THE UNIVERSITY OF BRITISH COLUMBIA
CPSC 310: SAMPLE FINAL EXAM

Name: ___________________________________ **Student #:** _________________

Signature: _________________________________

Notes about this examination

1. You have **150 minutes** to write this examination.
2. There are **120 marks** available. You may wish to start with the questions that you consider to be the easiest.
3. No notes, books, or any type of electronic equipment is allowed, including cell phones and calculators.
4. Good luck!

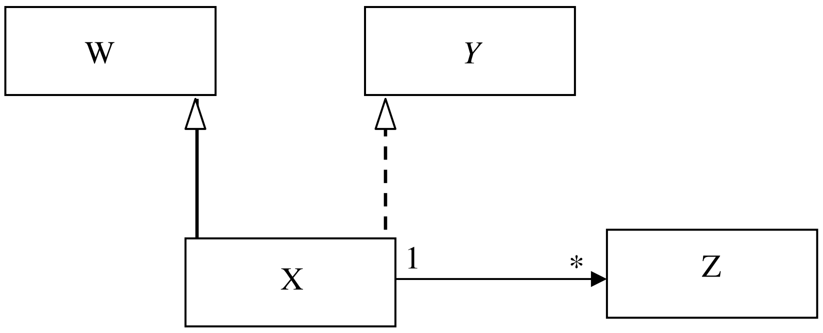
	Marks	**Max**
Multiple Choice		**20**
True/False		**20**
21		**6**
22		**8**
23		**8**
24		**12**
25		**6**
26		**6**
27		**8**
28		**8**
29		**6**
30		**6**
31		**6**
Total		**120**

1. **[2 marks]** Select the best functional requirement from the list of requirements below.
   a) A warning dialog should pop up if the student's assignment does not compile.
   b) The system needs to manage the student grades.
   c) The marker should be able to use this tool to aid the assignment evaluation process.
   d) The marker must be able to edit comments in the marking report.

2. **[2 marks]** You should use this design pattern when extension by subclassing is impractical and you need to add responsibilities to individual objects dynamically.

   a) Composite
   b) Mediator
   c) Decorator
   d) Singleton
   e) None of the above

3. **[2 marks]** Which names make the most sense for the UML diagram shown below?



   a. W: Person, X: Employee, Y: ComparableInterface, Z: Responsibility
   b. W: Cat, X: Dog, Y: Animal, Z: Leg
   c. W: Inventory, X: RetailInventory, Y: AbstractInventory, Z: Item
   d. W: Person, X: Parent, Y: Employee, Z: Child

[2 marks] Which design principle is violated in the following code snippet?

```
public class FurnitureTable {

import java.awt.Color;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import com.sample.model.Content;
import com.sample.model.FurnitureEvent;
import com.sample model.FurnitureListener;
import com.sample.model.ContentManager;
import com.sample.model.Home;
import com.sample.model.HomePieceOfFurniture;
import com.sample.model.SelectionEvent;
import com.sample.model.SelectionListener;
import com.sample.model.UserPreferences;
import com.sample.model.HomePieceOfFurniture.SortableProperty;
import com.sample.model.FurnitureCatalog;
...
```

a. Open/Closed Principle
b. Law of Demeter
c. Liskov Substitution Principle
d. High Cohesion
e. Low Coupling
f. None of the above

4. **[2 marks]** Conflict between members of a software development team:

   a. Can lead to a better product
   b. Can be handled poorly and cause some team members to feel uncomfortable
   c. Can arise if one or more team members have different ideas about the best way to solve a problem
   d. a and b
   e. All of the above

5. **[2 marks]** You have to test a *factorial* function, but you don't have access to the code. Which of the following is the best set of test inputs?

   a) 1, 2, 5, 200
   b) -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5
   c) -4, 1, 5, 250
   d) All the positive integers until the program crashes

[2 marks] What type of coverage would you get if you tested the following code with the values `x = 3, y = 4` and `x = 2, y = 1`.

```
public void calc(int x, int y) {
      if (x < y)
      {
            if (x > 2)
            {
                  y++;
            }
      }
      else
      {
            y--;
      }
}
```

a. Statement coverage but not branch or path coverage
b. Statement and branch coverage but not path coverage
c. Statement, branch and path coverage
d. None of the above

6. **[2 marks]** When you are part of a test-driven development team, in which order should complete the following tasks?

   a. Write the code, Make the code compile, Write the tests, Make the tests pass
   b. Write the tests, Write the code, Make the code compile, Make the tests pass
   c. Write the code, Write the tests, Make the code compile, Make the tests pass
   d. None of the above

7. **[2 marks]** You are working on a poster to advertise around campus for your new tech start-up. You'd like to include a picture of a puppy on your poster to soften your audience. You go to Google Images, do a search for "puppy" and find the perfect image on an artist's website. There is no copyright notice anywhere on the image or on the website. Are you free to include this image in your presentation?

   a. Yes - since you are using only one of their images, and it won't be the main focus of your poster, this qualifies as fair dealing.
   b. Yes - since there is no copyright notice anywhere on the image or website, the image is not protected by copyright and you may use it freely.
   c. Yes - copyright only applies to printed materials, not to electronic media that you find online, so you are free to use the image.
   d. No - this is copyright infringement and is illegal.
   e. No - you must wait until 20 years after the image was published before you may use it freely.

[2 marks] What's the worst code smell in the following code snippet?

```
public class CapitalStrategy {

   // other code not shown ...

  public double capital(Loan loan){
       if (loan.isExpired() && !loan.isMature())
       {
              return loan.getCommitment() * loan.duration() * loan.riskFactor();
       }
       if (!loan.isExpired() && loan.isMature())
       {
              if (loan.getUnusedPercentage() != 1.0)
                     return loan.getCommitment() * loan.getUnusedPercentage() *
                     loan.duration() * loan.riskFactor();
              else
                     return (loan.amount() * loan.duration() * loan.riskFactor())
                     + (loan.unusedAmount() * loan.duration() * loan.unusedFactor());
       }
       return 0.0;
  }
}
```

a. Data Class
b. Duplicated Code
c. Decompose Conditional
d. Feature Envy
e. Long Method

The space below is intentionally blank.

True/False

For each true/false question below provide a *one sentence* justification of your answer.

8. **[2 marks]** The EPL (Eclipse Public License) is more business-friendly than the Gnu GPL (General Public License).

9. **[2 marks]** You cannot use XP if you cannot create automated tests for your software.

10. **[2 marks]** A use case diagram is a complete graphical representation of a use case.

11. **[2 marks]** Testing is used to demonstrate that software is free of errors.

12. **[2 marks]** It is expected that a team of experienced software developers using an agile development process will produce smelly code.

13. **[2 marks]** Design patterns encapsulate existing, well-proven re-usable blocks of design and therefore should be applied without modification.

14. **[2 marks]** When using a version control system (eg. Jazz), a new version of the software (aka, a baseline) should only be created if you are planning on publically releasing the version.

15. **[2 marks]** Reusing previously tested and deployed software components always leads to higher software quality.

16. **[2 marks]** Clear and specific requirements can be very useful when planning acceptance testing.

17. **[2 marks]** Professional software engineers do not need to worry about intellectual property issues unless they own their own company.

Short Answer

18. **[6 marks]** You decided to start a new software company with some friends. Unfortunately, most of your friends who are starting the company with you are inexperienced programmers and don't have great programming style. They tend to leave out comments, format code in a hard to understand way, and generally deliver code with poor style. What could you do to improve this situation and why would it work?

19. **[8 marks]** What are some advantages (at least 4) of using Jazz compared to using a text editor and a command line compiler for software development? For each advantage, explain why it is an advantage.

a. **[4 marks]** Draw a high level use case diagram showing the main functions of a self-service vending machine that dispenses snacks and drinks.

b. **[2 marks]** Write a "brief use case" for one of the use cases shown in your diagram.

c. **[2 marks]** Write two functional requirements that would be necessary based on your use case from part b.

21. **[12 marks]** For each of the following design patterns, provide a two-three sentence description, an example of when it could be applied (a different example than the ones in the notes), and at least one benefit of using it.

    a. Abstract Factory

    b. Composite

c. Mediator

The space below is intentionally blank.

22. **[6 marks]** For each of the following design principles
    - define it in 2 sentences or less
    - explain how following it helps you create better designs

    a. information hiding

    b. open/closed principle

    c. Law of Demeter

23. **[6 marks]** Given your preference, would you choose to work in a company with democratic teams or hierarchical teams? Be sure to identify the tradeoffs you considered in your answer.

The space below is intentionally blank.

24. **[8 marks]** Define each of the following, and identify a key way in which each plays a role in the development of quality software.

pair programming

code reviews

regression tests

system tests

25. **[8 marks]** Agile methods are receiving a lot of attention in terms of modern approaches to software engineering. Describe at least 4 of the practices that are considered to be agile, and discuss their tradeoffs with the more traditional waterfall-based methods.

26. **[6 marks]** Compare and contrast the traditional waterfall model with the staged-delivery model.

The space below is intentionally blank.

27. **[6 marks]** A software developer on your team refuses to refactor. She claims that since the code works it's a waste of her time to refactor because she could be working on new features instead. How would you convince her that she should refactor code when necessary?

The space below is intentionally blank.

28. Consider the following java code.

```
private void fireDotPressed(KeyEvent e) { // 1 check if last word is an
       // array
       String invoker = null; // 2 check if last word is
       // an object
       String className = null; // 3 check if last word is a
       // class

       invoker = getInvoker(); // retrieve the previous word
       if (invoker != null)
       {
              if (invoker.equals("this")) {
                     className = getFileName();
                     try {
                            System.out.println("verifying if this class has a package");
                            Class.forName(className, false,
                                          new URLLoader(EJE.getEJE().getClassPath()));
                            System.out.println("verifying if class has no package");
                            classWizard = new ClassWizard(className,ClassWizard.OBJ);
                            showClassWizard(e);
                            return;
                     } catch (ClassNotFoundException ex) {
                            System.out.println("verifying if class has a package as import");
                            className = verifyImports(className);
                            classWizard = new ClassWizard(className, ClassWizard.OBJ);
                            showClassWizard(e);
                            return;
                     }
              }
              if (invoker.equals("super")) {
                     className = getSuperClassName();
                     classWizard = new ClassWizard(className, ClassWizard.OBJ);
                     showClassWizard(e);
                     return;
              }
              // check if invoker is a class
              try {
                     Class.forName(invoker, false, new java.net.URLClassLoader(EJE
                                   .getEJE().getClassPathForReflection()));
                     // IS A CLASS IN DIRECTORY OR A FULLY QUALIFIED NAME
                     classWizard = new ClassWizard(invoker, ClassWizard.CLASS);
                     showClassWizard(e);
                     return;
              } catch (ClassNotFoundException ex) {
                     System.out.println("Class name: " + invoker + " not found, verifying imports");
                     // 90 LINES OMITTED FROM THIS CATCH HANDLER
                     // SO THAT THIS EXAMPLE WOULD FIT ON ONE PAGE!
              }
       }
}
```

a. **[1 mark]** What smells exist in the above code?

b. **[1 mark]** Which smell is the worst?

c. **[2 marks]** What is the most appropriate refactoring to use to start getting rid of the most important code smell? How would you apply the refactoring? (either write the code or **clearly and concisely** explain what changes you would make)

d. **[2 marks]** Why do you think the refactoring in part c is the most appropriate?