Programming Problems

1. Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz"

2. Write a program that prints the squares of all integers from 1 to 100.

Sample output:

```
1       1
2       4
...
10      100
```

3. Children are taught to add multi-digit numbers from right to left, one digit at a time. Many find the "carry operation", where 1 is carried from one digit position to the next, to be a significant challenge. Your job is to count the number of carry operations for each set of addition problems so that educators may assess their difficulty.

Input:

The input will be two unsigned integers with less than 10 digits.

Output:

Compute the number of carry operations that result from adding the two numbers and print them in the format shown.

Sample Input		Sample Output
123	456	No carry operations
555	555	3 carry operations

4. Hashmat is a brave warrior who with his group of young soldiers moves from one place to another to fight against his opponents. Before Fighting he just calculates one thing, the difference between his soldier number and the opponent's soldier number. From this difference he decides whether to fight or not. Hashmat's soldier number is never greater than his opponent.

(problem #4 continued)

Input:

The input contains two numbers: These two numbers denotes the number of soldiers in Hashmat's army and his opponent's army or vice versa.

Output:

Print the difference of the number of soldiers between Hashmat's army and his opponent's army.

Sample Input		Sample Output
10	12	2
10	14	4
100	200	100

5. Computing the area of a rectangle is a common operation in geometry and computer graphics. The input to your program will be the length and width of rectangles, and the output will contain the area of the rectangles.

Input:

Each line will consist of two values, the length and width of the rectangle, respectively. The length and width of these rectangles will be integer values between 1 and 100. The last line of the input will be when the two integers are 0.

Output:

Print out a single line of text for each rectangle that contains the area of the rectangle. The area should be preceded by the string "#x: ", where x is the line number.

Sample Input	Sample Output
68 42 1 2 5 5 0 0	#1: 2856 #2: 2 #3: 25

6. Write a predicate function **IsSorted(array, n)** that takes an integer array and its effective size as parameters and returns **true** if the array is sorted in nondecreasing order.

In C++ the function definition would be like this:

```
bool IsSorted(int arr[], int size);
```

7. You are given a sorted sequence of n integers $S = s_1, s_2, ..., s_n$ and a sorted sequence of m integers $Q = q_1, q_2, ..., q_m$. Please, print in the ascending order all such s_i that does not belong to Q.

Input:

In the first line you are given one integer, n, such that $2 <= n <= 100$, and in the following line n integers:
$-100 <= s_i <= 100, \quad s_i <= s_{i+1}$

In the third line you are given one integer, m, such that $2 <= m <= 100$, and in the following line m integers:
$-100 <= q_i <= 100, \quad q_i <= q_{i+1}$

(problem #7 continued)

Output:

The sequence of requested integers separated by spaces.

Sample Input	Sample Output
5 -2 -1 0 1 4 6 -3 -2 -1 1 2 3	0 4

8. In the third century B.C., the Greek astronomer Eratosthenes developed an algorithm for finding all the prime numbers up to some upper limit N. To apply the algorithm, you start by writing down a list of the integers between 2 and N. For example, if N were 20, you would begin by writing down the following list:

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Next you circle the first number in the list, indicating that you have found a prime. You then go through the rest of the list and cross off every multiple of the value you have just circled, since none of those multiples can be prime. Thus, after executing the first step of the algorithm, you will have circled the number 2 and crossed off every multiple of two, as follows:

From this point, you simply repeat the process by circling the first number in the list that is neither crossed off nor circled, and then crossing off its multiples. In this example, you would circle 3 as a prime and cross off all multiples of 3 in the rest of the list, which would result in the following state:

Eventually, every number in the list will either be circled or crossed out, as shown in this diagram:

The circled numbers are the primes; the crossed-out numbers are composites. This algorithm for generating a list of primes is called the **sieve of Eratosthenes.**

Write a program that uses the sieve of Eratosthenes to generate a list of the primes between 2 and 1000.

Sources of the programming problems:

Problem # 1 – **Fizz Buzz Test** – http://c2.com/cgi/wiki?FizzBuzzTest

Problem # 3 – **Primary Arithmetic** – https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=12&page=show_problem&problem=976

Problem # 4 – **Hashmat the Brave Warrior** – https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=996

Problem # 5 – **Area of Rectangle** – from trial problems of a programming competition

Problem # 6 – **IsSorted function** – a problem from the course reader of Stanford's "CS 106B – Programming Abstractions" (https://see.stanford.edu/Course/CS106B) (page 78)

Problem # 7 – **Fun with Sequences** – http://www.spoj.com/problems/SMPSEQ3/

Problem # 8 – **Sieve of Eratosthenes** – another problem from the course reader of Stanford's "CS 106B – Programming Abstractions" (https://see.stanford.edu/Course/CS106B) (page 78)

NOTE: you can download the CS106B course reader from - http://www.cas.mcmaster.ca/~qiao/courses/cs2s03/textbook/

– – – – – – – – –