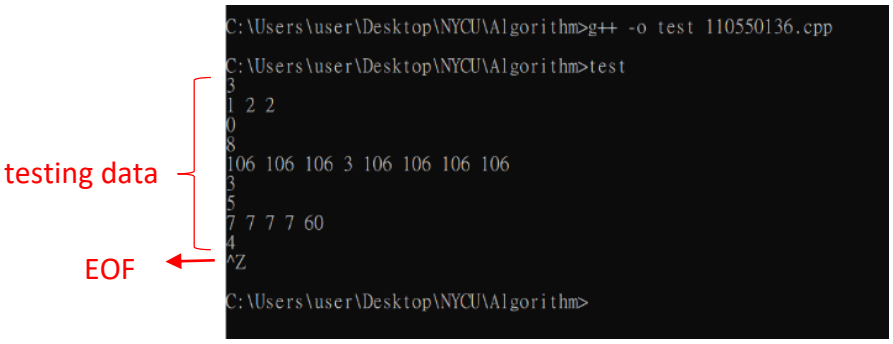
Ex1 - Divide and Conquer

- **Environment**

- Windows
- g++
- Dev C++ 5.11
- How to run my program: (in CMD)



- **Algorithm**

```
Find_Fake (n, start, arr, length)
/* int n: number of coins ; int s: starting index
   int* arr: an array storing the weights of coins ; int length: length of arr */
1.  if n==1                                              // termination
        then the fake coin is start
2.  else if n==2
3.      if start.weight!=real weight
            then the fake coin is start
4.      else the fake coin is start+1
5.  else partition arr into 3 even groups: A, B, C       // divide
6.      if A.weight==B.weight==C.weight                  // the fake coin is not in A, B, C
7.          if n%3==2                                    // the fake coin is the last two coin
                Find_Fake(2, start+3*k, arr, length)
8.          else the fake coin is start+3*k              // the fake coin is the last coin
9.      else if A.weight==B.weight                       // the fake coin is in C
            Find_Fake(k, start+2*k, arr, length) // examine C
10.     else if A.weight==C.weight                       // the fake coin is in B
            Find_Fake(k, start+k, arr, length)   // examine B
11.     else                                             // the fake coin is in A
            Find_Fake(k, start, arr, length);    // examine A
```

- **Time Complexity**

- Divide - Θ(1) ... Line5
- Conquer - T(n/3) ... Line6~11
- Combine - none

➔ T(n)= T(n/3)+ Θ(1)

➔ By master method, T(n)= Θ(lgn)

- **My Ideas**

I interpreted the idea of “scale” as comparing the two sides without knowing their actual weight. Therefore, in each recursion, I divide the group into three even subgroups, and see whether or not they are “balanced”(equal). Pick out the subgroup that is weighted differently and repeat the steps until the fake one is found.