Final Part 1 of 6

Started: Aug 19 at 3:26pm

Quiz Instructions

Welcome to the CSC207 Summer 2021 final assessment. This final assessment comes in 6 parts. Each one is a different quiz on Quercus. You have until 14:00 EST on Friday 20 August 2021 to submit your answers to all of them.

SAVE YOUR ANSWERS IN A TEXT FILE ON YOUR COMPUTER before copying them into Quercus and clicking submit.

You can use spell-check, grammar-check, IntelliJ, the internet, and your notes from the course to answer these questions. IF YOU COPY SOMEONE ELSE'S WORDS it is considered to be CHEATING!!!! **Be sure to put everything in your own words.** Give enough detail so that we are convinced that you understand the concepts.

To ask a question during the exam, go to our usual lecture Zoom session (**link (https://utoronto.zoom.us/j/89521849618)**) during the following times:

- 14:00--16:00 ET on Thursday 19 Aug
- 22:00--23:59 ET on Thursday 19 Aug
- 9:00--11:00 ET on Friday 20 Aug
- 13:00--14:00 ET on Friday 20 Aug

The first question in Final Part 1 of 6 contains the statement of academic integrity - please make sure to read it.

Question 1 0 pts

Welcome to the CSC207 Summer 2021 final assessment. This final assessment comes in 6 parts. Each one is a different quiz on Quercus. You have until 14:00 ET on Friday 20 August 2021 to submit your answers to all of them.

SAVE YOUR ANSWERS IN A TEXT FILE ON YOUR COMPUTER before copying them into Quercus and clicking submit.

You can use spell-check, grammar-check, IntelliJ, the internet, and your notes from the course to answer these questions. IF YOU COPY SOMEONE ELSE'S WORDS it is considered to be CHEATING!!!! **Be sure to put everything in your own words.** Give enough detail so that we are convinced that you understand the concepts.

To ask a question during the exam, go to our usual lecture Zoom session (link (https://utoronto.zoom.us/j/89521849618)) during the following times:

- 14:00--16:00 ET on Thursday 19 Aug
- 22:00--23:59 ET on Thursday 19 Aug
- 9:00--11:00 ET on Friday 20 Aug
- 13:00--14:00 ET on Friday 20 Aug

Statement 1: I understand that submitting and taking credit for someone else's words and/or code is an academic offence and I will not do that during this final assessment.

- (●) True
- ( ) False

Question 2 1 pts

Consider the following two classes:

Vehicle.java

```
public class Vehicle {

    int numWheels = 1;

    public void ride(Vehicle v) {
        System.out.println(v.getNumWheels());
    }

    public int getNumWheels() {
        return numWheels;
    }

    public static void main(String[] args) {
        Vehicle v1 = new Bicycle();
        Bicycle v2 = new Bicycle();
        v1.ride(v2);

    }
}
```

Bicycle.java

```
public class Bicycle extends Vehicle {

    int numWheels = 2;

    @Override
```

```
    public int getNumWheels() {
        return numWheels;
    }

    void ride(String s) {
        System.out.println(s);
    }

    public void ride(Vehicle v) {
        System.out.println(this.getNumWheels());
    }
}
```

When we run the `main` method, the output to the screen is:

- ○ v1
- ◉ 2
- ○ v2
- ○ 1

Question 3

2 pts

The following classes are part of a larger program that keeps track of the inventory in a commercial kitchen. Explain why the following code prevents the rest of the program from compiling:

```
public class FoodItem {
    private String name;
    private int quantity;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getQuantity() {
        return quantity;
```

```
    }

    public void addToQuantity(int amount) {
        quantity += amount;
    }
}
```

```
public class Apple extends FoodItem{
    private int numSeeds;

    public Apple(int ns){
        super("Apple", 1);
        numSeeds=ns;
    }

    public int getNumSeeds(){
        return numSeeds;
    }

    public void setNumSeeds(int ns){
        numSeeds=ns;
    }
}
```

Since the parent class FoddItem does not have the constructor, when the subclass Apple inherits the method, the super method is not able to find the constructor in the parent class.

p

31 words

Question 4

2 pts

Look up the List interface on the Oracle website here:
(https://docs.oracle.com/javase/8/docs/api/java/lang/Class.html)

https://docs.oracle.com/javase/8/docs/api/java/util/List.html
(https://docs.oracle.com/javase/8/docs/api/java/util/List.html)

Use the information on this website to explain what the following code does. Your answer should contain at least three sentences or more.

```
import java.util.List;
import java.util.Stack;

public class Main {
    public static void main(String[] args) {
        List<String> ex = new Stack();
        ex.add("Hello");
        System.out.println(((Stack<String>) ex).pop
());
    }
}
```

First of all, the first two lines import the List and Stack packages. Then it declares a List<String> type variable named ex and initializes it with a Stack instance since the Stack implements the interface List<E>. In the next step, it uses add method from the Stack class to add "Hello" into ex because of the look down rule. Finally, since there is no pop method in the List interface, the look down rule is not applied here. We need to cast ex to the Stack<String> in order to use the pop method, and it pops the "Hello", which is printed in the end.

p 104 words </>

Question 5 2 pts

Consider a program that allows the user to scan barcodes or manually enter barcodes in order to enter new products into the inventory of a store. This program contains class `Book` that extends `StockItem` and class `Cake` that extends `FoodItem`.

`Book` and `Cake` both contain methods `autoReorder` and `checkThreshold`, which allows the system to automatically re-order both items if their quantity goes below a certain threshold value. The threshold value is different for each object.

There are other `StockItem` objects that can be automatically re-ordered, but not all of them. There are other `FoodItem` objects that can be automatically re-ordered, but not all of them.

We want to change the program so that we can loop through all objects that can be automatically re-ordered, using a for-all loop. To accomplish this in a way that is most extensible, we should make it so that:

- ○ `Book` extends `Cake`.
- ◉ Both `Book` and `Cake` implement an interface that contains the `autoReorder` and `checkThreshold` methods.
- ○ `Cake` extends `Book`.
- ○ Both `Book` and `Cake` directly extend the same abstract class.

Question 6 0 pts

Explain your answer to the previous question.

Since not all the StockItem objects and FoodItem objects could be automatically re-ordered, we cannot inherit all the items to one parent class. The interface can provide specific methods and use as the type in the for loop.

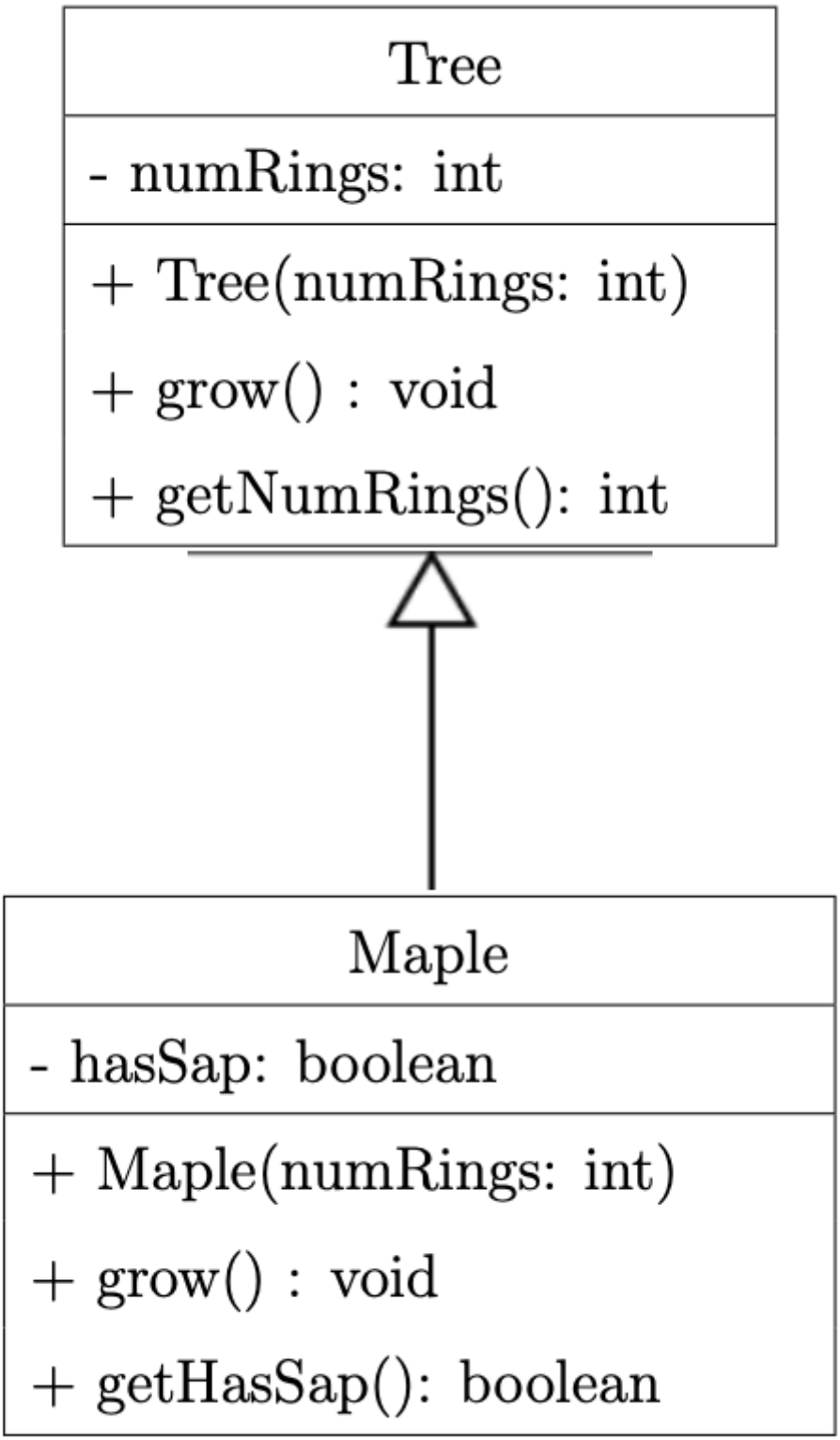
p ▸ span 38 words </>

Question 7

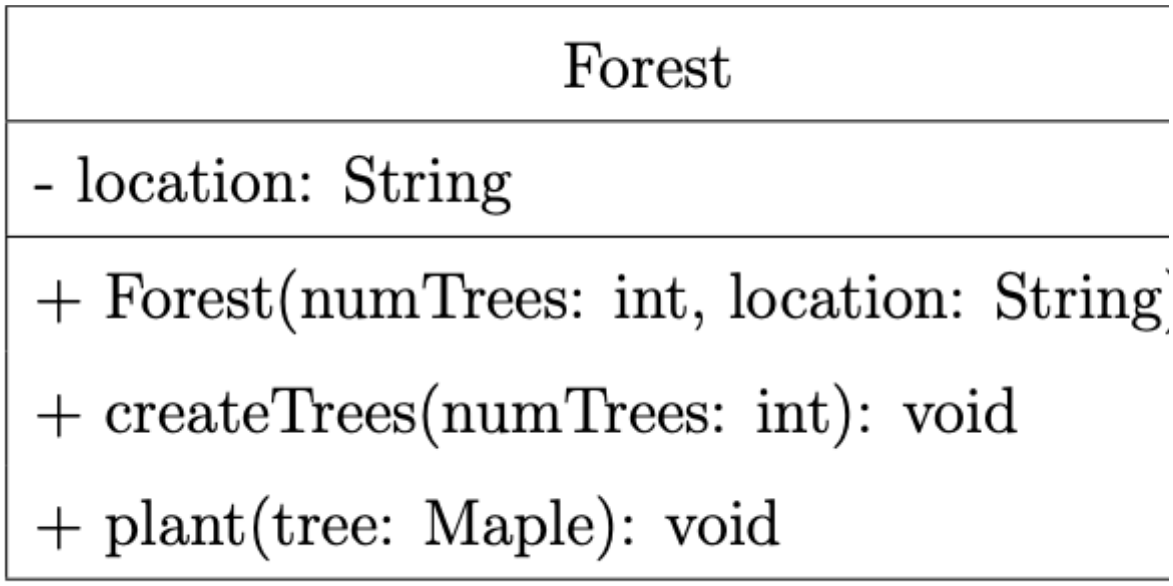
1 pts

Consider the following UML diagrams for the `Tree`, `Maple`, and `Forest` classes.

UML for `Tree` and `Maple`:



UML for `Forest`:



Which of the following lines of code is the first to prevent the entire `main` method from compiling?

```
public static void main (String[] args) {

 1. Tree t = new Tree(20);
 2. Tree x = new Maple(30);
 3. Maple m = new Maple(40);
 4. int z = m.getNumRings();
 5. x.grow();
 6. m.grow();
 7. Forest forest = new Forest(10, "High Park");
 8. forest.plant(x);
 9. forest.createTrees(3);
10. boolean b = t.getHasSap();

}
```

- ○ line 5
- ○ line 6
- ○ line 9
- ◉ line 8
- ○ ones of the constructor calls on lines 1, 2, 3, or 7
- ○ line 10
- ○ line 4

Question 8

1 pts

Consider the following code:

```
int a = 2;
int b = 2;
Integer w = new Integer(2);
Integer x = new Integer(2);
Integer y = x;
Integer z = new Integer(w);
```

Select all of the true statements:

- [ ] `a.equals(x)`

  returns true because they store the same value.

- [x] `a == x`

  returns true because Java automatically unboxes the value that is stored in `x`.

- [x] `w.equals(x)`

  returns true because they store the same value.

- [ ] `w == z`

  returns true because `w` and `z` are aliases for the same object.

- [x] `x == y`

  return true because `x` and `y` are aliases for the same object.

- [x] `a == b`

  is true because `a` and `b` are both primitive variables.

Question 9

1 pts

Consider the following code:

```
public class Person {...}
```

```
public class Student extends Person {...}
```

In a Controller class:

```
public Person graduate(Student s) {
  return s;
}
```

The graduate method does which of the following? Select the most accurate answer.

- ○ The `graduate` method does not change the way the program interacts with `s`.
- ◉ The `graduate` method casts `s` as an instance of `Person`.
- ○ The `graduate` method creates a new `Person` object and makes `s` point at it.

Quiz saved at 4:25pm Submit Quiz