005];

    while(t--){
        cin>>c1>>c2>>c3>>c4;

        cin>>n>>m;
        for(int i=0;i<n;i++){
            cin>>rick[i];
        }

        for(int i=0;i<m;i++){
            cin>>cab[i];
        }

        int rickcost = 0;
        for(int i=0;i<n;i++){
            rickcost += min(c1*rick[i],c2);
        }
        rickcost = min(rickcost,c3);

        int cabcost = 0;
        for(int i=0;i<m;i++){
            cabcost += min(c1*cab[i],c2);
        }
        cabcost = min(cabcost,c3);

        int finalAns = min(c4,rickcost+cabcost);
        cout<<finalAns<<endl;
    }
    return 0;
}
```

```
//Factorial of huge numbers

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

int main(){
    int n;
    cin>>n;

    int arr[10000];
    arr[0] = 1;
    int carry = 0;
    int noOfDigs = 1;
    int x;

    for(int i = 1; i<=n; i++){
        for(int j = 0; j<noOfDigs; j++){
            x = arr[j]*i + carry;
            arr[j] = x%10;
            carry = x/10;
        }

        while(carry > 0){
            arr[noOfDigs] = carry%10;
            carry = carry/10;
            noOfDigs++;
        }
    }
    for(int i = noOfDigs-1; i>=0; i--){
        cout<<arr[i];
    }
    return 0;
}
```

```
#include<iostream>
#include<math.h>
using namespace std;

bool isArmstrong(int num){
    int temp = num,
        sum = 0,
        power;

    power = std::to_string(temp).length();
    /*
        std::to_string(integer) coverts our number into a string so as to
operate upon it various string
        functionalities
    */
    // cout<<power;

    while(num>=1){
        sum = sum + pow((num%10), power);
        num = num/10;
    }
    return(sum == temp);
}

int main(){
    int n;
    cin>>n;

    isArmstrong(n)?cout<<"true":cout<<"false";
    return 0;
}
```

```
#include<iostream>
#include<math.h>
using namespace std;

//checks if it is perfect square or not
bool isSqrt(int num){
    int s = sqrt(num);
    return(s*s == num);
}

//checks if it the two terms 5n^2 + 4 are perfect squares or not
bool isFibonacci(int n){
    return isSqrt(5*n*n + 4) || isSqrt(5*n*n - 4);
}

int main(){
    //the main principle behind whether a number is a fibonacci series
element or not is,
    // say the number is n, it is in fibonacci, if and only if, one or
both, (5n^2 + 4) and (5n^2 - 4) are perfect squares.

     int number;

     cout<<"Enter the number which you want to search inside the series:
";
     cin>>number;

     isFibonacci(number)? cout<<"Yes the number is present in series." :
                          cout<<"No, it is not present in the list.";
}
```

```
//KEDANE'S ALGORITHM

/*
5
1 -3 2 1 -1

ans = 3
*/

#include <iostream>
using namespace std;

int main() {
    int n ;
    cin>>n;

    int i, arr[100];
    for(i = 0; i<n; i++){
        cin>>arr[i];
    }

    //ans is the temporary variable.
    int currSum = 0, ans = 0;

    int maxSum = arr[0], localMax = arr[0];

    for(i = 1; i<n; i++){
        localMax = arr[i];
        currSum = localMax + maxSum;
        maxSum = max(localMax, currSum);
        ans = max(ans, maxSum);
    }

    cout<<ans;
      return 0;
}
```

```
/*
1
11
111
1001
11111
100001
1111111
10000001
*/

#include <iostream>
using namespace std;

int main() {
      //take the number of rows
      int n, i, j;
      cin >>n;
      //start the looping through each row
      //if i%2 = 0, then set all the entries as 1s
      //else 1, followed by 0s and then again 1.

      for(i = 1; i<=n; i++){
          if(i%2 == 1){
              for(j = 1; j<=i; j++){
                  cout<<"1";
              }
          }else{
              cout<<"1";
              for(j = 1; j<=(i-2); j++){
                  cout<<"0";
              }cout<<"1";
          }
          cout<<endl;
      }
      return 0;
}
```

```
/*
1
11
202
3003
40004
500005
6000006
70000007
*/

#include <iostream>
using namespace std;

int main() {
      int n;
      cin>>n;

      cout<<"1"<<endl;

      for(int i = 1; i<n; i++){
          cout<<i;

          for(int j =0; j<i-1; j++){
              cout<<"0";
          }

          cout<<i<<endl;
      }
      return 0;
}
```

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

int largest(int arr[], int n)
{
    int max = arr[0];

    for (int i = 1; i < n; i++){
        if (arr[i] > max){
            max = arr[i];}
        }

    return max;
}


int main()
{
    int n, arr[100];

    cin>>n;

    for(int i = 0; i<n; i++){
        cin>>arr[i];
    }

    cout<< largest(arr, n);
    return 0;
}
```

```
//which row or columnmn has the maximum sum, and what is that max sum.

#include <iostream>
using namespace std;

int maxElement(int a[]);

int main() {
      int arr[100][100], i, j, row_Sum[10], column_Sum[10],m,n;

      cin>>m;
      cin>>n;

      for(i = 0; i<m; i++){
          for(j = 0; j<n; j++){
              cin>>arr[i][j];
          }
      }

      for(i = 0; i<m; i++){
          for(j = 0; j<n; j++){
              row_Sum[i] += arr[i][j];
          }
      }

      for(j = 0; j<n; j++){
          for(i = 0; i<m; i++){
              column_Sum[j] += arr[i][j];
          }
      }

    int maxR = maxElement(row_Sum);
    int maxC = maxElement(column_Sum);

    int answer = (maxR > maxC)?maxR:maxC;

    cout<<answer;

      return 0;
}

int maxElement(int a[]){
    int largest = 0, i;

    int length = sizeof(a)/sizeof(a[0]);

    for(i = 0; i<length; i++){
        if(a[i] > largest){
            largest = a[i];
        }
    }

    return largest;
}
```

```
/*
Input:
9
-4 1 3 -2 6 2 -8 -9 4

Output:
The maximum sum is: 10
1   5
The subarray with maximum element sum is:
1 3 -2 6 2
The length of the subarray is: 5

*/

#include <iostream>
using namespace std;

int main() {
      int n, a[100], k, j, i;

      int currSum = 0,
          maxSum = 0,
          left, right = -1;

      cin>>n;

      for(i = 0; i<n; i++){
          cin>>a[i];
      }

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

              //this is being updated for every subarray, hence
reinitialise currSum = 0
              currSum = 0;

              for(k = i; k<=j; k++){
                  currSum += a[k];
              }
              if(currSum > maxSum){
                  maxSum = currSum;
                  left = i;
                  right = j;
              }
          }
      }

      cout<<"The maximum sum is: "<<maxSum<<endl;
      cout<<left<<"   "<<right<<endl;

      cout<<"The subarray with maximum element sum is: \n";
      for(k = left; k<=right; k++){
          cout<<a[k]<<" ";
```

```
        }

        cout<<endl;

        cout<<"The length of the subarray is: "<<right + 1 - left<<endl;
        return 0;
}
```

```
/*
9
-4 1 3 -2 6 2 -8 -9 4

The maximum sum of any subarray is: 10
1 3 -2 6 2
The length of this subarray is: 5

*/
#include <iostream>
using namespace std;

int main() {
      int n, i, j, arr[100];
      int cumSum[100], maxSum = 0, currentSum = 0, left = -1, right = -1;

      cin>>n;

      for(i = 0; i<n; i++){
          cin>>arr[i];
      }

      cumSum[0] = arr[0];

      for(i = 1; i<n; i++){
          cumSum[i] = cumSum[i-1] + arr[i];
      }

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

              currentSum = 0;
              currentSum = cumSum[j] - cumSum[i-1];

              if(currentSum > maxSum){
                   maxSum = currentSum;
                   left = i;
                   right = j;
              }
          }
      }

      cout<<"The maximum sum of any subarray is: "<<maxSum<<endl;

      for(i = left; i<=right; i++){
          cout<<arr[i]<<" ";
      }

      cout<<endl<<"The length of this subarray is: "<<right + 1 -
left<<endl;

      return 0;
}
```

```
/*
9
-4 1 3 -2 6 2 -8 -9 4

The maximum sum is: 10

*/

#include<iostream>
using namespace std;

int main(){
    int n, i, currSum = 0, max = 0, arr[100];
    cin>>n;

    for(i = 0; i<n; i++){
        cin>>arr[i];
    }

    //Kedane's A;gorithm for maximum sub array sum
    for(i = 0; i<n; i++){
        currSum += arr[i];

        if(currSum < 0){
            currSum = 0;
        }

        if(currSum > max){
            max = currSum;
        }
    }

    cout<<"The maximum sum is: "<<max;

    return 0;
}
```

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

int main(){
    /*

    1    ____1
    2    ___123
    3    __12345
    4    _1234567
    5    123456789

    i REPRESENTS THE ROW NUMBER

    the number of spaces has to be given by (n-i)
    the number if digits printed is (2i - 1)
    followed by endl

    */

    int n;
    cin>>n;

    int i = 1;
    while(i<=n){

        int space = 1;
        while(space <= n-i){
            cout<<" ";
            space++;
        }

        int num = 1;
        while(num <= (2*i - 1)){
            cout<<num;
            num++;
        }

        cout<<endl;
        i++;
    }
    return 0;
}
```

```
/*

            1
        2 1   1 2
    3 2 1       1 2 3
4 3 2 1           1 2 3 4
    3 2 1       1 2 3
        2 1   1 2
            1

*/

#include<iostream>
using namespace std;
int main() {
    int n;
    cin>>n;
    for(int i=1;i<=(n+1)/2;i++){
        for(int j=1;j<=(2*(n+1-2*i));j++){
            cout<<" ";
        }
        for(int j = i;j>0;j--){
            cout<<j<<" ";
        }

        for(int j=1;j<(4*(i-1)-1);j++){
            cout<<" ";
        }
        if(i>1){
        for(int j=1;j<=i;j++){
            cout<<j<<" ";
        }
    }
        cout<<endl;
    }
    for(int i=n/2;i>0;i--){
         for(int j=1;j<=(2*(n+1-2*i));j++){
            cout<<" ";
        }
        for(int j = i;j>0;j--){
            cout<<j<<" ";
        }

        for(int j=1;j<(4*(i-1)-1);j++){
            cout<<" ";
        }
        if(i>1){
        for(int j=1;j<=i;j++){
            cout<<j<<" ";
        }
        }
        cout<<endl;
    }
      return 0;
```

```
}
```

```
/*

        *********
       *        *
      *        *
     *        *
    *        *
   *        *
  *        *
 *        *
*********

*/
#include <iostream>
using namespace std;

int main() {
      int n, i ,j;
      cin>>n;

      for(i = 1; i <= n; i++){
          for(j = 1; j<=n-i; j++){
              cout<<" ";
          }

          if(i == 1 || i == n){
              for(j = 1; j<=n; j++){
                  cout<<"*";
              }
          }
          else{
              cout<<"*";

              for(j = 1; j<=n-2; j++){
                  cout<<" ";
              }

              cout<<"*";
          }
          cout<<endl;
      }
      return 0;
}
```

```
/*
*   *   *   *   *   *   *   *   *
*   *   *   *       *   *   *   *
*   *   *               *   *   *
*   *                       *   *
*                               *
*   *                       *   *
*   *   *               *   *   *
*   *   *   *       *   *   *   *
*   *   *   *   *   *   *   *   *

*/

#include<iostream>

using namespace std;

int main()
{
     int i,j,k=0,n;
     cin>>n;
     for(i=0;i<n;i++)
     {
          if(i==0||i==n-1)
          {
               for(j=0;j<n;j++)
                    cout<<"*\t";
          }
          else if(i<=n/2)
          {
               for(j=0;j<(n/2)-i+1;j++)
               cout<<"*\t";
               for(j=0;j<2*i-1;j++)
               cout<<"\t";
               for(j=0;j<(n/2)-i+1;j++)
               cout<<"*\t";
               if(i==n/2)
                    k=2;
          }
          else
          {
               for(j=0;j<k;j++)
                    cout<<"*\t";
               for(j=0;j<n-(2*k);j++)
                    cout<<"\t";
               for(j=0;j<k;j++)
                    cout<<"*\t";
               k++;
          }
          cout<<"\n";
     }
}
```

```
/*

1 2 3 4 5 6 7
1 2 3 4 5 6 *
1 2 3 4 5 * * *
1 2 3 4 * * * * *
1 2 3 * * * * * * *
1 2 * * * * * * * * *
1 * * * * * * * * * * *

*/

#include <iostream>
using namespace std;

int main() {
      int n,i,j;
      cin>>n;

      for(i = 1; i<=n; i++){
          cout<<i<<" ";
      }cout<<endl;

      for(i = 1; i<n; i++){
          for(j = 1; j<=n-i; j++){
              cout<<j<<" ";
          }

          for(j = 1; j<=(2*i - 1); j++){
              cout<<"* ";
          }

          cout<<endl;
      }
      return 0;
}
```

```
/*

            1
           2     3     2
      3    4     5     4     3
4     5    6     7     6     5     4

*/

#include <iostream>
using namespace std;

int main() {
      int n,i,j,k;
      cin>>n;

      for(i = 1; i<=n; i++){
          for(j = 1; j<= (n-i); j++){
              cout<<"\t";
          }

          for(k = i; k<=(2*i - 1); k++){
              cout<<k<<"\t";
          }k -= 2;

          for( ; k>=i; k--){
              cout<<k<<"\t";
          }

          cout<<endl;
      }
      return 0;
}
```

```
/*
1
2     2
3     0     3
4     0     0     4
5     0     0     0     5
*/

#include <iostream>
using namespace std;

int main() {
      int n,i,j;
      cin>>n;

      cout<<"1"<<endl;

      for(i = 1; i <= n-1; i++){
          cout<<i+1<<"\t";
          for(j = 1; j<i; j++){
              cout<<"0"<<"\t";
          }
          cout<<i+1<<"\t"<<endl;
      }
      return 0;
}
```

```
/*

5                   5
5 4               4 5
5 4 3           3 4 5
5 4 3 2       2 3 4 5
5 4 3 2 1   1 2 3 4 5
5 4 3 2 1 0 1 2 3 4 5
5 4 3 2 1   1 2 3 4 5
5 4 3 2       2 3 4 5
5 4 3           3 4 5
5 4               4 5
5                   5

*/
#include<iostream>
using namespace std;


int main(){

      int n;
      cin>>n;


      for(int i=n;i>=1;i--){
            for(int j=n;j>=i;j--){
                  cout<<j<<" ";
            }

            for(int j=2*i-1;j>=1;j--){
                  cout<<"  ";
            }

            for(int j=i;j<=n;j++){
                  cout<<j<<" ";
            }
            cout<<endl;
      }


    for(int i=n;i>=0;i--)
        cout<<i<<" ";
    for(int i=1;i<=n;i++)
        cout<<i<<" ";
    cout<<endl;


      for(int i=1;i<=n;i++){
            for(int j=n; j>=i;j--){
                  cout<<j<<" ";
            }

            for(int j=1;j<=2*i-1;j++){
```

```
                  cout<<"  ";
            }

            for(int j=i;j<=n;j++){
                  cout<<j<<" ";
            }
            cout<<endl;
      }
      return 0;
}
```

```
/*

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

*/

#include <iostream>
using namespace std;

int main() {
      int n,i,k,j;
      cin>>n;

      int rows = n;

      for(i = 1; i <= n; i++){
          for(j = 1; j <= i; j++){
              cout<<j<<"\t";
          }j--;

          for(k = 1; k<=(2*rows - 3); k++){
              cout<<"\t";
          }rows--;

          if(i != n){
              for(;j>0; j--){
                  cout<<j<<"\t";
              }
          }

          else if(i == n){
              j--;
              for(;j>0; j--){
                  cout<<j<<"\t";
              }
          }

          cout<<endl;
      }
      return 0;
}
```

```
//PREFIX SUM 2D MATRIX BRUTE FORCE

/*
input:
3 3
10 20 30
5 10 20
2 4 6


output:
10 30 60
15 45 95
17 51 107

*/

#include <iostream>
using namespace std;

int main() {
      int i, j, m, n, arr[100][100];

      cin>>m;
      cin>>n;

      for(i = 0; i<m; i++){
          for(j = 0 ; j<n; j++){
              cin>>arr[i][j];
          }
      }

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

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


      for(i = 0; i<m; i++){
          for(j = 0 ; j<n; j++){
              cout<<arr[i][j]<<" ";
          }
          cout<<endl;
      }

      return 0;
}
```

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

bool primeOrNot(int num){

    if(num<=1){
        return false;
    }

    for(int i = 2; i<=num/2; i++){
        if(num % i == 0){
            return false;
        }else{
            return true;
        }
    }
}

int main(){
    int n;
    cin>>n;

    primeOrNot(n) ? cout<<"Prime" : cout<<"Not Prime";
    return 0;
}
```

```
//print reverse of a number

#include<iostream>
using namespace std;

int main(){
    long long int num;
    cin>>num;

    while(num>=1){
        int rev = num%10;
        cout<<rev;
        num = num/10;
    }
    return 0;
}
```

```
/*
 Question is :

Take the following as input.

A number (N1)
A number (N2)
Write a function which prints first N1 terms of the series 3n + 2 which
are not multiples of N2.


*/

#include<iostream>
using namespace std;

int main() {
    int N1, N2;
    cin>>N1>>N2;
    int i = 1;
    int counter = 0;

    while(counter< N1){
        if((3*i + 2)% N2 != 0){
            cout<<(3*i + 2)<<endl;
            counter++;
        }
        i++;
    }
      return 0;
}
```

```
//pythaTriplets 1.0

#include <iostream>
using namespace std;

int main() {
      int n;
      cin>>n;

      if(n==1 || n==2){
          cout<<"-1";
      }

      /*
        for pythagorean triplets,

        if n is even,
            the triplets could be given by,
             n^2/4-1, n and n^2/4+1
             so if a variable say, var = n^2/4-1, then

             a = var
             b = n
             c = var + 2

        if n is odd,
            the triplets could be given by,
             (n^2-1)/2, n and (n^2+1)/2
             so if a variable say, var = (n^2-1)/2, then

             a = var
             b = n
             c = var + 1
      */

      if(n%2 == 0){
          //for even
          int var = (n*n)/4 - 1;
          cout<<var<<"    "<<var+2;
      }

      if(n%2 != 0){
          //for odd
          int var = (n*n-1)/2;
          cout<<var<<"    "<<var+1;
      }
      return 0;
}
```

```
//PythaTriplets 2.0

#include <iostream>
using namespace std;

int main() {
      int num;
      cin>>num;

      if(num == 1 || num == 2){
          cout<<"-1";
      }

      /*
        for pythagorean triplets,

        if n is even,
            the triplets could be given by,
             n^2/4-1, n and n^2/4+1
             so if a variable say, var = n^2/4-1, then

             a = var
             b = n
             c = var + 2

        if n is odd,
            the triplets could be given by,
             (n^2-1)/2, n and (n^2+1)/2
             so if a variable say, var = (n^2-1)/2, then

             a = var
             b = n
             c = var + 1
      */

      else if(num%2==0){
            long m=num/2;
            long n=1;
            cout<<(m*m-n*n)<<" "<<(m*m+n*n);
        }else{
            long m=(num + 1)/2;
            long n=(num - 1)/2;
            cout<<(2*m*n)<<" "<<(m*m+n*n);
        }
      return 0;
}
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main() {
   int a, b, c;
   cin>>a>>b>>c;

   float d, x1, x2;
   if (a == 0) {
      cout << "This is not a quadratic equation, enter the coefficients
again.";
   }else {
      d = b*b - 4*a*c;
      if (d > 0) {
         x1 = (-b - sqrt(d)) / (2*a);
         x2 = (-b + sqrt(d)) / (2*a);
         cout << "Real and Distinct" << endl<<x1<<" "<<x2;
      } else if (d == 0) {
            x2 = (-b + sqrt(d)) / (2*a);
         cout << "Real and Equal" << endl<<x2<<" "<<x2;
      }else {
         cout<<"Imaginary Roots";
      }
   }
   return 0;
}
```

```
/*

10
0 2 1 3 0 1 2 1 2 1

The amount of water that could be stored is equal to: 5

*/
#include <iostream>
using namespace std;

int main() {
      int towerCount, heights[100], right[100], left[100], units = 0, i;
      cin>>towerCount;

      for(i = 0; i<towerCount; i++){
          cin>>heights[i];
      }

      //leftmost ones
      left[0] = heights[0];
      for(i = 1; i<towerCount; i++){
          left[i] = max(left[i-1], heights[i]);
      }

      //rightmost ones
      right[towerCount-1] = heights[towerCount-1];
      for(i = towerCount-2; i>=0; i--){
          right[i] = max(right[i+1], heights[i]);
      }

      for(i = 0; i<towerCount; i++){
          units+= (min(left[i], right[i]) - heights[i]);
      }
      cout<<"The amount of water that could be stored is equal to:
"<<units;
      return 0;
}

/*

PS: This question was a son of a bitch. Took literally an hour and half
to get the code right.

*/
```

```
//rotate 90 anti

#include <iostream>
using namespace std;

void transpose(int **arr, int **trans, int size){
    for(int i = 0; i<size; i++){
        for(int j = 0; j<size; j++){
            trans[i][j] = arr[j][i];
        }
    }
}


int main(int argc, char *argv[]) {
  int size;

  cin>>size;

  int **arr = new int*[size];
  for(int i = 0; i<size; i++){
      arr[i] = new int[size];
  }

  int **trans = new int*[size];
    for(int i = 0; i<size; i++){
        trans[i] = new int[size];
    }

  for(int i = 0; i<size; i++){
      for(int j = 0; j<size; j++){
          cin>>arr[i][j];
      }
  }

    transpose(arr, trans, size);

    for(int i = size-1; i>=0; i--){
        for(int j = 0; j<=size-1; j++){
            cout<<trans[i][j]<<" ";
        }
        cout<<endl;
    }
  return 0;
}
```

```
/*
    Given a list of numbers, stop processing input after the cumulative
sum of all the input becomes negative.
*/

#include<iostream>
using namespace std;
int main(){
    int n,sum=0;
    while(true){
     cin>>n;
     sum=sum+n;
     if(sum>=0){
         cout<<n<<endl;
     }
     else{
         break;
     }
    }
}
```

```
#include <iostream>
#include<algorithm>

using namespace std;

bool compare(int a, int b){
    return a>b;
}

int main() {
      int a[] = {2,40,5,61,48,89,1,56};
      int n = sizeof(a)/sizeof(a[0]);

      sort(a,a+n, compare);  //by default the nature of sorting is
ascending
                             //in order to customize this, we used a
comparator function.

      for(int i = 0; i<n; i++){
          cout<<a[i]<<" ";
      }
      return 0;
}
```

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

void spiralPrintAnti(int **arr, int r, int c){
    int i,j;

    int startRow = 0,
          endRow = r-1,
          startCol = 0,
          endCol = c-1;

      while(startRow <= endRow && startCol <= endCol){

          //firstCol
          for(i = startRow; i<=endRow; i++){
              cout<<arr[i][startCol]<<", ";
          }
          startCol++;

          //lastRow
          for(j = startCol; j<=endCol; j++){
              cout<<arr[endRow][j]<<", ";
          }
          endRow--;

          //lastCol
          for(i = endRow; i>=startRow; i--){
              cout<<arr[i][endCol]<<", ";
          }
          endCol--;

          //first row
          for(j = endCol; j>=startCol; j--){
              cout<<arr[startRow][j]<<", ";
          }
          startRow++;
      }
      cout<<"END";
}

int main(int argc, char *argv[]) {
      int r,c;

      cin>>r>>c;

      int **arr = new int*[r];
      for(int i = 0; i<r; i++){
          arr[i] = new int[c];
      }

      for(int i = 0; i<r; i++){
          for(int j = 0; j<c; j++){
              cin>>arr[i][j];
          }
```

```
        }

        spiralPrintAnti(arr, r, c);
        return 0;
}
```

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

int main(){
    //take the number and the number of places up to which the precision
of root is desired
    int n, p;
    cin>>n>>p;

    int ans = 1, times = 0;
    float inc = 1;
    /*
        everything happening for calculating the face value of the root
is repeated p number of times
        the increment is dynamic, since it depends on the order of the
precision.
        for 1 decimal place, 0.1
        for 2, 0.01
        for 3, 0.001
        that means we need to divide this increment by 10, after every
iteration
    */
    while(times <= p){

        while(ans*ans <= n){
            ans = ans + inc;
        }
        ans = ans - inc;
        inc = inc/10;
        times++;
    }
    cout<<ans<<endl;
    return 0;
}
```

```
/*
4 4
1 4 6 9
5 10 12 13
8 14 15 16
19 20 22 23
14

The element is found at location ( 2, 1 )
*/

#include <iostream>
using namespace std;

void staircaseSearch(int a[100][100], int R, int C, int key){
    //start with (0, C-1)th element
    //return inside a void function, throws the compiler right outta the
function.
    int i, j;
    i=0, j= C-1;

    while(i<R&&j>=0){
        if(key == a[i][j]){
            cout<<"The element is found at location"<<" ( "<<i<<",
"<<j<<" )"<<endl;
            return; //breaks you right out of this function.
                    //break could not be used here, because it wil cause
output to be 10.
                    //because it will break you out of just this loop,
then cout<<0 will also function.
        }
        else if(key>a[i][j]){
            i++;
        }
        else{
            j--;
        }
    }
    cout<<"Not found";
    return;
}

int main(){
      int arr[100][100], R, C, elem;

      cin>>R;
      cin>>C;

      for(int i = 0; i<R; i++){
          for(int j = 0; j<C; j++){
              cin>>arr[i][j];
          }
      }
```

```
        cin>>elem;

        staircaseSearch(arr, R, C, elem);
        return 0;
}
```

```
/*
30
Apple
hahah
hy my name is
yo
*/

#include <iostream>
#include <cstring>    //for STL
#include <algorithm> //for sort method
using namespace std;

int main() {
    string s2 = "it is not about lyrics anymore";
    cout<<s2.length()<<endl;

      string s1[] = {"hy my name is", "Apple", "yo", "hahah"};
      sort(s1, s1+4);   //sort(arr, arr+n);    where n is the size of the
string array.

      for(int i=0; i<4; i++){
          cout<<s1[i]<<endl;
      }
      return 0;
}
```

```
//find out the dual subarrays summing to a specific key.

#include <iostream>
using namespace std;

int main() {
    int n, key, arr[100];
    cin>>n;

    for(int i = 0; i<n; i++){
        cin>>arr[i];
    }

    cin>>key;

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

            int sum = 0;
            sum = arr[i] + arr[j];

            if(key == sum){
                if(arr[i]<arr[j]){
                    cout<<arr[i]<<" and "<<arr[j]<<endl;
                }
                else if(arr[j]<arr[i]){
                    cout<<arr[j]<<" and "<<arr[i]<<endl;
                }
            }
        }
    }
      return 0;
}
```

```
//find out the triplets subarrays summing to a specific key.

#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

int main() {
      int n;
    cin>>n;

    vector<int> arr(n);

    for(int i=0; i<n; i++)
            cin>>arr[i];

      int target;
    cin>>target;

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

    for(int i=0; i<n; i++)
      {
            int temp = target - arr[i];
            int start = i+1;
            int end = n-1;
            while(start<end)
            {
                  if(arr[start] + arr[end] > temp)
                        end--;
                  else if(arr[start]+arr[end] < temp)
                        start++;
                  else
                  {
                        cout<< arr[i] << ", " << arr[start] << " and " <<
arr[end] << endl;
                        start++;
                        end--;
                  }
            }
      }
      return 0;
}
```

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

//typecasting
// decreasing order of prioirty of datatypes, float>int>char>boolean
int main(){
int a = 3;
char ch = 'A';

//IMPLICIT
//expected output is 68 because here the + operator typecasts the char ch
as an integer, causing the two vars to get added.
cout<<(a+ch)<<endl;

int a1 = 5,b = 8;
float avg, avg2;

avg = (a1+b)/2;
avg2 = (a1+b)/2.0;
//expected output is 6 (not 6.5, even though avg is of type float)
because, (a1 + b) and 2 are of type int, and int/int => int
//in avg2, int/float gives a float response
cout<<avg<<endl<<avg2<<endl;

//EXPLICIT
cout<<int(ch)<<endl<<bool(a1);
return 0;
}
```

```
#include <iostream>
#include<math.h>
using namespace std;

int main() {
      int n;
      cin>>n;

      for(int i = 0; i<n; i++){
          string bin;
          cin>>bin;

          int count = 0;

          for(int j = 0; j<bin.length(); j++){
              count = count + ((bin[j] - '0')*pow(2, (bin.length() -
(j+1))));
          }
          cout<<count<<endl;
      }
      return 0;
}

//Von neumann loves binary
```

```
// wave print, col wise

#include <iostream>
using namespace std;

int main() {
      int r, c, i, j, arr[100][100];

      cin>>r;
      cin>>c;

      for(i = 0; i<r; i++){
          for(j = 0; j<c; j++){
              cin>>arr[i][j];
          }
      }

      for(j = 0; j<c; j++){
          if(j%2 == 0){
              for(i = 0; i<r; i++){
                  cout<<arr[i][j]<<", ";
              }
          }
          else{
              for(i = r-1; i>=0; i--){
                  cout<<arr[i][j]<<", ";
              }
          }
      }
      cout<<"END";
      return 0;
}
```