1. Infix To Postfix

```
def infixTOpostfix(input1):
    OP = set(['+', '-', '*', '/', '(', ')', '^'])
    PR = {'+':1, '-':1, '*':2, '/':2, '^':3}
    s = []
    ans = ''
    for ch in input1:
        if ch not in OP:
            ans+= ch
        elif ch=='(':
            s.append('(')
        elif ch==')':
            while s and s[-1]!= '(':
                ans+=s.pop()
            s.pop()
        else:
            while s and s[-1]!='(' and PR[ch]<=PR[s[-1]]:
                ans+=s.pop()
            s.append(ch)
    while s:
        ans+=s.pop()
    return ans
input1=input()
print(infixTOpostfix(input1))
```

2. Longest Decreasing Subsequent :

```
def lds(arr, n):
    lds = [0] * n
    max = 0
    lds[i] = 1

    for i in range(1, n):
        for j in range(i):
            if (arr[i] < arr[j] and
                lds[i] < lds[j] + 1):
                lds[i] = lds[j] + 1

    for i in range(n):
        if (max < lds[i]):
            max = lds[i]

    # returns the length of the LDS
    return max
```

3. Longest Increasing Subsequent :

```java
public class LongestIncreasingSubsequent
{
    /* lis() returns the length of the longest
    increasing subsequence in arr[] of size n */
    static int lis(int arr[],int n)
    {
        int lis[] = new int[n];
        int i,j,max = 0;

        /* Initialize LIS values for all indexes */
        for ( i = 0; i < n; i++ )
            lis[i] = 1;

        /* Compute optimized LIS values in
            bottom up manner */
        for ( i = 1; i < n; i++ )
            for ( j = 0; j < i; j++ )
                        if ( arr[i] > arr[j] &&
                                lis[i] < lis[j] + 1)
                    lis[i] = lis[j] + 1;

        /* Pick maximum of all LIS values */
        for ( i = 0; i < n; i++ )
            if ( max < lis[i] )
                max = lis[i];

            return max;
    }

    public static void main(String args[])
    {
        int arr[] = { 41, 18467, 6334, 26500, 19169 };
            int n = arr.length;
            System.out.println(lis( arr, n ));
    }
}
```

4. Longest Common Subsequent :

NextSmalles... HighestDist... PrintPalind... FirstNonRep... Pow

```java
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
public class LongestCommonSubsequent {
    public static String lcs(String str1, String str2)
    {
        int l1 = str1.length();
        int l2 = str2.length();
        int[][] arr = new int[l1 + 1][l2 + 1];

        for (int i = l1 - 1; i >= 0; i--)
        {
            for (int j = l2 - 1; j >= 0; j--)
            {
                if (str1.charAt(i) == str2.charAt(j))
                    arr[i][j] = arr[i + 1][j + 1] + 1;
                else
                    arr[i][j] = Math.max(arr[i + 1][j], arr[i][j + 1]);
            }
        }
        int i = 0, j = 0;
        StringBuffer sb = new StringBuffer();
        while (i < l1 && j < l2)
        {
            if (str1.charAt(i) == str2.charAt(j))
            {
                sb.append(str1.charAt(i));
                i++;
                j++;
            }
            else if (arr[i + 1][j] >= arr[i][j + 1])
                i++;
            else
                j++;
        }
        return sb.toString();
    }
    public static void main(String[] args) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("\nEnter string 1");
        String str1 = br.readLine();
        System.out.println("\nEnter string 2");
        String str2 = br.readLine();
        String result = lcs(str1, str2);
        System.out.println("\nLongest Common Subsequence : "+ result);
    }
}
```

```python
//Longest Common Subsequent
def lcs(input1,input2):
    m = len(input1)
    n = len(input2)

    L = [[None]*(n + 1) for i in range(m + 1)]

    for i in range(m + 1):
        for j in range(n + 1):
            if i == 0 or j == 0 :
                L[i][j] = 0
            elif input1[i-1] == input2[j-1]:
                L[i][j] = L[i-1][j-1]+1
            else:
                L[i][j] = max(L[i-1][j], L[i][j-1])
    return L[m][n]
```

5. Longest Palindrome Subsequent :

```
int longestPalindromeSubseq(string s) {
    int n=s.length();
    int dp[n+1][n+1];
    memset(dp,0,sizeof(dp));
    for(int i=0;i<n;i++)
      dp[i][i]=1;
    for(int l=1;l<=n;l++)
      for(int j=0;j<n-l+1;j++)
        dp[j][j+l]=s[j]==s[j+l]?2+dp[j+1][j+l-1]:max(dp[j][j+l-1],dp[j+1]
[j+l]);
    return dp[0][n];
  }
```

6:07 PM

6. Number of Selective Arrangement :

Language: C

Compiler : gcc 5.4.0

```c
#include<stdio.h>
#include<string.h>
// Read only region start

int arrangements(int input1)
{
    // Read only region end
    // Write code here
    if(input1==0)
     return 1;
    if(input1==1)
     return 0;
    if(input1==2)
     return 1;
    return(input1-1)*(arrangements(input1-1)+arrangements(input1-2));

}
```

7. Number of Puzzle :

```
#include <bits/stdc++.h>
using namespace std;
int helper(int n, int arr[])
{
    int penalty=0;
    sort(arr,arr+n);
    for(int i=1;i<n;i++)
        penalty+=abs(arr[i]-a
    return penalty;
}
int main(){
    int n;
    cin>>n;
    int a[n];
    int i;
    for(i=0;i<n;i++)
        cin>>a[i];
    cout<<helper(n,a);
    return 0;
}
```

8. Number of Puzzle :

Square.cpp | floor.cpp | stack_book.cpp | Max Sub array.cpp | Number puzzle Healper.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;
int helper(int n, int arr[])
{
    int penalty=0;
    sort(arr,arr+n);
    for(int i=1;i<n;i++)
        penalty+=abs(arr[i]-ar
    return penalty;
}
int main(){
    int n;
    cin>>n;
    int a[n];
    int i;
    for(i=0;i<n;i++)
        cin>>a[i];
    cout<<helper(n,a);
    return 0;
}
```

"C:\Users\Aer...

```
4
1 6 -2 4
8
Process retur
ned 0 (0x0)
```

9. The Cuckoo Sequent:

cuckoo.cpp

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin>>n;

    int cuckoo[n];
    int value = 1;
    cuckoo[1]= 0;
    cuckoo[2]= 1;

    for(int i = 3 ; i <= n ; i++)
    {
        cuckoo[i] =  i * cuckoo[i-1];
        value = value + cuckoo[i];
    }

    cout<<value;


    return 0;

}
```

10. Bitwise Operation :

gp.cpp | bitwise.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n;

    cin>>n;
    int a = 0;
    int b = -1;
    int c = -1;
    int i = 0;

    while(n>0)
    {
        if(n&1)
        {
            a++;

            if(b == -1)
            {
                b = i;
            }

            c = i;
        }

        i++;

        n = n >> 1;
    }

    cout<<a<<" "<<b<<" "<<c;
}
```

Resources | Compile Log

12. Ceaser Cipher :

```c
#include<stdio.h>

int main()
{
    char message[100], ch;
    int i, key;

    printf("Enter a message to encrypt: ");
    gets(message);
    printf("Enter key: ");
    scanf("%d", &key);

    for(i = 0; message[i] != '\0'; ++i){
        ch = message[i];

        if(ch >= 'a' && ch <= 'z'){
            ch = ch + key;

            if(ch > 'z'){
                ch = ch - 'z' + 'a' - 1;
            }

            message[i] = ch;
        }
        else if(ch >= 'A' && ch <= 'Z'){
            ch = ch + key;

            if(ch > 'Z'){
                ch = ch - 'Z' + 'A' - 1;
            }

            message[i] = ch;
        }
    }

    printf("Encrypted message: %s", message);

    return 0;
}
```

13. Next Greater Number :

```
from itertools import permutations
input1 = input()
input2 = int(input())
l1 = list(permutations(str(input2)))
l2 = []
str1 = ""
for i in range(len(l1)):
    for j in range(len(l1[0])):
        str1 += l1[i][j]
    l2.append(int(str1))
    str1 = ""
l2.sort()
print(str(l2[1]))
```

14. Reminder Mod 11

```python
from bisect import bisect_right,bisect_left
def solve():
    n = int(input())
    l = [int(i) for i in input().split()][:n]
    x = [0 for i in range(n + 1)]
    len_ = 1
    x[0] = l[0]
    for i in range(1,n):

        if l[i] < x[0]:
            x[0] = l[i]

        elif l[i] > x[len_ - 1]:
            x[len_] = l[i]
            len_ += 1

        else:
            a = bisect_left(x[:len_],l[i])
            x[a] = l[i]

    print(len_)
solve()
```

Remainder mod 11 java

```java
import java.util.*;
public class Main
{
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int num = scan.nextInt();
        int result = num % 11;
        System.out.println(result);
    }
}
```

15. Moving Apple :

Language: PYTHON3 Compiler : Python 3.6

```python

# Read only region start
class UserMainCode(object):
    @classmethod
    def moveApples(cls, input1, input2):
        '''
        input1 : int
        input2 : int[]

        Expected return type : int
        '''
        # Read only region end
        avg = sum(input2)//input1
        result = 0
        for i in range(input1):
            result+=abs(avg - input2[i])
        return result//2
```

16. GP :

```
#include <bits/stdc++.h>
using namespace std;
int main() {

    double second;

    double third;

    int n;

    cin>>second>>third>>n;

    double r = third/second;

    double a = second/r;

    double nth = a * pow( r , n-1 );

    cout<<nth;


    return 0;

}
```

17. Character Count :

```
l1 = [int(i) for i in input().split(",")]
l2 = [int(i) for i in input().split(",")]
count = 0
for i in l1:
    for j in l2:
        if i == j:
            count += 1
    print(f"{i}-{count}")
    count = 0
```

18. Moving Apple :