Test 3

The second test will be 75 minutes and will consist of two parts.

The programming part will be about Chapter 2-5, excluding Section 2.6, 4.5, 5.2 and 5.3. You will be asked to implement one class. We will already provide you with a skeleton which includes the javadoc. This part will be worth 50% of the marks. If your code does not compile, you get a 50% penalty (that is, your score for the programming part will be divided by 2 if your code does not compile).

The “written” part will be about Chapter 2-5, excluding Section 2.6, 4.5, 5.2 and 5.3. This part will consist of six questions (two multiple choice, two short answer questions and two longer answer questions). This part will be worth the remaining 50% of the marks.

During the test, you will have access to the textbook. You may bring a blank piece of paper to the test.

Chapter 5: Aggregation and Composition

EECS 1030

moodle.yorku.ca

Aggregation

Definition

Aggregation is a binary relation on classes. The pair (A, P) of classes is in the aggregation relation if class A (aggregate) has a non-static attribute of type P (part).

Aggregation

Definition

Aggregation is a binary relation on classes. The pair (A, P) of classes is in the aggregation relation if class A (aggregate) has a non-static attribute of type P (part).

The aggregation relation is also known as the *has-a* relation. Instead of saying that (A, P) is in the aggregation relation, we often simply say that A has-a P.

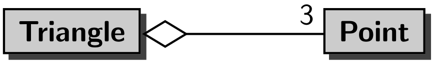
Composition

Composition is a special type of aggregation. The aggregate A and its part P form a composition if "A owns P", that is, each object of type A has exclusive access to its attribute of type P.

The designer and the implementer of a class determine whether an aggregation is a composition.

Java does not provide any special language constructs for implementing compositions. The constructors, accessors and mutators are implemented in a particular way.

Aggregation or composition?



Aggregation or composition?

The API of the Triangle class can be found here.

Question

Assume that you only have the bytecode of the Triangle class, not its Java code. How do you determine whether Triangle and Point form an aggregation or a composition?

Copy

Problem

```
Point first = new Point(0, 0);
Point second = new Point(1, 2);
Point third = new Point(2, 0);
Triangle triangle = new Triangle(first, second, third);
```

Draw the diagram representing memory at the end of the execution of the above main method.

Copy

Problem

```
Point first = new Point(0, 0);
Point second = new Point(1, 2);
Point third = new Point(2, 0);
Triangle triangle = new Triangle(first, second, third);
Triangle alias = triangle.getAlias();
```

Draw the diagram representing memory at the end of the execution of the above main method.

Copy

Problem

Add to the Triangle class the method

```
public Triangle getAlias()
```

Copy

Problem

```
Point first = new Point(0, 0);
Point second = new Point(1, 2);
Point third = new Point(2, 0);
Triangle triangle = new Triangle(first, second, third);
Triangle shallowCopy = triangle.getShallowCopy();
```

Draw the diagram representing memory at the end of the execution of the above main method.

Copy

Problem

Add to the Triangle class the method

```
public Triangle getShallowCopy()
```

Copy

Problem

```
Point first = new Point(0, 0);
Point second = new Point(1, 2);
Point third = new Point(2, 0);
Triangle triangle = new Triangle(first, second, third);
Triangle deepCopy = triangle.getDeepCopy();
```

Draw the diagram representing memory at the end of the execution of the above main method.

Copy

Problem

Add to the Triangle class the method

```
public Triangle getDeepCopy()
```

Refactor

Problem

Implement this API using the Calendar class instead of the Date class.

Chapter 6: Inheritance

EECS 1030

moodle.yorku.ca

Aggregation

Combine (simple) data into more complex data.

Inheritance

Add some (simple) data to already existing complex data.

For the resulting data, add new operations (mainly to handle the added simple data) and possibly redefine some of the operations of the complex data.

Inheritance

Definition

Inheritance is a binary relation on classes. The pair (C, P) of classes is in the inheritance relation if the API of the class C (child) contains

```
class C extends P
```

The API of the class P (parent) may (but does not have to) contain

```
Direct Known Subclasses: C
```

The inheritance relation is also known as the is-a relation. Instead of saying that (C, P) is in the inheritance relation, we often simply say that C is-a P.

GoldenRectangle

The API of the GoldenRectangle class can be found here.

Inheritance

Question

Which is correct?

(a) `GoldenRectangle` is-a `Rectangle`

(b) `Rectangle` is-a `GoldenRectangle`

Inheritance

Question

Which is correct?

(a) `GoldenRectangle` is-a `Rectangle`

(b) `Rectangle` is-a `GoldenRectangle`

Answer

(a)

Inheritance

Question

Which is correct?

(a) `GoldenRectangle` is-a `Rectangle`

(b) `Rectangle` is-a `GoldenRectangle`

Answer

(a)

Question

Which is correct?

(a) `Object` is-a `Rectangle`

(b) `Rectangle` is-an `Object`

Inheritance

Question

Which is correct?

(a) `GoldenRectangle` is-a `Rectangle`

(b) `Rectangle` is-a `GoldenRectangle`

Answer

(a)

Question

Which is correct?

(a) `Object` is-a `Rectangle`

(b) `Rectangle` is-an `Object`

Answer

(b)

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Question

Which is correct?

(a) `Rectangle` is a superclass of `GoldenRectangle`
(b) `GoldenRectangle` is a superclass of `Rectangle`

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Question

Which is correct?

(a) `Rectangle` is a superclass of `GoldenRectangle`

(b) `GoldenRectangle` is a superclass of `Rectangle`

Answer

(a)

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Question

Which is correct?

(a) `Rectangle` is a subclass of `GoldenRectangle`

(b) `GoldenRectangle` is a subclass of `Rectangle`

Inheritance

Definition

P is a superclass of C if C is-a P.

C is a subclass of P if C is-a P.

Question

Which is correct?

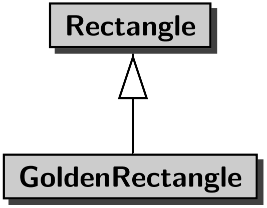
(a) `Rectangle` is a subclass of `GoldenRectangle`

(b) `GoldenRectangle` is a subclass of `Rectangle`

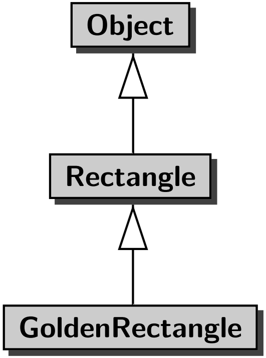
Answer

(b)

UML Diagrams



UML Diagrams



State of an object

Question

What is the state of an object?

State of an object

Question

What is the state of an object?

Answer

The non-static attributes and their values.

State of an object

```
final int WIDTH = 3;
final int HEIGTH = 4;
final int WEIGHT = 80;
GoldenRectangle rectangle =
  new GoldenRectangle(WIDTH, HEIGHT, WEIGHT);
```

State of an object

```
final int WIDTH = 3;
final int HEIGTH = 4;
final int WEIGHT = 80;
GoldenRectangle rectangle =
  new GoldenRectangle(WIDTH, HEIGHT, WEIGHT);
```

Question

What is the state of `rectangle`?

State of an object

```
final int WIDTH = 3;
final int HEIGTH = 4;
final int WEIGHT = 80;
GoldenRectangle rectangle =
  new GoldenRectangle(WIDTH, HEIGHT, WEIGHT);
```

Question

What is the state of `rectangle`?

Answer

`width` $\mapsto$ 3
`height` $\mapsto$ 4
`weight` $\mapsto$ 80

State of an object

Question

Are private non-static attributes inherited by a subclass?

State of an object

Question

Are private non-static attributes inherited by a subclass?

Answer

No! But they are part of the state of an instance of the subclass.

State of an object

```
final int WIDTH = 3;
final int HEIGTH = 4;
final int WEIGHT = 80;
GoldenRectangle rectangle =
  new GoldenRectangle(WIDTH, HEIGHT, WEIGHT);
```

Problem

Draw the memory diagram.