**Programming Skills: C++** **Homework 2: C++ Basics**

**Due Thu 2/28 (start of class)**

Name: _____

1. Given the following Monster class:

```
1  #include <string>
2
3  class Monster {
4  public:
5      Monster();
6      Monster(const std::string& name, int hp=0);
7      Monster(const Monster& other);
8  private:
9      std::string name_;
10     int hp_;
11 };
```

   For each of the following declarations, indicate whether the default constructor, **D**, string constructor **S**, copy constructor, **C**, or no constructor, **N**, is invoked.

   _____ `Monster m1;`

   _____ `Monster m2();`

   _____ `Monster m3(m1);`

   _____ `Monster& m4(m1);`

   _____ `Monster m5 = m1;`

   _____ `Monster* m6;`

   _____ `Monster m7[4];`

   _____ `Monster *m8 = new Monster("Ralph");`

2. Given the following code:

```
#include <iostream>

using std::cout;
using std::endl;

class Base {
public:
    void one() { cout << "Base::one" << endl; }
    virtual void two() { cout << "Base::two" << endl; }
};

class Derived : public Base {
public:
    void one() { cout << "Derived::one" << endl; }
    void two() { cout << "Derived::two" << endl; }
};

class Derived2 : public Derived {
public:
    void one() { cout << "Derived2::one" << endl; }
    void two() { cout << "Derived2::two" << endl; }
};

int main() {
    cout << "b1" << endl;
    Base *b1 = new Derived();
    b1->one(); b1->two();

    cout << "d1" << endl;
    Derived *d1 = new Derived();
    d1->one(); d1->two();

    cout << "d2" << endl;
    Derived2 *d2 = new Derived2();
    d2->one(); d2->two();

    cout << "d3" << endl;
    Derived *d3 = new Derived2();
    d3->one(); d3->two();

    cout << "b2" << endl;
    Base *b2 = new Derived2();
    b2->one(); b2->two();
}
```

Show the output of the program.

3. Given the following code:

```
1  #include <string>
2
3  class Base {
4  public:
5      Base(const std::string& name) :
6              name_(name) {}
7  private:
8      std::string name_;
9  };
10
11 class Derived : public Base {
12 public:
13     Derived(const std::string& name, int num_things) :
14             Base(name),
15             things_(new int[num_things]) {}
16     ~Derived() { delete [] things_; }
17 private:
18     int* things_;
19 };
20
21 int main() {
22     Base b1("b1");
23     Derived d1("d1", 4);
24     Base* bPtr = new Derived("thing", 10);
25     delete bPtr;
26 }
```

Valgrind reports 40 bytes are lost when the program exits. Explain what is wrong and how to fix it by only changing the `Base` class implementation.

4. Given the following code:

```
1  #include <iostream>
2  #include <vector>
3
4  class Performer {
5  public: virtual void perform() {std::cout << "Performer" << std::endl;}
6  };
7  class Musician : public Performer {
8  public: void perform() {std::cout << "Musician" << std::endl;}
9  };
10 class Bassist : public Musician {
11 public: void perform() {std::cout << "Bassist" << std::endl;}
12 };
13 int main() {
14     std::vector<Performer> performers;
15     performers.push_back(Performer());
16     performers.push_back(Musician());
17     performers.push_back(Bassist());
18     for (unsigned int i=0; i<performers.size(); i++) {
19         performers[i].perform();
20     }
21     return 0;
22 }
```

(a) What is output of the program?

(b) What is the technical term for what is happening to the objects that are being added to the collection?

(c) Briefly explain how you would fix this problem.

5. Given the following class:

```
1  class Calculator {
2  public:
3      Calculator() : num_mults_(0), memory_(0) {}
4      int multiply(int val1, int val2) {
5          ++num_mults_;
6          return val1 * val2;
7      }
8      int multiplies() const { return num_mults_; }
9      int recall() const { return memory_; }
10     void store(int val) { memory_ = val; }
11 private:
12     int num_mults_;
13     int memory_;
14 };
```

(a) Rewrite the class so it is templated on a user defined type to be multiplied.

(b) Indicate what restrictions the template class imposes on the parameterized type. In other words, if the `Calculator` class was instantiated with a user defined class, `Currency`, what methods must be present in `Currency`? You must provide the full list of methods, even if the compiler generates some automatically. In addition to listing the restrictions, indicate which segments of the code require which methods (using the line numbers from the original code).

6. Given the following abstract class:

```
1  #include <string>
2
3  class Weapon {
4  public:
5      Weapon(const std::string& name, int quality);
6      std::string name() const;
7      int quality() const;
8      virtual std::string attack() = 0;
9  protected:
10     int quality_;
11     std::string name_;
12 };
```

Write a complete concrete subclass, `Lightsaber`. It has this additional functionality:

(a) It can be constructed with a name (string), quality (int) and a color (string).

(b) It has a public accessor for the color.

(c) It can be copy constructed or assigned to with the same name as the other object, but the quality should be set to 0.

(d) When attacking the quality of the weapon should be reduced by 1. It should then return the string `"The saber glows bright color!"`, where `color` is the color of this light saber.

7. The Jedi council wants to keep an inventory of all commissioned lightsabers in a `set`. The lightsabers should be ordered first by decreasing quality, and second by color (alphabetically). Give a piece of code that creates the set, `lightsabers`, along with code that allows the lightsabres to be ordered how the council wants.