Full Name: _____

Student ID#: _____

CSE 1325 OBJECT-ORIENTED PROGRAMMING

PRACTICE #4 Exam #1 «---» 9 1 001 «---» Exam #1 PRACTICE #4

Instructions

1. Students are allowed pencils, erasers, UTA Student ID, and beverage only. **A UTA Student ID is required to turn in the exam.**
2. All books, bags, backpacks, phones, **smart watches**, and other electronics, etc. must be placed along the walls. **Silence all notifications.**
3. PRINT your name and student ID at the top of this page **and every additional pastel coding sheet**, and verify that you have all 10 pages.
4. **Read every question completely before you start to answer it.** If you have a question, please raise your hand. You may or may not get an answer, but it won't hurt to ask.
5. If you leave the room, you may not return.
6. You are required to SIGN and ABIDE BY the following Honor Pledge for each exam this semester.

Honor Pledge

On my honor, I pledge that I will not attempt to communicate with another student, view another student's work, or view any unauthorized notes or electronic devices during this exam. I understand that the professor and the CSE 1325 Course Curriculum Committee have zero tolerance for cheating of any kind, and that any violation of this pledge or the University honor code will result in an automatic grade of zero for the semester and referral to the Office of Student Conduct for scholastic dishonesty.

Student Signature: _____

Vocabulary

Write the word or phrase from the Words list below to the left of the definition that it best matches. Each word or phrase is used at most once, but some will not be used. {10 at 2 points each}

Vocabulary

Word	Definition
1	The class from which members are inherited
2	A named scope
3	The task of keeping a system consisting of many versions well organized
4	A special class member that creates and initializes an object from the class
5	An object created to represent an error or other unusual occurrence and then propagated via special mechanisms until caught by special handling code
6	A procedure for solving a specific problem, expressed in terms of an ordered set of actions to execute
7	A style of programming focused on the use of classes and class hierarchies
8	A block of memory associated with a symbolic name that contains a primitive data value or the address of an object instance
9	Bundling data and code into a restricted container
10	Code for which specified assertions are guaranteed to be true (often, a class in which fields cannot change after instantiation)

Word List

Abstract Class	Abstract Method	Abstraction	Algorithm	Assertion
Class	Constructor	Data Validation	Declaration	Definition
Destructor	Encapsulation	Enumerated type	Exception	Field
Garbage Collector	Inheritance	Interface	Invariant	Method
Multiple Inheritance	Namespace	Object	Object-Oriented Programming	Operator
Override	Package	Primitive type	Reference Counter	Subclass
Superclass	UML	Validation Rules	Variable	Version Control

Multiple Choice

Read the full question and every possible answer. Choose the one best answer for each question and write the corresponding letter in the blank next to the number. The image below clarifies the meaning of open and closed arrows and diamonds. Note that "e.g." means "for example". {15 at 2 points each}

1. ____ **A package-private class member**

    A. Is visible anywhere in the program

    B. Is visible to any package member, but not outside the package

    C. Is visible within the class, but not outside the class

    D. Is visible to other package-private class members, but not to public class members

2. ____ **By default, if an exception is not caught in a Java program,**

    A. The exception will be reported to standard error (STDERR) and execution will continue

    B. The exception will be noted in the program log and execution will continue

    C. The program will abort

    D. The exception will be ignored

3. ____ **A UML class is represented by a rectangle with 3 panes. The top pane contains**

    A. The fields

    B. The name of the class and an optional stereotype

    C. The executable members such as constructors and methods

    D. The top pane is always empty

4. ____ **Class Tree is a final class. Which Java statement will create an instance of class Tree?**

    A. `Tree t();`

    B. `Tree t = new Tree();`

    C. `Tree t;`

    D. A final class cannot be instanced

5. ____ **Given our standard build.xml file, we can delete all .class files using which bash command?**

    A. `java clean`

    B. `make clean`

    C. `clean`

    D. `ant clean`

6. ____ **Which of the following is TRUE about JavaDoc?**

    A. JavaDoc does not permit custom tags to be defined

    B. A JavaDoc comment begins with /**

    C. JavaDoc includes public, package-private, and protected class members by default

    D. JavaDoc can only produce PDF documents, not websites

7. ____ **What is TRUE of Java's toString() method?**

    A. We can override toString() for both primitive and custom types

    B. The default toString() concatenates all public fields

    C. It is used to convert an object to a String

    D. All of these are true

8. ____ **From the statement** `grades.add(Grade.A);`**, we can infer that** `grades` **is defined as**

    A. `Grade[] grades;`

    B. `String grades;`

    C. `ArrayList<Grade> grades;`

    D. `Grade grades;`

9. ____ **To compare the values of** `Fruit f1` **and** `Fruit f2` **in Java, write**

    A. `*f1 == *f2`

    B. `equals(f1, f2);`

    C. `f1.equals(f2)`

    D. `f1 == f2`

10. ____ **Which class member can be overridden by a subclass?**

    A. An abstract method

    B. A final method

    C. A static method

    D. A constructor

11. ____ **Refer to the image at the start of this section. Composition in UML is represented by a line with**

    A. A closed arrow

    B. A closed diamond

    C. An open arrow

    D. An open diamond

12. ____ **Error messages should be written to which Java stream?**

    A. `System.in`

    B. `System.err`

    C. `System.out`

    D. `System.oops`

13. ____ **Which statement is TRUE about String in Java?**

    A. Each char in a String is 8 bits, just like in a C char*

    B. The length of String s is determined using `strlen(s)`

    C. A new String may be initialized using `String s = "cse1325";`

    D. A specific char within a String may be accessed with subscripts, e.g., `s[3]`

14. ____ **Which can NOT be done with an abstract class?**

    A. Use it as a superclass (that is, extend it)

    B. Use it as a parameter type

    C. Instance it

    D. Use it as a return type

15. ____ **In JavaDoc comments, the name of the class author is tagged with**

    A. `.. author ::`

    B. `@author`

    C. `#author`

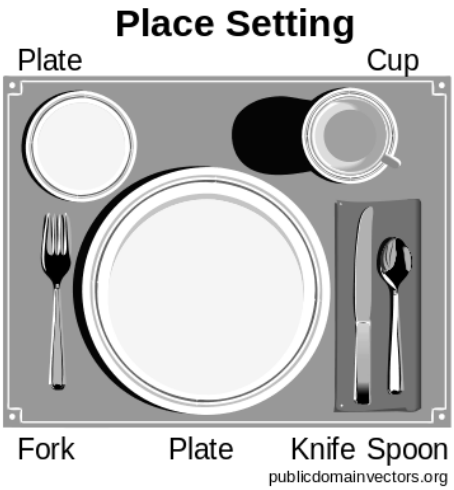
    D. `<author>`

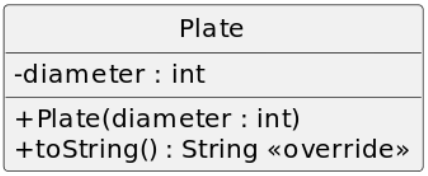
Free Response

Write the solutions to the Free Response questions in the provided space. Additional space is on back, and additional coding sheets are available on request. **Write your name and student ID on EVERY additional coding sheet you use.**

Multi-part questions are based on the class diagrams shown, however, *each question is independent* of the others. Don't ignore the remaining questions if you have trouble answering one, just skip to the next question and continue.

1. {Code a Class} A place setting as shown below consists of 2 plates, a cup, and 3 utensils of cutlery - a knife, a fork, and a spoon. Consider the class diagram. Class `Plate` represents a plate with the diameter (the size of the plate) given in inches.





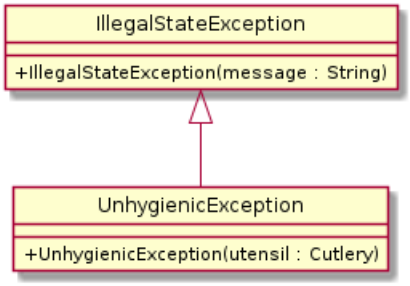
   a. {5 points} In file Plate.java, **begin writing Java class Plate**. For this question, declare the Plate class so that it will be visible to any code in the system, and then define the field. DO NOT write the constructor and method members of class Plate here - they may be written for separate questions below. Assume any required imports - you need not write them.

b. {5 points} In file Plate.java, assuming all fields are properly declared, **write just the Plate constructor**, properly initializing the field to the parameter value. | | | |

   c. {5 points} In file Plate.java, assuming all fields and constructors are properly declared, **override the toString method** to convert the object to a String. Ensure that the superclass has a matching toString method to override. For example, an 8" diameter plate would print `8" plate`. (DO NOT return this as a *literal* String, but use the actual field value instead.)

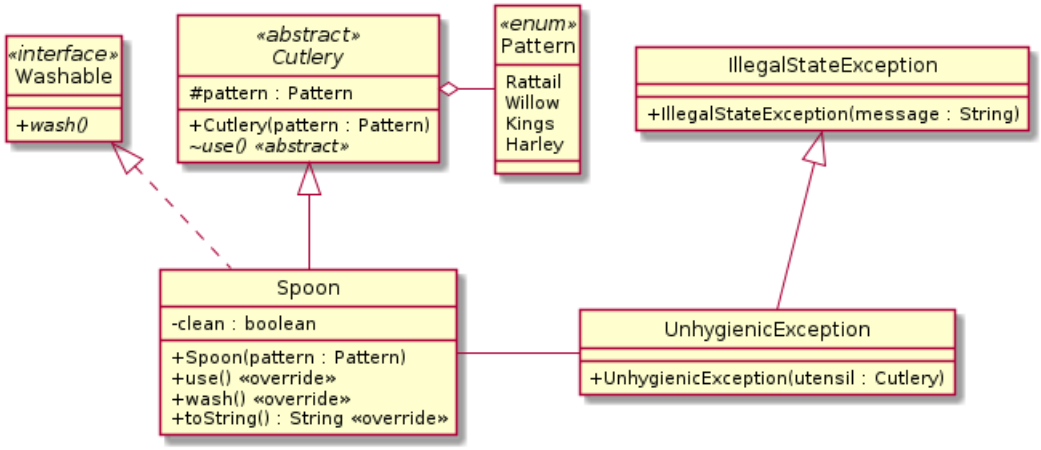
2. {3 points} {Code an enum} Cutlery comes in many ornate forms called *patterns*. **Write the enumeration** in Java of some popular cutlery patterns as shown in the class diagram so it is visible *only* within the current package.

3. {6 points} {Code a custom exception} Unwashed plates, cups, and cutlery are *unhygienic*, or dirty enough to perhaps make you sick. To report the use of unwashed items in a place setting, **create a custom Java exception** `UnhygienicException` as shown in the class diagram. Assume all imports - you do not need to write them. Write the constructor for `UnhygienicException` to chain to its superclass constructor with the String representation of parameter `utensil` followed by "isn't clean!" as its parameter. **Important:** `IllegalStateException` is part of the Java Class Library - DO NOT write it! Write only `UnhygienicException`.



4. {2 points} {Code an interface} Most dirty items can be washed to make them clean again. In file Washable.java, **write interface Washable** as shown in the class diagram.

5. {Inheritance} Here is a more complete class diagram for cutlery in our place setting (Knife and Fork are omitted).



a. {4 points} Write the name of **any one class member** (NOT enum) in the above diagram that is

- private:
- protected:
- package-private:
- public:

b. {6 points} In file Cutlery.java, **write abstract class Cutlery**. The constructor should initialize the field to its parameter. Assume any required imports - you need not write them.

c. {3 points} In file Spoon.java, **begin writing Java class Spoon**. For this question, declare the Spoon class so that it will be visible to any code in the system and includes the relationships shown, and then define the field. DO NOT write the constructor and method members of class Spoon here - they may be written for separate questions below. Assume any required imports - you need not write them.

d. {5 points} In class Spoon, the `clean` field is true if the Spoon object has never been used or has since been washed. In file Spoon.java, **continue writing class Spoon**, writing *only* method `use()`, ensuring that it overrides a superclass or interface method declaration. In method `use()`, if the Spoon is not `clean`, throw an `UnhygienicException` with the current object as the parameter. Otherwise, print "Stirring with" and the object's own String representation to System.out, then set field `clean` to false.

6. {4 points} {Write a main method} In file Setting.java, assuming all classes and enums above are properly defined in the same package, write static method `main`. In `main`:

- Create an object of type `Spoon` that uses a Harley pattern.
- `use` the spoon.
- If an `UnhygienicException` is thrown when the spoon is used, `wash` it and then `use` it again.

BONUS #1: {+3 points} Write a valid bash command that would do the following:

- List the files in the current directory: _____
- Create a new directory named bonus: _____
- Edit file Bonus.java (any editor you choose): _____
- Add and then commit file Bonus.java to the git repository (using any commit message you choose):

  _____ ; _____
- Replace file Bonus.java with the most recently committed version: _____

BONUS #2: {+2 points} In ONE brief sentence, explain why you should always select a License for public GitHub repositories containing code that you intend for others to use.

Additional Free Response

Include the question number with your answer!