

1. Infix To Postfix

```
382 def infixToPostfix(input1):
383     OP = set(['+', '-', '*', '/', '(', ')', '^'])
384     PR = {'+':1, '-':1, '*':2, '/':2, '^':3}
385     s = []
386     ans = ''
387     for ch in input1:
388         if ch not in OP:
389             ans+= ch
390         elif ch=='(':
391             s.append('(')
392         elif ch==')':
393             while s and s[-1]!='(':
394                 ans+=s.pop()
395             s.pop()
396         else:
397             while s and s[-1]!='(' and PR[ch]<=PR[s[-1]]:
398                 ans+=s.pop()
399             s.append(ch)
400     while s:
401         ans+=s.pop()
402     return ans
403 input1=input()
404 print(infixToPostfix(input1))
```

2. Longest Decreasing Subsequent :

```
382 def lds(arr, n):  
383     lds = [0] * n  
384     max = 0  
385     lds[i] = 1  
386  
387     for i in range(1, n):  
388         for j in range(i):  
389             if (arr[i] < arr[j] and  
390                 lds[i] < lds[j] + 1):  
391                 lds[i] = lds[j] + 1  
392  
393     for i in range(n):  
394         if (max < lds[i]):  
395             max = lds[i]  
396  
397     # returns the length of the LDS  
398     return max
```

3. Longest Increasing Subsequent :

```
2 public class LongestIncreasingSubsequent
3 {
4     /* lis() returns the length of the longest
5     increasing subsequence in arr[] of size n */
6     static int lis(int arr[],int n)
7     {
8         int lis[] = new int[n];
9         int i,j,max = 0;
10
11         /* Initialize LIS values for all indexes */
12         for ( i = 0; i < n; i++ )
13             lis[i] = 1;
14
15         /* Compute optimized LIS values in
16         bottom up manner */
17         for ( i = 1; i < n; i++ )
18             for ( j = 0; j < i; j++ )
19                 if ( arr[i] > arr[j] &&
20                     lis[i] < lis[j] + 1)
21                     lis[i] = lis[j] + 1;
22
23         /* Pick maximum of all LIS values */
24         for ( i = 0; i < n; i++ )
25             if ( max < lis[i] )
26                 max = lis[i];
27
28         return max;
29     }
30
31     public static void main(String args[])
32     {
33         int arr[] = { 41, 18467, 6334, 26500, 19169 };
34         int n = arr.length;
35         System.out.println(lis( arr, n )); |
36     }
37 }
```

4. Longest Common Subsequent :

```
NextSmalles... HighestDist... PrintPalind... FirstNonRep... Pow
1 import java.io.BufferedReader;
2 import java.io.InputStreamReader;
3 import java.io.IOException;
4 public class LongestCommonSubsequent {
5     public static String lcs(String str1, String str2)
6     {
7         int l1 = str1.length();
8         int l2 = str2.length();
9         int[][] arr = new int[l1 + 1][l2 + 1];
10
11         for (int i = l1 - 1; i >= 0; i--)
12         {
13             for (int j = l2 - 1; j >= 0; j--)
14             {
15                 if (str1.charAt(i) == str2.charAt(j))
16                     arr[i][j] = arr[i + 1][j + 1] + 1;
17                 else
18                     arr[i][j] = Math.max(arr[i + 1][j], arr[i][j + 1]);
19             }
20         }
21         int i = 0, j = 0;
22         StringBuffer sb = new StringBuffer();
23         while (i < l1 && j < l2)
24         {
25             if (str1.charAt(i) == str2.charAt(j))
26             {
27                 sb.append(str1.charAt(i));
28                 i++;
29                 j++;
30             }
31             else if (arr[i + 1][j] >= arr[i][j + 1])
32                 i++;
33             else
34                 j++;
35         }
36         return sb.toString();
37     }
38     public static void main(String[] args) throws IOException
39     {
40         BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
41         System.out.println("\nEnter string 1");
42         String str1 = br.readLine();
43         System.out.println("\nEnter string 2");
44         String str2 = br.readLine();
45         String result = lcs(str1, str2);
46         System.out.println("\nLongest Common Subsequence : " + result);
47     }
48 }
```

```
1 //Longest Common Subsequent
2 def lcs(input1,input2):
3     m = len(input1)
4     n = len(input2)
5
6     L = [[None]*(n + 1) for i in range(m + 1)]
7
8     for i in range(m + 1):
9         for j in range(n + 1):
10             if i == 0 or j == 0 :
11                 L[i][j] = 0
12             elif input1[i-1] == input2[j-1]:
13                 L[i][j] = L[i-1][j-1]+1
14             else:
15                 L[i][j] = max(L[i-1][j], L[i][j-1])
16     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

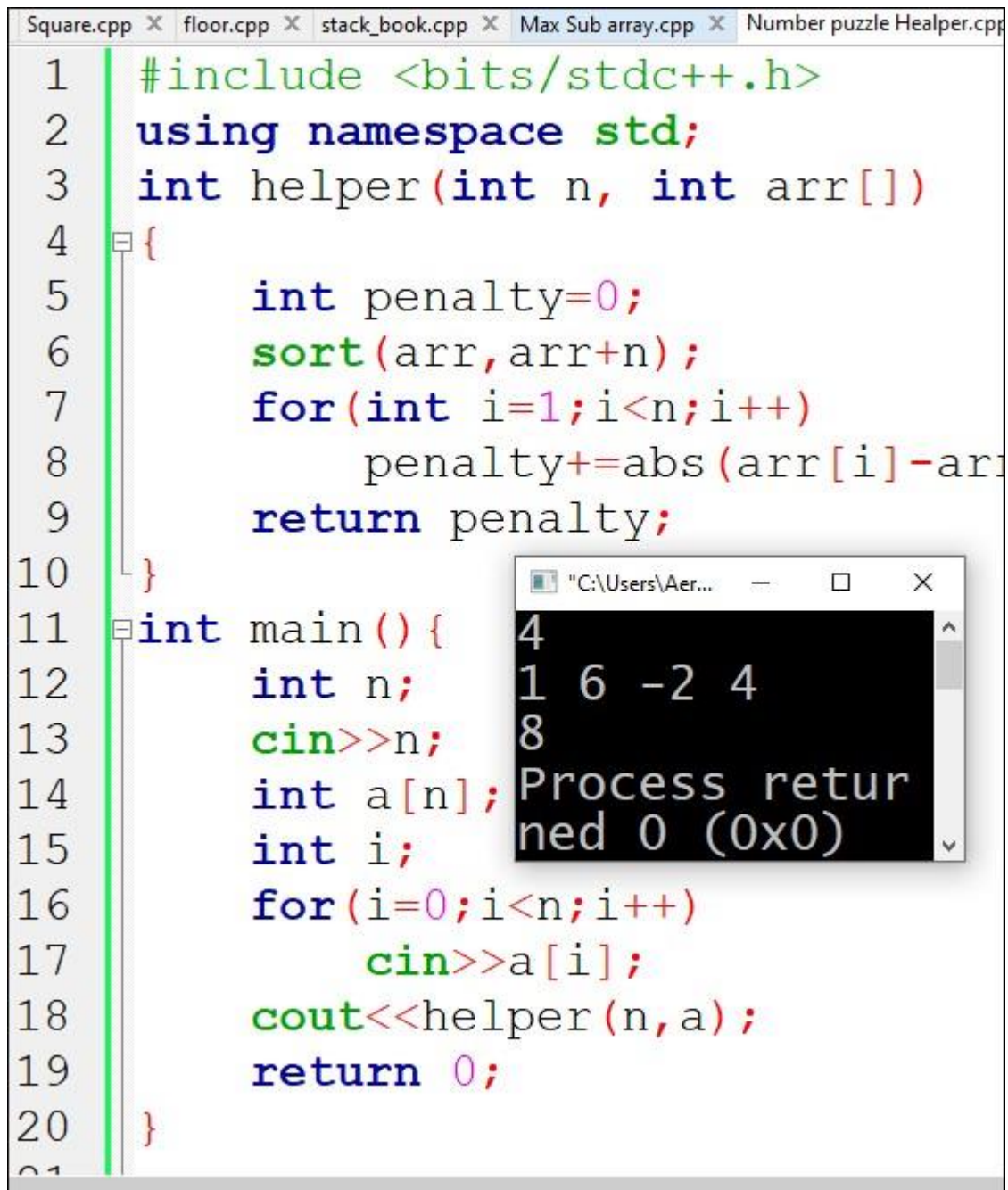
6. Number of Selective Arrangement :

```
Language: C Compiler : gcc 5.4.0
1  #include<stdio.h>
2  #include<string.h>
3  // Read only region start
4
5  int arrangements(int input1)
6  {
7      // Read only region end
8      // Write code here
9      if(input1==0)
10         return 1;
11         if(input1==1)
12             return 0;
13         if(input1==2)
14             return 1;
15         return(input1-1)*(arrangements(input1-1)+arrangements(input1-2));
16
17 }
```

7. Number of Puzzle :

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int helper(int n, int arr[])
4  {
5      int penalty=0;
6      sort(arr,arr+n);
7      for(int i=1;i<n;i++)
8          penalty+=abs(arr[i]-a
9      return penalty;
10 }
11 int main() {
12     int n;
13     cin>>n;
14     int a[n];
15     int i;
16     for(i=0;i<n;i++)
17         cin>>a[i];
18     cout<<helper(n,a);
19     return 0;
20 }
```


8. Number of Puzzle :

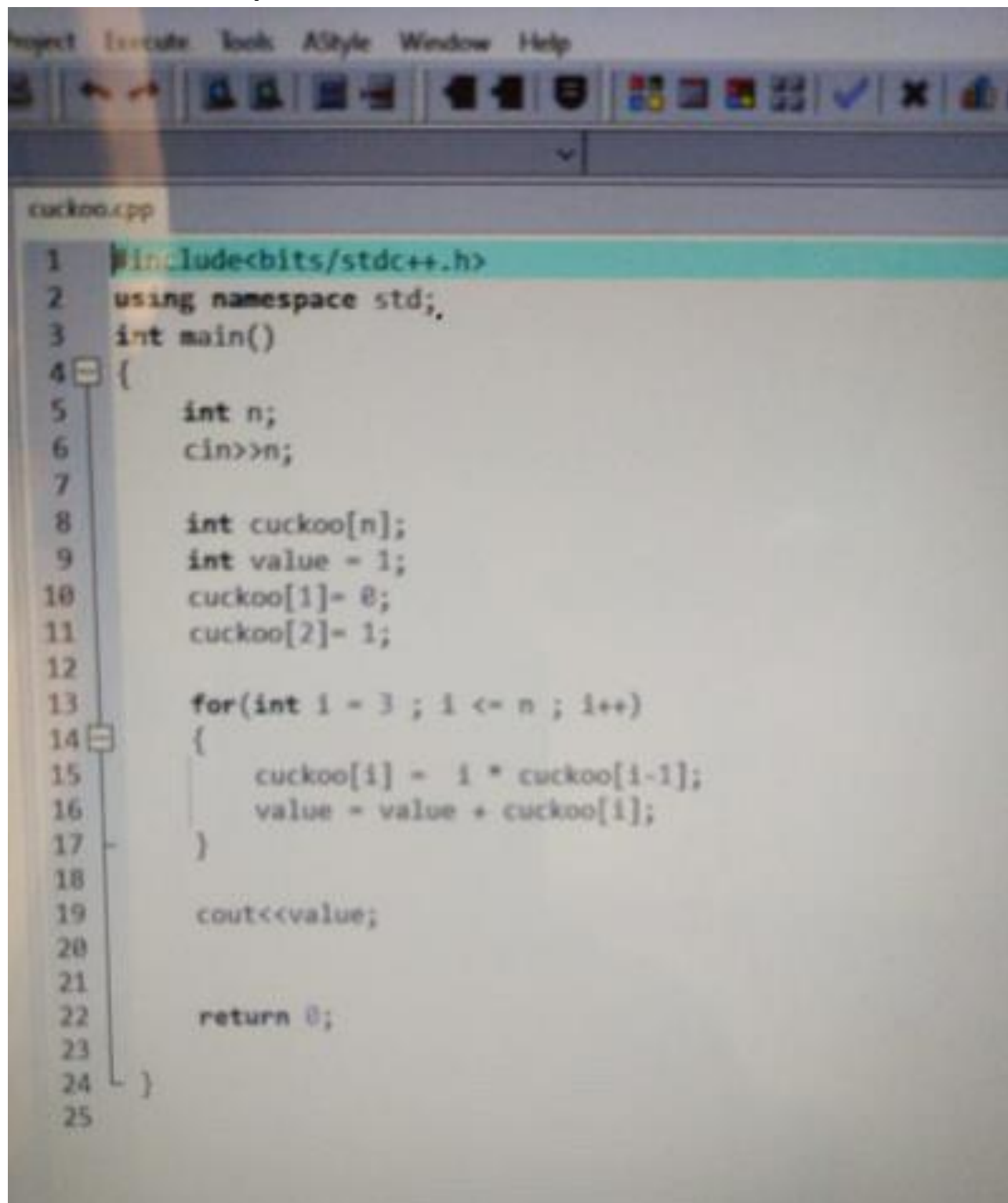


```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int helper(int n, int arr[])
4  {
5      int penalty=0;
6      sort(arr, arr+n);
7      for(int i=1; i<n; i++)
8          penalty+=abs(arr[i]-arr[i-1]);
9      return penalty;
10 }
11 int main() {
12     int n;
13     cin>>n;
14     int a[n];
15     int i;
16     for(i=0; i<n; i++)
17         cin>>a[i];
18     cout<<helper(n, a);
19     return 0;
20 }
```

Console Output:

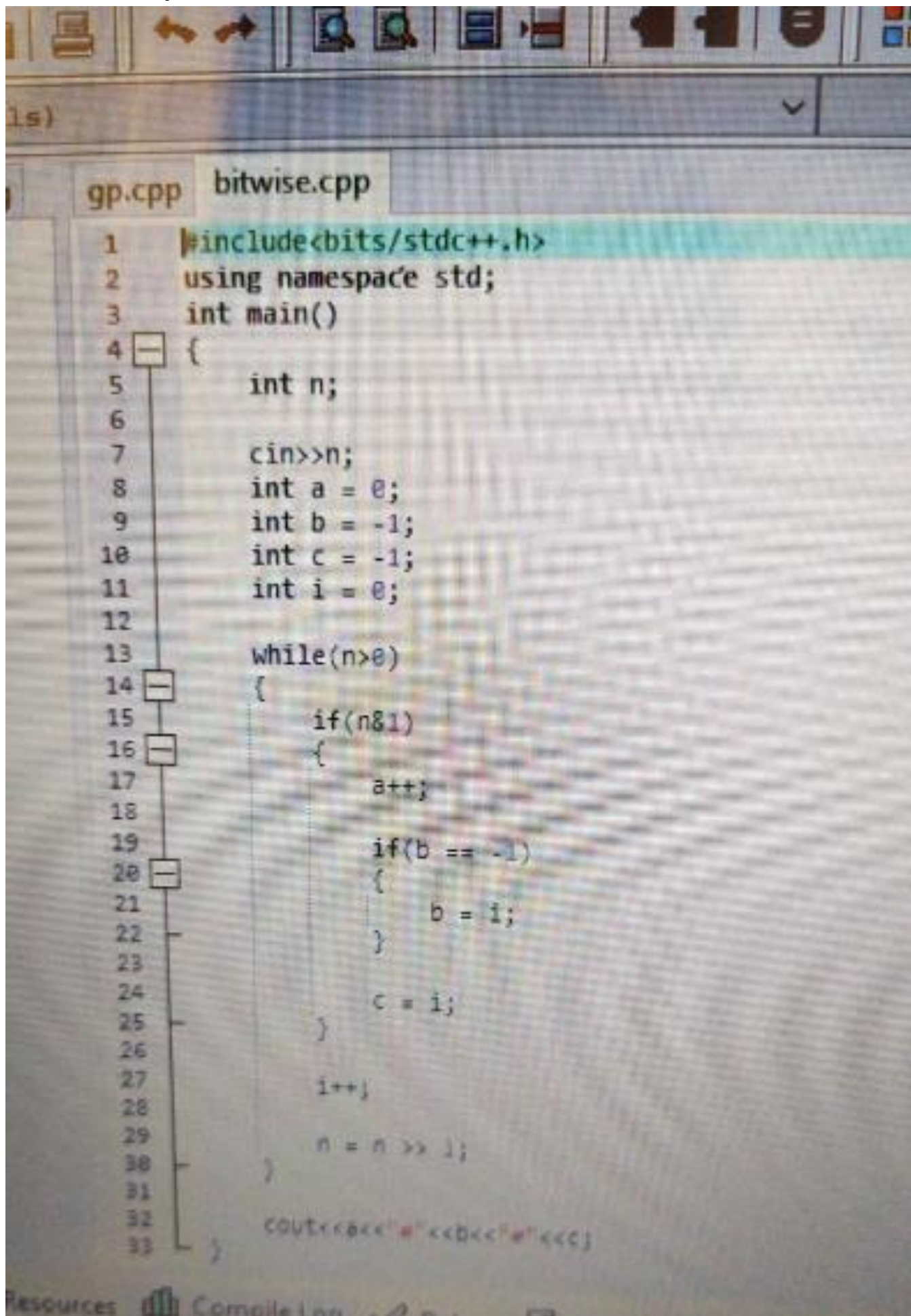
```
4
1 6 -2 4
8
Process returned 0 (0x0)
```


9. The Cuckoo Sequent:

A screenshot of a C++ IDE window titled 'cuckoo.cpp'. The code is as follows:



```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main()
4  {
5      int n;
6      cin>>n;
7
8      int cuckoo[n];
9      int value = 1;
10     cuckoo[1]= 0;
11     cuckoo[2]= 1;
12
13     for(int i = 3 ; i <= n ; i++)
14     {
15         cuckoo[i] = i * cuckoo[i-1];
16         value = value + cuckoo[i];
17     }
18
19     cout<<value;
20
21
22     return 0;
23 }
24
25
```

10. Bitwise Operation :



The image shows a screenshot of a C++ IDE with a toolbar at the top and a code editor window. The code editor has two tabs: 'gp.cpp' and 'bitwise.cpp'. The 'bitwise.cpp' tab is active, displaying a C++ program. The program includes the header <bits/stdc++.h>, uses the std namespace, and defines a main function. Inside main, it declares an integer n, reads input from cin, and initializes variables a, b, c, and i. A while loop runs as long as n is greater than 0. Inside the loop, it checks if the least significant bit of n is 1 (n & 1). If true, it increments a, and if b is -1, it sets b to i. It then increments i and right-shifts n by 1. After the loop, it prints the values of a, b, and c.

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main()
4  {
5      int n;
6
7      cin>>n;
8      int a = 0;
9      int b = -1;
10     int c = -1;
11     int i = 0;
12
13     while(n>0)
14     {
15         if(n&1)
16         {
17             a++;
18
19             if(b == -1)
20             {
21                 b = i;
22             }
23
24             c = i;
25         }
26         i++;
27         n = n >> 1;
28     }
29
30     cout<<a<<" "<<b<<" "<<c<<endl;
31 }
32
33
```

Resources  Compile Log  Run

12. Ceaser Cipher :

```
#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

```
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

```
1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scan = new Scanner(System.in);
6         int num = scan.nextInt();
7         int result = num % 11;
8         System.out.println(result);
9     }
10 }
11
```

15. Moving Apple :

```
Language: PYTHON3 ▾ Compiler : Python 3.6

1
2 # Read only region start
3 class UserMainCode(object):
4     @classmethod
5     def moveApples(cls, input1, input2):
6         ...
7         input1 : int
8         input2 : int[]
9
10        Expected return type : int
11        ...
12        # Read only region end
13        avg = sum(input2)//input1
14        result = 0
15        for i in range(input1):
16            result+=abs(avg - input2[i])
17        return result//2
18        |
19
20
```

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 :