Question - 1

FizzBuzz

SCORE: **50 points**

Easy Loops Problem Solving Core Skills

Given a number *n*, for each integer *i* in the range from *1* to *n* inclusive, print one value per line as follows:

- If *i* is a multiple of both *3* and *5*, print *FizzBuzz*.
- If *i* is a multiple of *3* (but not *5*), print *Fizz*.
- If *i* is a multiple of *5* (but not *3*), print *Buzz*.
- If *i* is not a multiple of *3* or *5*, print the value of *i*.

Function Description

Complete the function *fizzBuzz* in the editor below. The function must print the appropriate response for each value $i \in \{1, 2, \dots n\}$ in ascending order, each on a separate line. There is no return value expected.

fizzBuzz has the following parameter(s):

n: upper limit of values to test (inclusive)

Constraints

- $0 < n < 2 \times 10^5$

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The single integer *n*, the limit of the range to test: [1, 2, ...n].

▼ Sample Case 0

Sample Input 0

```
15
```

Sample Output 0

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
```

Help

```
14
FizzBuzz
```

Explanation 0

The numbers *3*, *6*, *9*, and *12* are multiples of *3* (but not *5*), so print *Fizz* on those lines.

The numbers *5* and *10* are multiples of *5* (but not *3*), so print *Buzz* on those lines.

The number *15* is a multiple of both *3* and *5*, so print *FizzBuzz* on that line.

None of the other values is a multiple of either *3* or *5*, so print the value of *i* on those lines.

Question - 2
Binary Search

SCORE: **5 points**

Binary search involves taking a *sorted* array and then successively focussing our interest on half of the current array until we find (or don't find) the element we are looking for.
What is the complexity of this algorithm where *N* is the length of the original sorted array?

○ O(N)

● O(log N)

○ O(N log N)

○ O(N/2)

Question - 3
Time Complexity

SCORE: **5 points**

What is the time complexity of the following code:

```
int a = 0, i = N;
while (i > 0) {
   a += i;
   i /= 2;
}
```

○ O(N)

○ O(Sqrt(N))

○ O(N / 2)

● O(log N)

Question - 4
Order of Growth

SCORE: **5 points**

Which of the given options provides the increasing order of time complexity of functions f1, f2, f3, and f4?

f1(n) = O(2^n)
f2(n) = O(n^(3/2))
f3(n) = O(nLogn)
f4(n) = O(n^3)

● f3, f2, f4, f1

○ f3, f2, f1, f4

○ f2, f3, f1, f4

○ f2, f3, f4, f1