COMP 202 Winter 2005 Assignment #4

On objects, aggregation, encapsulation, etc.

Distributed: March 4th, Due: Mar 18th

Playing with monsters

One of the best known applications of computers are video games. They are also some of the most sophisticated and interesting pieces of software, including all sorts of components dealing with many different and complex issues.

In fact, there is a research group in the School of Computer Science here at McGill solely devoted to studying computer games (see http://gram.cs.mcgill.ca).

There are many different genres such as RTS (Real-Time Strategy,) FPS (First-Person Shooters,) and RPG (Role Playing Games,) amongst others.

Many games deal with several kinds of objects. Some objects are innanimated (such as a box with resources) while others are animated (such as units which are part of an "army" in a war game.) Animated units can be computer controlled or human controlled.

In this problem you are asked to develop a very simple video game (without the graphics for simplicity.)

Monsters

The game consists of several "monsters" which have special features, and which can fight each other.

A monster can attack other monsters (but only if it is alive,) can receive damage when attacked by other monsters, and can heal.

Each monster has a number of "hit-points." The hit-points represent the “health” of a monster. Each time the monster receives some damage when attacked by another monster, its hit-points are reduced. When the hit-points of a monster reach zero, the monster dies. Each monster starts with a maximum number of hit-points.

Each monster also has an “attack-strength” representing the amount of damage that it can produce on another monster.

When a monster receives damage, its hit-points get decremented by a certain given amount.

When a monster attacks another monster, it produces an amount of damage on the target monster. This amount is the attack-strength plus or minus a random number between zero and ten percent of the attacker’s attack-strength points.

When a monster heals, its hit-points are incremented by 10% of the monster's maximum hit-points.

There are two classes of monsters: "simple" monsters and "coupled" monsters.

A simple monster has hit-points, attack-strength and the capability of attacking other monsters, of receiving damage, and of healing, as described above.

A coupled monster is the combination of two simple monsters. It has its own hit-points in addition to its component monsters. Initially its maximum hit-points are 90% of the sum of the hit-points of each monster in the pair. For example, if A is a simple monster with 80 hit-points and B is a simple monster with 110 hit-points, then a coupled monster (A,B) would have 0.9*(80+110)=171 maximum hit-points. Similarly it's attack-strength is 90% of the combined attack-strength of the monsters in the pair. It also has the capabilities of attacking, receiving damage, and healing.

When a coupled monster receives damage, its own hit-points are decremented by the amount of damage received, and each component monster will receive half of that damage.

When a coupled monster attacks, it attacks as a whole, so the amount of damage in the victim is the overall attack-strength (90% of the sum of each attack strength in the pair) plus or minus a random number between zero and fifteen percent of the coupled monster's attack-strength.

When a coupled monster heals, its hit-points are incremented by 5% and each simple monster in the pair heals, as described above for simple monsters.

A coupled monster dies when one of its component monsters dies, or if its total hit-points reach 0.

Any monster that is already dead cannot be damaged further, cannot attack, and cannot heal.

The game

The game will have several monsters: one (simple monster) controlled by the user, and two by the computer (one simple and one coupled.) The game works as follows: each monster has a turn and can take two possible actions: attack another monster, or rest in order to heal.

If the monster is controlled by the user, the user decides which action to take. If the action is to attack, the user is asked which of the other two monsters to attack. If the action is to rest, it will simply heal.

If the monster is controlled by the computer, it decides randomly whether to attack or rest. If it attacks it decides randomly which of the other two monsters to attack. If the action is to rest, it will heal.

The game is won by the last monster standing.

Representing simple monsters

The essential issues in this problem are how to represent monsters as objects, define how monsters should react to messages from other monsters, and how they interact overall.

Define a class `SimpleMonster` whose objects have as attributes the characteristics described above (name, hit-points, maximum hit-points, and attack strenght,) and can react to the "messages" sent to it (i.e. it can perform its operations): damage, attack, heal. In addition, these objects should be able to tell any object who asks, whether it is alive or not, how many hit-points it has left, what are the maximum hit-points, what is its attack strength, type or name. In other words, it should have the following methods:

- `SimpleMonster(String name, int maxHitPoints, int attackStrength)`

  This is the constructor, which initialises the state of the monster. The hit-points must be initialised to the maximum hit-points.

- `String getName()`

  which returns the monster's name.

- `int getHitPoints()`

  which returns the monster's current hit-points.

- `int getMaxHitPoints()`

  which returns the monster's maximum hit-points.

- `int getAttackStrength()`

  which returns the monster's attack strength.

- `boolean isAlive()`

  which returns true if the monster is still alive, and false otherwise.

- `void damage(int amount)`

  which reduces the hit-points by the given amount. When the hit-points reach 0 or below, the monster dies.

- `void attack(SimpleMonster victim)`

  which damages the victim monster, by an amount which is equal to the attacker's attack strength plus or minus a random number between zero and ten percent of the attacker's attack strength.

- `void attack(CoupledMonster victim)`

  which damages the victim monster, by an amount which is equal to the attacker's attack strength plus or minus a random number between zero and ten percent of the attacker's attack strength. It is the same as the previous, but the victim is a coupled monster.

- `void heal()`

  which increments the monster's hit-points by 10% of the maximum hit-points, but never goes above the maximum hit-points.

- `public String toString()`

  which returns a string representation of the monster with the format

  `(name='`*name*`', type=simple, status=dead` *or* `alive, max_hp=`*maxhit-points*`, hp=`*hit-points*`, as=`*attack-strength*`)`

Representing coupled monsters

A coupled monster is a pair of two simple monsters, refered to as "top" and "bottom". It has a name, hit-points, maximum hit-points and attack strength, as described above. Its name is the concatenation of its two component monsters's names. It can get damaged and can attack and heal, but the rules are a bit different than for simple monsters. See the description below. It must be possible to obtain the normal monsters that make up a coupled monster. To represent coupled monsters we need a class `CoupledMonster` which should have the following methods:

- `CoupledMonster(SimpleMonster top, SimpleMonster bottom)`

  which is the constructor, receiving two monsters top and bottom respectively, and initialising all necessary variables.

- `String getName()`

  which returns the monster's name.

- `int getHitPoints()`

  which returns the monster's current hit-points (initially 90% of the combined hit-points of the top and bottom monsters).

- `int getMaxHitPoints()`

  which returns the monster's maximum hit-points (90% of the combined maximum hit-points of the top and bottom monsters).

- `int getAttackStrength()`

  which returns the monster's attack strength (90% of the combined attack strength of the top and bottom monsters).

- `SimpleMonster getTopMonster()`

  which returns a reference to the top monster.

- `SimpleMonster getBottomMonster()`

  which returns a reference to the bottom monster.

- `boolean isAlive()`

  which returns true if the monster is still alive, and false otherwise. A coupled monster is alive if both top and bottom monsters are alive, and its hit-points are greater than 0.

- `void damage(int amount)`

  which reduces the coupled monster's overall hit-points, by the given amount, and damage each of its individual monsters (top and bottom) by half the amount. (Being coupled has its benefits!)

- `void attack(SimpleMonster victim)`

  which damages the victim monster, by an amount which is equal to the attacker's attack strength plus or minus a random number between zero and fifteen percent of the attacker's attack strength.

- `void attack(CoupledMonster victim)`

  which damages the victim monster, by an amount which is equal to the attacker's attack strength plus or minus a random number between zero and fifteen percent of the attacker's attack strength. It is the same as the previous, but the victim is a coupled monster.

- `void heal()`

  which increments the monster's hit-points by 5% of the maximum hit-points, but never goes above the maximum hit-points. It should also heal the top and bottom monsters of this coupled monster.

- `public String toString()`

  which returns a string representation of the monster with the format

  `(name='`*name*`', type=coupled, status=dead` *or* `alive, max_hp=`*maxhit-points*`, hp=`*hit-points*`, as=`*attack_strength*`,`

  `top='`*name of top monster*`', top_hp=`*top monster's hit-points*`,`

  `bot='`*name of bottom monster*`', bot_hp=`*bottom monster's hit-points*`)`

The main loop

First the user is asked to enter the names of its monster and the names of the other three simple monsters involved. Then the maximum hit-points of each simple monster are determined at random, between 80 and 120. Similarly for the attack strengths, but each attack strength should be between 10 and 30. Then the simple monsters are created, and two of the monsters meant to be computer-controlled are used to create a single coupled monster. So to summarize, there is one simple monster controlled by the user, one simple monster controlled by the computer and one coupled monster also controlled by the computer.

Then the fight begins. Each of the three monsters (two simple, one coupled) take turns (you can decide who starts, and the order may be the same every round.) If it's the user's turn, and its monster is not dead, it is asked whether to attack or rest. If it decides to attack, the user is asked who should be the target of the attack (one of the other two.) If it is the turn of any computer-controlled monster and the monster is not dead, the action of attacking or resting is decided at random, as well as the target of the attack (if it is an attack move.)

After each player makes a move, the action is printed (e.g. “Monster A attacked monster B” or “Monster A rested”) and the state of all the monsters is printed. (Hint: use the toString methods.)

The game finnishes when only one monster is alive, and the name of the winner is printed.

Sample run

```
Monsters battle!
Please enter the name of your monster: Julius
Please enter the name of monster #1: Asterix
Please enter the name of monster #2: Obelix
Please enter the name of monster #3: Idefix

The state of all monsters is now:
(name='Julius', type=simple, status=alive, max_hp=91, hp=91, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=113, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

It's your turn. What action will you take?
1: Attack or 2: Rest? 1
Who would you like to attack: 1: Asterix or 2: ObelixIdefix? 1

The state of all monsters is now:
(name='Julius', type=simple, status=alive, max_hp=91, hp=91, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=94, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

Asterix rested

The state of all monsters is now:
(name='Julius', type=simple, status=alive, max_hp=91, hp=91, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=105, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

ObelixIdefix rested

The state of all monsters is now:
(name='Julius', type=simple, status=alive, max_hp=91, hp=91, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=105, as=24)
```

```
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

It’s your turn. What action will you take?
1: Attack or 2: Rest? 2

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=91, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=105, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

Asterix rested

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=91, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=113, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

ObelixIdefix attacked Julius

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=62, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=113, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

It’s your turn. What action will you take?
1: Attack or 2: Rest? 1
Who would you like to attack: 1: Asterix or 2: ObelixIdefix? 2

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=62, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=113, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=152, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=91)

Asterix rested

The state of all monsters is now:
```

```
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=62, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=113, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=152, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=91)

ObelixIdefix attacked Asterix

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=62, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=78, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=152, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=91)

It’s your turn. What action will you take?
1: Attack or 2: Rest? 1
Who would you like to attack: 1: Asterix or 2: ObelixIdefix? 2

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=62, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=78, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=133, as=32,
  top=’Obelix’, top_hp=73,
  bot=’Idefix’, bot_hp=82)

Asterix attacked Julius

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=38, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=78, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=133, as=32,
  top=’Obelix’, top_hp=73,
  bot=’Idefix’, bot_hp=82)

ObelixIdefix attacked Asterix

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=38, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=49, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=133, as=32,
  top=’Obelix’, top_hp=73,
  bot=’Idefix’, bot_hp=82)

It’s your turn. What action will you take?
1: Attack or 2: Rest? 2
```

```
The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=47, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=49, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=133, as=32,
  top=’Obelix’, top_hp=73,
  bot=’Idefix’, bot_hp=82)

Asterix attacked Julius
The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=23, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=49, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=133, as=32,
  top=’Obelix’, top_hp=73,
  bot=’Idefix’, bot_hp=82)

ObelixIdefix rested

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=23, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=49, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=141, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=92)

It’s your turn. What action will you take?
1: Attack or 2: Rest? 1
Who would you like to attack: 1: Asterix or 2: ObelixIdefix? 1

The state of all monsters is now:
(name=’Julius’, type=simple, status=alive, max_hp=91, hp=23, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=30, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=141, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=92)

Asterix attacked Julius

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=30, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=141, as=32,
  top=’Obelix’, top_hp=82,
  bot=’Idefix’, bot_hp=92)

ObelixIdefix rested
```

```
The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=30, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=149, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=30, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=149, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

Asterix rested

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=41, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=149, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

ObelixIdefix rested

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=41, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=41, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

Asterix attacked Julius

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=41, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

ObelixIdefix attacked Julius

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=41, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=41, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

Asterix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=157, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

ObelixIdefix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=165, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

The state of all monsters is now:

(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=165, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

Asterix attacked Julius
```

```
The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=165, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

ObelixIdefix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=171, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

Asterix attacked ObelixIdefix

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=147, as=32,
  top='Obelix', top_hp=79,
  bot='Idefix', bot_hp=88)

ObelixIdefix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=155, as=32,
  top='Obelix', top_hp=88,
  bot='Idefix', bot_hp=98)

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=155, as=32,
  top='Obelix', top_hp=88,
  bot='Idefix', bot_hp=98)

Asterix attacked Julius

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=155, as=32,
  top='Obelix', top_hp=88,
  bot='Idefix', bot_hp=98)

ObelixIdefix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=163, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=52, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=163, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

Asterix rested

The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=63, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=163, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

ObelixIdefix attacked Asterix
The state of all monsters is now:
(name='Julius', type=simple, status=dead, max_hp=91, hp=0, as=19)
(name='Asterix', type=simple, status=alive, max_hp=113, hp=29, as=24)
(name='ObelixIdefix', type=coupled, status=alive, max_hp=171, hp=163, as=32,
  top='Obelix', top_hp=91,
  bot='Idefix', bot_hp=100)

The state of all monsters is now:
```

```
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=29, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=163, as=32,
  top=’Obelix’, top_hp=91,
  bot=’Idefix’, bot_hp=100)

Asterix attacked ObelixIdefix

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=alive, max_hp=113, hp=29, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=139, as=32,
  top=’Obelix’, top_hp=79,
  bot=’Idefix’, bot_hp=88)

ObelixIdefix attacked Asterix

The state of all monsters is now:
(name=’Julius’, type=simple, status=dead, max_hp=91, hp=0, as=19)
(name=’Asterix’, type=simple, status=dead, max_hp=113, hp=-3, as=24)
(name=’ObelixIdefix’, type=coupled, status=alive, max_hp=171, hp=139, as=32,
  top=’Obelix’, top_hp=79,
  bot=’Idefix’, bot_hp=88)

ObelixIdefix wins!
```

Special considerations

- Your program must make appropriate use of encapsulation.
- In the main class (the driver class) you should use static methods in order to make your code more modular. It is bad design putting all code in one method alone. For example, finding out if the game is over, or executing the turn of a computer-controlled monster, may be better placed in their own methods. This will help make your code easier to develop.

What to submit

- You must submit the following files:
  - `SimpleMonster.java`
  - `CoupledMonster.java`
  - `MonstersGame.java`
- The names of the methods must be exactly as asked.

Marking scheme

- Syntax: 1 point
- Style (documentation, variable names, etc.): 1 point
- Program structure: 6 points
- Correct algorithms: 6 points
- Correct execution: 6 points