Assignment 3 – Polymorphism

Goals-

Identify requirements for a program using polymorphism
Create a program to demonstrate polymorphism

You will create a simple class hierarchy as the basis for a fantasy combat game. Your 'universe' contains Goblins, Barbarians, Reptile People, Blue Men and possibly others. Each will have characteristics for attack, defense, armor, and strength points.

Type	Attack	Defense	Armor	Strength Points
Medusa[1]	2d6 *Glare	1d6	3	8
Gollum[2]	1d6 *Ring	1d6	3	8
Reptile People[3]	3d6	1d6	7	18
Blue Men[4]	2d10	3d6*	3	12 *Mob
Harry Potter[5]	2d6	2d6	0	10/20*Hogwarts

3d6 is rolling three 6-sided dice. 2d10 is rolling two 10-sided dice.

*Glare- If a Medusa rolls a 12 in attack then the target has looked her in the eyes and is turned to stone. The Medusa wins! NOTE: This ability does **NOT** require a separate function! It can be done by modifying this creature's attack function.

*Ring- Gollum as the one ring. It can make him invisible. That is essentially his defense! It also allows him to occasionally get behind his opponent and jump on their back. There is a 5% chance he makes an attack with 3d6. Note- Implement this AFTER you have tested the base attack function and save a copy of that file. Just in case. ☺

*Mob- The Blue Men are actually a swarm of small individuals. For every 4 points of damage (round down) they lose one defense die. For example, when they reach strength of 8 they only have 2d6 for defense.

*Hogwarts- If Harry dies (i.e. strength >=0) he immediately recovers. His strength becomes 10 again, for a max of 20 for the fight. If he were to die again then he's dead.

NOTE- Implement all creatures without their special abilities. AFTER you have tested the normal defense function then add the specials. But first save a copy of that file. Just in case. ☺

To resolve an attack you will need to generate 2 dice rolls. The attacker rolls the appropriate number and type of dice under Attack. That amount of damage is passed to the defender in the defender object's defense function. The defender rolls the appropriate number and type of dice under Defense. You subtract the Defense roll from the Attack roll. That is the damage. To apply the damage you subtract the Armor value. The result is then subtracted from the Strength Points. That value becomes the new Strength Points for the next round. If Strength Points goes to 0 or less then the character is out of the combat.

HINT: Carefully think through how the attack and defense functions will interact. Something like this:

```
hits = Creature1.attack();
Creature2.defense(hits);
```

You need to create a creature, or character class. Call it what you choose. Then you will have a subclass for each of these characters. The parent class will be an **abstract class**. Each subclass will vary only in the values in the table. Since each starts with the same data elements you will only need one constructor. It is part of your design task to determine what functions you will need. Maybe attack() and defense()? Incorporate them into your class hierarchy.

To manage the combat you will need two pointers to Creature, say fighter1 and fighter2. You will instantiate each to the appropriate derived class object. All that your main function should do is to call the attack and defend functions. All other actions should be enclosed in each derived class.

You must complete your design document. It must include a class hierarchy. In your reflections you can discuss how the original design may have changed as you worked through the problem. You must also submit a test plan. The test plan should cover all logic paths. So you should have each character type have combat with all character types (including another of its own). Remember to submit these documents as PDF files.

To test your classes create a program that instantiates 2 creatures of the type specified by the user; such as Medusa vs Reptile People, or Blue Men vs The Shadow. Conduct rounds of combat until only one has strength points remaining. Can you have a draw? That is can both run out of strength points in the same round? Look at your design for the answer. ☺

It is not hard, just a lot to think about. The TAs will be asked to grade your project against your design so please do not just throw together some random stuff so you have a file to submit. You are not required to implement only the original design. BUT, your reflections will need to explain the difference. If you give us a random design you will need to explain each step in how you got to the code submitted. In other words, that will make it much more difficult. So, learn a good habit and think about it before you start coding. ☺

HINT: This program has a random element. You will need to address that in your test plan. It will also affect debugging. Your design should address this (potential) problem. It is not hard but you need to think about it.

What you need to submit in your zip archive:

Your program file(s) with the implementation of these five creatures inheriting from a single parent.
    You should have a header file and source file for each class
    You should have a program file containing your code to test and demonstrate the use of the other classes. This will be a program that just runs a combat between 2 creatures.
Your makefile
Your design document (basically the class hierarchy)
Your test plan
Your reflections document

1. Scrawny lady with snakes for hair. They help with fighting. Just don't look at her!

2. He is small, not fast, and not strong.

3. Big, powerful, with tough skin. But very slow.

4. They are small (6" tall), fast and tough. So they are hard to hit and can take some damage. As for the attack value, you can do a LOT of damage when can crawl inside the armor or clothing of your opponent. ☺ And yes they are the Nac Mac Feegle of Discworld. I just wanted a shorter name for your project.

5. Why are you reading this? How can you not know who Harry Potter is? ☺

NOTE: The sample creatures are unbalanced intentionally. This will help you in debugging your program! If Gollum is winning all the time something is probably wrong.

======

Grading:

- programming style and documentation (10%)
    1. create the base abstract class (10%)
    2. create the base class and the Reptile People class (10%)

3. create a test driver program to create character objects which make attack and defense rolls required to show your classes work correctly (10%)

- create the Medusa class- redefine attack function (10%)
- create the Gollum class- redefine attack function (10%)
- create the Blue Men class- redefine defense function (10%)
- create the Harry Potter class- redefine defense function (10%)
- reflections document to include the design description, test plan, test results, and comments about how you resolved problems during the assignment (20%)

Suggestion: The grading is set up to encourage you to develop your program incrementally! Start with the base and Reptile People classes. Create the testing program that runs sample combats for you toe test your code. They are numbered to emphasize this point. How do you handle random die rolls when testing your code? Then do The Shadow and Blue Men. Then do Medusa and Hydra, without their special abilities. Then add the special attack and defense features.

Hint: Create your design before coding anything! You should even be outlining your test plan. At each step of the process make notes about what worked, what changed. Doing this as you progress will make writing the reflections easier.