NAME: _____

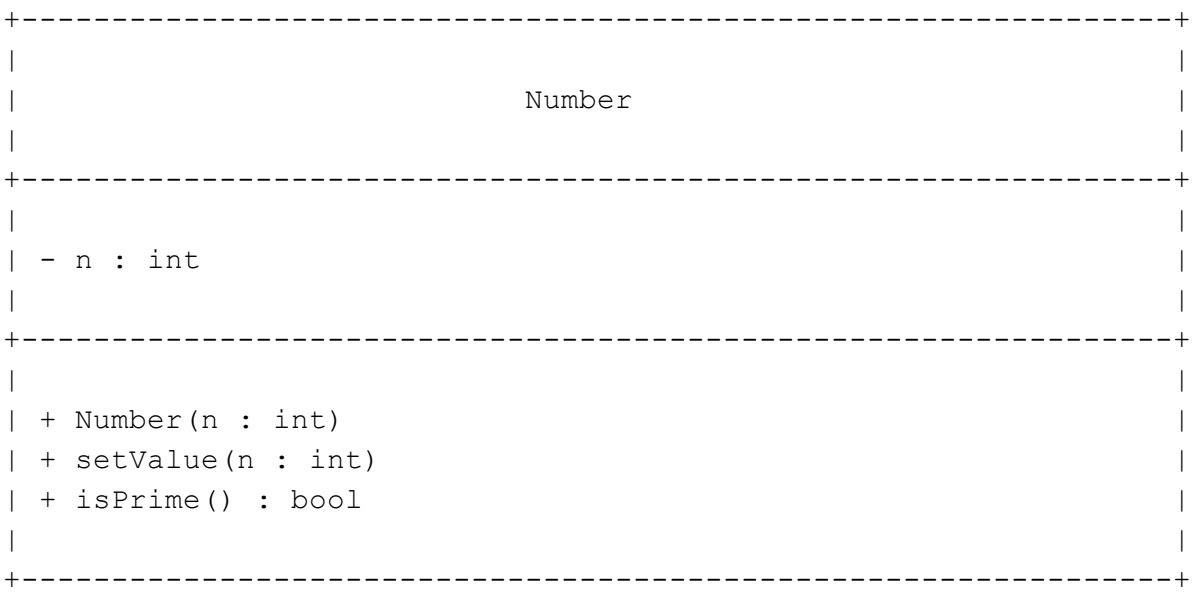

Figure 1

Instances of the *Number* class represent integers. The integer value they represent is passed into the constructor and stored in member variable *n*. The *setValue* function takes an integer argument, which it copies into it's private member variable *n*. The *isPrime* function returns true if member variable *n* is prime and false if not prime.

1) Provide an implementation of the constructor for the Number class. (25 points)

2) Provide an implementation of the *setValue* function in the *Number* class. (25 points)

3) Provide an implementation of the *isPrime* function for the *Number* class. (25 points)

4) Provide test code for the *isPrime* function. Use *assert* statements for this purpose. Make sure that your test code executes every line of code in the function. (25 points)

```
bool areIdentical(const vector<int> & a, const vector<int> & b)
```

Figure 2

5) Write a function that checks whether two vectors are identical (contain exactly the same elements in the same order). A declaration of the function is shown in Figure 2. The function returns true if the two vectors are identical; otherwise it returns false. (25 points)

```
int minValue(vector<int> v)
```

Figure 2

6) Provide an implementation of the minValue function whose declaration is shown in Figure 2 above. The function takes a single argument, which is a vector of *int*. The function returns the smallest *int* that is in the vector. (25 points)

```
int countNegatives(int a[100][100]);
```

Figure 4

7) Implement a function that counts the number of negative integers in a two-dimensional array with 100 rows and 100 columns. The function takes a two-dimensional array of *int* and returns the number of negative numbers in it. A declaration of the function is shown in Figure 4. A declaration of the function is shown in Figure 4. (25 points)

```
int search(const vector<int> & v, int k);
```

Figure 5

8) Implement a function that searches for a given value in a vector of integers. If the value is found, the function returns the index of the value in the vector; otherwise it returns -1. Do not assume the values are in order; do not use binary search. For example, for v = (-2, 4, 18, 6) the function would return 2 for k = 18 and it would return -1 for k = 1. A declaration of the function is shown in Figure 5. (25 points)

```
int binarySearch(const vector<int> & v, int k);
```

Figure 6

9) Implement a function that uses binary search to search for a given value in a vector of integers whose elements are in strictly increasing order. If the value is found, the function returns the index of the value in the vector; otherwise, it returns -1. You can assume that the values passed into the function are in strictly increasing order. For example, for v = (-2, 4, 5, 6) the function returns -1 for k = 2 and 1 for k = 4. A declaration of the function is shown in Figure 6. (25 points EXTRA CREDIT)