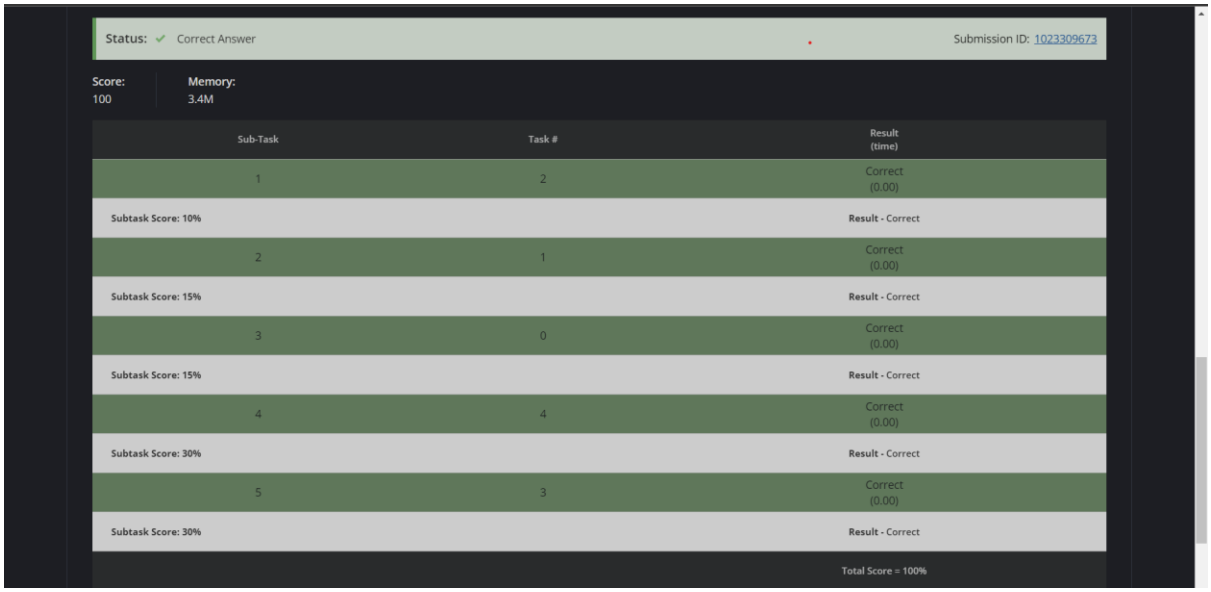
1)Create String.

SOL) #include <iostream>

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

int main() {
    string s1, s2;
    int T;
    cin>>T;
    while(T){
    cin >> s1;
    cin >> s2;
    if(s1.length()%s2.length()!=0){
        printf("No");
        return 0;
    }

    int H1[26] = {0};
    int H2[26] = {0};

    for (int i = 0; i < s1.length(); i++) {
        if (isalpha(s1[i])) {
            H1[toupper(s1[i])-'A']++;
        }
    }

    for (int i = 0; i < s2.length(); i++) {
        if (isalpha(s2[i])) {
            H2[toupper(s2[i]) - 'A']++;
        }
    }

    int c = 0;
    for (int i = 0; i < 26; i++) {
        if (H1[i] >= H2[i]) {
            c++;
        }
    }

    if (c == 26) {
        cout << "Yes" << endl;
    } else {
        cout << "No" << endl;
    }
    T--;
    }

    return 0;
}
```

APPROACH:This code will work wor all the letters not just for C,O,K,F.Usinig two hash arrays to strore the frequencies of each letter (using ascii values) in the rwo give string if..if str2 contains all the letters in str1..then str1 can be created from str2..provided that the len(str2)%len(str1)==0,since all the letters of the string2 should be used in permutaion to form string1;



2)Add powers of 2

SOL) #include<iostream>

#include<cmath>

using namespace std;

```
int main(){
   int T;
   cin>>T;
   while(T--){
      int n,k;
      cin>>n>>k;

      int c=0;

      if(k>n){
         int m=k-n;
         while(m>0){
            if(m&1){
               c++;
            }
```

```
            m=m>>1;
          }
        }
        cout<<c<<endl;
    }
    return 0;
}
```

APPROACH: First I am subtracting n from k to get the result(m) as the sum of powers of two. Then I am using binary operetors to count the number of 1's in m's binary form which will give the number of powers of 2 that have to be added to get m.

3)Staircase

SOL) #include<iostream>

#include<vector>

#include<stack>

using namespace std;

int main() {

    int T;

    cin>>T;

```
while(T--) {
    int n;
    cin>>n;
    stack<int>st;
    vector<int>arr(n);

    for(int i=n;i>=1;i--) {
        st.push(i);
    }

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

    int c=0;

    for(int i=0;i<2;i++) {
        for(int j=0;j<n;j++) {
            if(arr[j]==st.top()) {
                st.pop();
            }
            if(st.empty()) {
                c=1;
                break;
            }
        }
        if(c==1) {
            break;
        }
    }
```

```
    if(c==1) {
       cout<<"Yes";
    }
    else {
       cout<<"No";
    }
  }

  return 0;
}
```

APPROACH:creating a stcak to store the sizes of the blocks with 1 at the top and n at the bottom of the stack.String the sizes of the block inn the order in ehich the chef gets it ina vector array.Since chef can go near the blocks only two times I am traversing through the array twice and if the element in the array is equal to the top of the stack I am popping the stack which shows that the chef is able to collect the blocks in sorted manner…if the stack is emty that mean the staircase is constructable in only two trips. EXAMPLE: arr={1,4,3,2};st=[1,2,3,4]

1)1 is at the top and also the first block size that the chef gets so it is popped.Similary for 2 as well.

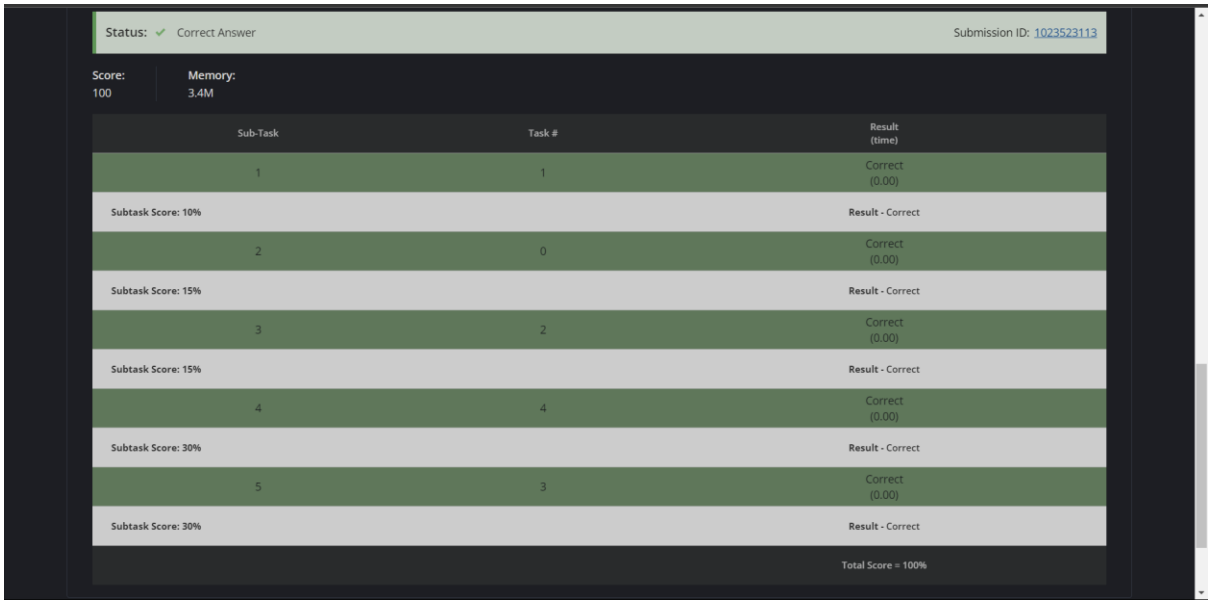
2)Next block size is 4 but 4 is not at the top.

3)Similarly for block size 3.

4)Next block size is 2 and the top is also 2 so it is popped.Now the chef has completed the first trip and go back with bloc 1 and 2.

5)in the next trip..chef can only collect 3 since 3 is at the top..

6)chef cannot complete the staifcase as he could not collect all the block in the two trips.



6)BITWISE XOR

SOL) #include <iostream>

#include <vector>

using namespace std;

```
int main() {
    int T;
    cin>>T;

    while(T>0){
        int n;
        cin>>n;
        vector<int>arr(n);
        vector<int>r(2);
        for(int i=0;i<n;i++) {
            cin>>arr[i];
        }
        cin>>r[0]>>r[1];
        int len =arr.size();
        int x=r[0];
```

```
    int s=r[1];
    int c=0;

    for(int i=0;i<len-1;i++) {
       for(int j=i+1;j<len;j++) {
          if((arr[i]^arr[j])==x && (arr[i]+arr[j])==s) {
             c++;
          }
       }
    }

    cout<<c<<endl;
    T--;
  }

  return 0;
}
```

APPROACH:traversing through the array and calculating the sum an d xor of all the pairs inn the array if it is equal to the given required sum and xor value then increasing the counter by 1.