All programs will be executed using: “g++ <name_of_the_program>”

Question 1

A program is given that prints a set of ALL gray codes of n-digits. The program takes a number as input (n) via std::cin.
Assume n can be ANY natural number not bigger than 20.
Example:- If n=2, then the output should be =>

```
00
01
11
10
```

Note:-

1) Google the definition of gray code, or look here: https://en.wikipedia.org/wiki/Gray_code
2) You have to debug the code in “buggy_program_1.cpp”.

Question 2

I was given a file called “marksheet.csv” containing students' marks against their first names. My job was to arrange those entries so that students with higher marks appear upper on the list. Two students who have the same marks are arranged alphabetically. But my code is giving unwanted segmentation faults. Can you help me debug it?
Example: Say the marksheet.csv contains =>

```
Kavya 95
Saksham 91
Guramrit 92
Aditya 92
```

The expected output should be =>

```
Kavya 95
Aditya 92
Guramrit 92
Saksham 91
```

Note:-

1) There will be no two people with the same first name.
2) “marksheet.csv” should be in the same directory as the buggy code.
3) Some parts in the buggy code are strictly NOT to be modified. It’s mentioned in the code. The final debugged code should have those parts untouched.
4) Code uses “ifstream” for file handling; there is no need to worry about that.
5) Take very special care of the functions used from the “string” module of C++.
6) You have to debug the code in “buggy_program_2.cpp”.

Question 3

Consider an algorithm that takes as input a positive integer n. If n is even, the algorithm divides it by two, and if n is odd, the algorithm multiplies it by three and adds one. The algorithm repeats this until n is one. Example for n=3:-

3->10->5->16->8->4->2->1

Looks like a very simple algorithm, right? But my code was *somehow* failing for numbers like 159487, 270271, 665215, 704511, etc.

Input is a natural number n (<= 10^6) via std::cin, and you have to output ALL the numbers in each step of that algorithm with n.

Note:-

1) For n=159487, a buggy output file, “wrong_output.txt” and a correct output file, “expected_output.txt” is provided.
2) You have to debug the code in “buggy_program_3.cpp”.

Question 4

There are n apples with known weights. My task was to divide the apples into two groups so that the difference between the weights of the groups was minimal and only **print that minimal difference**. My idea was to calculate sums of ALL possible subsets of these weights and find the one CLOSEST to half of the total sum of the weights of all apples. But my code is giving me a logical error.

We take a “counter” variable from 0 to 2^n to go through ALL possible subsets. The binary representation of a particular “counter” variable value uniquely represents a subset.
For example:- For n=5, counter = 11 (binary form = “01011”) has “1” at 1st, 2nd, 4th positions (counting from right to left), so it represents the subset of 1st, 2nd, 4th weights.
So counter=0 represents null set and counter=2^n represents complete set.

To check whether 0 or 1 is present for a bit in a number, we use the **bitwise AND “&”** operator. If I wanted to know the value of 3rd bit of 11 (counted from right to left), I use 11&4. So as 3rd bit of 11 is 0, 11&4 = 0. Hence, to know the ith bit of n, we use n&(2^{i-1}).

The first input line has an integer n: the number of apples. The next line has n integers: p_1, p_2, p_3, … p_n the weight of each apple.

Example: If the input is =>
5
3 2 1 7 4
The output should be => 1

Explanation: Group 1 has weights 2, 3, and 4 (total weight 9), and group 2 has weights 1 and 7 (total weight 8). So, the expected output is 9-8 = 1.

Note:-

1) $1 <= n <= 20$ and $1 <= p_i <= 10^9$
2) You have to debug the code in “buggy_program_4.cpp”.