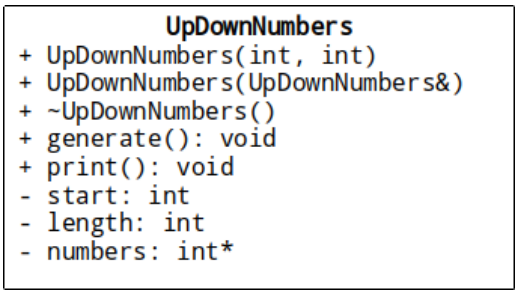
CENG241 Labwork 6

1. Implement the following **UpDownNumbers** class:



   You must also at least implement the **setStart** function. Feel free to implement other missing functions (default constructor, getters/setters, etc) if necessary.

   This class has a dynamic integer array, which starts from a number the user specifies and should be consecutively filled by *length* amount of numbers according to the following:

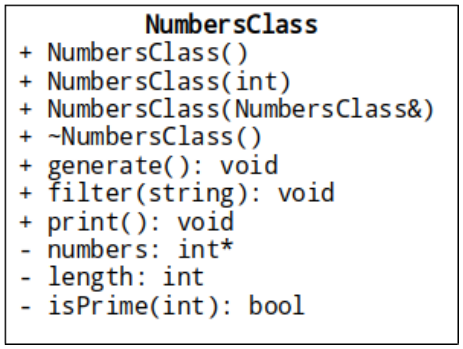
$$numbers_i = \begin{cases} \frac{numbers_{i-1}}{2} + 5, & numbers_{i-1} \text{ is even} \\ 3 * numbers_{i-1} - 1, & numbers_{i-1} \text{ is odd} \end{cases}$$

   Generate your numbers in **generate** function and print your numbers in **print** function. Implement a proper copy constructor for this class, so that the dynamic numbers are managed correctly when an object is copied from an another object.

```
-- [First output, incorrect] --
Enter length and start for Object A: 15 2
Object A (start: 2, length: 15): 2 6 8 9 26 18 14 12 11 32 21 62 36 23 68
Object B (b = a): 2 6 8 9 26 18 14 12 11 32 21 62 36 23 68
Enter new start for Object B: 13
b.start is 13 now
Object A (start: 2, length: 15): 13 38 24 17 50 30 20 15 44 27 80 45 134 72 41
Object B (start: 13, length: 15): 13 38 24 17 50 30 20 15 44 27 80 45 134 72 41

-- [Second output, correct] --
Enter length and start for Object A: 15 2
Object A (start: 2, length: 15): 2 6 8 9 26 18 14 12 11 32 21 62 36 23 68
Object B (b = a): 2 6 8 9 26 18 14 12 11 32 21 62 36 23 68
Enter new start for Object B: 13
b.start is 13 now
Object A (start: 2, length: 15): 2 6 8 9 26 18 14 12 11 32 21 62 36 23 68
Object B (start: 13, length: 15): 13 38 24 17 50 30 20 15 44 27 80 45 134 72 41
```

2. Implement the following NumbersClass class:



Feel free to implement set/get functions as necessary.

This class has a dynamic integer array whose length is determined by either **default constructor**, **overload constructor** or **setLength()** function. This array should be filled by random numbers by **generate()** function. The **filter()** function may accept either "prime" or "nonprime" as its parameter and replace numbers not meeting the parameter type with "-1".

Create an object of this class in main function and generate its numbers. Then Make two copies of this object: filter prime numbers in one of them and filter non-prime numbers in the other. Print all your objects in the end. Make sure that the **copy constructor** works so that you get proper results.

```
-- [First output, incorrect] --
Enter length: 15
A:
B:
C:

-- [Second output, correct] --
Enter length: 15
A: 6899 2724 8024 6792 5583 4916 8346 5696 4284 4314 4483 9955 9067 6809 6054
B: 6899 4483 9067
C: 2724 8024 6792 5583 4916 8346 5696 4284 4314 9955 6809 6054
```

Guess why nothing is printed in the first output.