CS 32 Week 5 Worksheet

This worksheet is entirely **optional**, and meant for extra practice. Some problems will be more challenging than others and are designed to have you apply your knowledge beyond the examples presented in lecture, discussion or projects. All exams will be done on paper, so it is in your best interest to practice these problems by hand and not rely on a compiler.

If you have any questions or concerns, please go to any of the office hours.

Concepts
Inheritance & Polymorphism

1.) What does the current code output, and what changes do you have to make to the following program to have it output "`I'm Gene`"?

HINT: You will need to use the virtual keyword!

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

class LivingThing {
   public:
         void intro() { cout << "I'm a living thing" << endl; }
};

class Person : public LivingThing {
   public:
         void intro() { cout << "I'm a person" << endl; }
};

class UniversityAdministrator : public Person {
   public:
         void intro() {
cout << "I'm a university administrator" << endl;
}
};

class Chancellor : public UniversityAdministrator {
   public:
         void intro() { cout << "I'm Gene" << endl; }
};
```

```
int main() {
   LivingThing* thing = new Chancellor();
   thing->intro();
}
```

Difficulty: Easy

2.) Given the following class declarations, complete the implementation of each constructor so that the program compiles. Your implementations should correctly assign constructor arguments to class member variables.

HINT: You will need to use initializer lists!

```
class Animal {
      public:
      Animal(string name);
      private:
      string m_name;
      };

      class Cat : public Animal {
      public:
      Cat(string name, int amountOfYarn);
      private:
            int m_amountOfYarn;
      };

      class Himalayan : public Cat {
      public:
            Himalayan(string name, int amountOfYarn);
      };

      class Siamese: public Cat {
public:
            Siamese(string name, int amountOfYarn, string
toyName);
      private:
            string m_toyName;
      };
```

Difficulty: Easy

3.) What is the output of the following code?

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

class Pet {
public:
      Pet() { cout << "Pet" << endl; }
      ~Pet() { cout << "~Pet" << endl; }
};

  // This is an unusual class that derives from Pet but also
  // contains a Pet as a data member.
class Dog : public Pet {
public:
      Dog() { cout << "Woof" << endl; }
      ~Dog() { cout << "Dog ran away!" << endl; }
private:
      Pet buddy;
};

int main() {
      Pet* milo = new Dog;
      delete milo;
      }
```

Difficulty: Medium

4.) Suppose the class declaration for Pet was changed as shown below. What is the output of the code in problem 3) with these new changes?

```
class Pet {
public:
      Pet() { cout << "Pet" << endl; }
      virtual ~Pet() { cout << "~Pet" << endl; }
};
```

Difficulty: Medium

5.) The following code has several errors. Rewrite the code so that it would successfully compile. Try to catch the errors without the use of a compiler.

```
class LivingThing {
private:
      int age;
};

class Person : public LivingThing {
public:
      Person(int a) { age = a; }
      void birthday() {
            age++;
      }
};
```

Difficulty: Easy

6.) Would the following work in C++? Why or why not?

```
class B;
      class A : public B { … code for A … };
      class B : public A { … code for B … };
```

Difficulty: Medium

7.) Examine the following code and determine its output.

```
#include <iostream>
#include <string>

using namespace std;

class A {
public:
A() : m_val(0) {
      cout << "What a wonderful world! " << m_val << endl;
}

virtual ~A() { cout << "Guess this is goodbye " << endl; }
virtual void saySomething() = 0;
```

```
virtual int giveMeSomething() = 0;

private:
      int m_val;
};

class B : public A {
public:
      B() : m_str("me"), m_val(1) {
            cout << m_str << " has just been birthed."  << endl;
      }
      B(string str, int val) : m_str(str), m_val(val) {
            cout << "More complex birth " << m_str << endl;
      }
      ~B() {
            cout << "Why do I have to leave this world!" << endl;
      }
      virtual void saySomething() {
            cout << "Coming in from " << m_str << " with "
                 << giveMeSomething() << endl;
      }
      virtual int giveMeSomething() { return m_val*5; }
      private:
      int m_val;
      string m_str;
};

class C {
public:
      C() : m_val(2) {
            m_b = new B("C", m_val);
            cout << "Hello World!!" << endl;
      }
      C(const B& b, int val) :  m_val(val) {
            m_b = new B(b);
            cout << m_b->giveMeSomething() << endl;
      }
      ~C() {
            m_b->saySomething();
            delete m_b;
            cout << "Goodbye world!" << endl;
      }
private:
      B* m_b;
      int m_val;
```

```
};

int main() {
      B* b_arr = new B[3];
      for(int i = 0; i < 3; i++) {
            b_arr[i].saySomething();
      }
      B b("B", 5);
      A* a = &b;
      cout << a->giveMeSomething() << endl;
      C c;
      C c2(b, b.giveMeSomething());
      delete [] b_arr;
}
```

Difficulty: Hard