1.) The expected outcome of this code is "GtevensQuiz stevensQuiz" in the console. The first object, s1 is initialized with the string "stevensQuiz". The second object, s2, is initialized with the value of s1 using the copy constructor. Then, the "change" method is called on s1 with the arguments 0 and 'G', which changes the first character of the string to 'G'. The "get" method is called on both s1 and s2 and their values are printed to the console.

2a.) False

2b.) True

2c.) False

2d.) False

2e.) True

3.) Lines 15, 17 and 19 are all “cout << endl;” The expression "cout << endl;" is used to create a new line in the console. Std value “cout” is used to output in the console, while "endl" is used to insert a new line in the console. In this program, it essentially just creates a blank space, similar to <br /> in html, allowing for output separation in the concole.

4a.) The expected output of this code is: **x = 10 y = 20**

4b.) Reordering the initialization of x and y should allow for a proper output of the code, although the initial code seemed functional already… Nonetheless, with some minor changes, it can be optimized:

```
#include<iostream>
using namespace std;

class Test {
private:
  int x;
  int y;

public:
  Test() : x(10), y(x + 10) {}
  void print();
};

void Test::print() {
  cout << "x = " << x << " y = " << y << endl;
}
```

```
int main() {
  Test t;
  t.print();
  return 0;
}
```

5a.) True

5b.) At first glance, expected outcome is 10, but due to the const treatment of i, it should experience a compilation error because of the statement **i = 20**, so answer is d?

5c.) a

5d.) Both constructors and destructors can be inherited, so both a and b would be correct.