# THE UNIVERSITY OF AUCKLAND

**FIRST SEMESTER, 2015**

**Computer Science**

# TEST

**Software Construction**

**(Time Allowed: 60 minutes)**

Note:

- This is a closed-book test. The use of calculators is NOT permitted.
- Before opening this booklet:
  - **Compare** the test version number on the Teleform sheet supplied with the version number above. If they do not match, ask the supervisor for a new sheet.
  - **Enter** your name and student ID on the Teleform sheet. Your name should be entered left aligned. If your name is longer than the number of boxes provided, truncate it.
  - **Write** your name, student ID, and UPI at the bottom of this page.
  - **Read** the notes on this page.
- When the signal is given to "open the test booklet", you should
  - **Read** through the entire test.
  - **Plan** your time, so that you will be able to review your Teleform answers for accuracy and completeness *before* the signal is given to "stop writing".
- Answer Sections A-E on the **Teleform answer sheet** provided.
  - **Use** a dark pencil to mark your answers clearly.
  - **Erase** carefully and completely, if you change your mind. (Only one answer per question!)
  - **Check** that the question number on the sheet corresponds to the question number in this question/answer book.
  - If you spoil your sheet, **ask** the supervisor for a replacement.
  - There are a total of 90 marks in these defined-response sections: 14 multiple-choice questions worth 3 marks each, and 24 true-false questions worth 2 marks each.
- Answer Section F in the space provided in this booklet.
  - **Write** your ID number at the top of each answer page.
  - **Write** your answers in the space provided in the short answer section.
  - The space provided is intended to be **more** than sufficient to answer each question: you can gain full marks even if you don't fill every text box.
- There are overflow pages at the end of this booklet.
  - You may use an overflow page to **explain** any reasonable assumption you made, when answering one of the questions in Sections A-E. Note that your marker will *not* look at anything you write on Sections A-E.
  - You may use an overflow page to **revise or extend** any answer you made in Section F.
  - Take care to **number your answers** if you write on an overflow page.

| | |
|---|---|
| **Surname:** | |
| **First Name(s):** | |
| **Student ID:** | |
| **Login Name(UPI):** | |

# Section A

1. **[3 marks]** Backwards compatibility of Java source code means that

   a. A Java program which was compiled in an old version of Java is likely to run without errors on a recent Java Runtime Environment (JRE).
   b. A Java program which was developed in an old version of Java should be executed on a version of the Java Runtime Environment (JRE) which is at least as old as the program.
   c. A Java program which was developed in an old version of Java is likely to recompile and run without errors on a recent Java Runtime Environment (JRE).
   d. Any previous version of the Java Development Kit (JDK) can be used to develop a Java program for the current Java Runtime Environment.

2. **[3 marks]** The JavaDoc comment for a method should be placed

   a. Just after the method body.
   b. Just before the method body.
   c. In the body of the method.
   d. Just before the method declaration.

3. **[3 marks]** The first line of a JavaDoc comment for a method should be

   a. A description of the most recent defect-repair on this method.
   b. A list of the method's parameters.
   c. A brief description of the method.
   d. The author's name.

4. **[3 marks]** If you write an application in which a Java method invokes itself, then

   a. Your application may deliver useful results.
   b. Your application will not compile.
   c. Your application will compile but it will not run.
   d. Your application will compile and run, but it will throw a runtime exception when the method calls itself for the first time.

5. **[3 marks]** Creating a new method by extracting some lines of code from an existing method is called

   a. Redesigning.
   b. Reworking.
   c. Refactoring.
   d. Remodelling.

6. **[3 marks]** What will happen when the following Java expression is evaluated?

```
String.format("%.1f", 12.345)
```

   a. The value `"12.3"` is returned by the `format()` method, and there is no console output.
   b. The value `"12.3"` is returned by the `format()` method, and the value `"12.3"` is printed to the console.
   c. The value `"12.3"` is printed to the console.

7. **[3 marks]** What value is assigned to a reference variable when it refers to no object?

   a. `void`
   b. `0`
   c. `null`
   d. `""`

8. **[2 marks]** We say a constructor is overloaded if the runtime system is unable to allocate sufficient memory to create a new object of that type.

   a. True
   b. False

9. **[2 marks]** An import statement may use the asterisk (*) wildcard character, to indicate that multiple packages should be imported.

   a. True
   b. False

10. **[2 marks]** An import statement may use the asterisk (*) wildcard character, to indicate that all classes and interfaces of a package should be imported.

    a. True
    b. False

11. **[2 marks]** A static import statement can be used to import `enum` types.

    a. True
    b. False

12. **[2 marks]** A package statement may be the first line of a Java source file.

    a. True
    b. False

13. **[2 marks]** If a package is not imported, then its types cannot be referenced and its methods cannot be invoked.

    a. True
    b. False

14. **[2 marks]** A class declaration may be the first line of a Java source file.

    a. True
    b. False

15. **[2 marks]** A method declaration may be the first line of a Java source file.

    a. True
    b. False

16. **[2 marks]** `this()` will invoke the default constructor, if it is executed in the body of a constructor with at least one parameter in its signature.

    a. True
    b. False

# Section B

Recall the Swing application you developed in Assignment 1, then answer the following True/False questions.

17. **[2 marks]** Every Swing application must explicitly instantiate at least one customised event handler, using the `new` keyword.

    a. True
    b. False

18. **[2 marks]** Every Swing application must explicitly register at least one customised event handler, by defining a class which implements the `ActionListener` interface.

    a. True
    b. False

19. **[2 marks]** A Swing application may have no customised event handlers or painters.

    a. True
    b. False

# Section C

The True/False questions in this section refer to the Java application listed below. This is a valid Java application, *i.e.* it will compile and run without any errors.

```
public class HuckleBuckle {
    public static String VERSION = "1.5";
    public static void main(String[] args) {
         System.out.println("Version = " + VERSION);
    }
}
```

20. **[2 marks]** If some other method in this application modifies the value of `VERSION`, the new value of `VERSION` must be either a `String` or a subtype of `String`.

    a. True
    b. False

21. **[2 marks]** The `out` object must be of type `System` or some subtype of `System`.

    a. True
    b. False

22. **[2 marks]** `main()` must be defined as a `void` method, otherwise this is not a valid Java application.

    a. True
    b. False

23. **[2 marks]** `main()` must be defined as a `static` method, otherwise this is not a valid Java application.

    a. True
    b. False

24. **[2 marks]** If `args` is typed as an `Integer[]`, then this class would still compile but it would not be a Java application.

    a. True
    b. False

25. **[2 marks]** The developer of this code has capitalised `VERSION` appropriately.

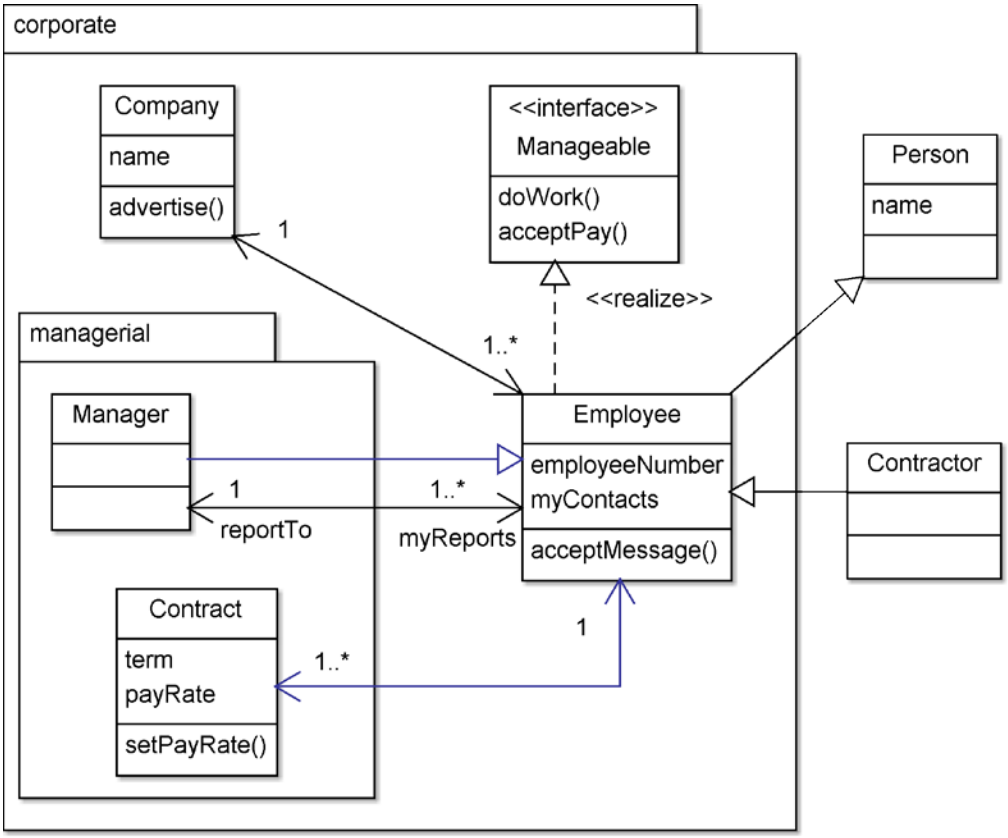
    a. True
    b. False

26. **[2 marks]** The `println()` method is invoked with two parameters of type `String`.

    a. True
    b. False

# Section D

The questions in this section refer to the following class diagram.




27. **[3 marks]** Which of the following options is the most appropriate visibility for the `setPayRate()` method, if managers (but not other employees) should be invoking this method?

    a. The `setPayRate()` option should be `private`.
    b. The `setPayRate()` option should have default visibility.
    c. The `setPayRate()` option should be `public`.
    d. None of the above.

28. **[2 marks]** A contractor might not have a contract.

    a. True
    b. False

29. **[3 marks]** If the `doWork()` method has `protected` visibility, would a contractor be able to invoke the `doWork()` method of a manager?

    a. No, because all methods of an interface must be `public`.
    b. Yes.
    c. No, because this access is not allowed for a `protected` method.

30. **[2 marks]** Every employee has exactly one contract.

    a. True
    b. False

31. **[2 marks]** One of the navigabilities in this OO design could be implemented with a class variable, of type `Collection<Employee>`, in the `Manager` class.

    a. True
    b. False

32. **[2 marks]** The source file for the `Manager` class must include an implementation of the `doWork()` and `acceptPay()` methods.

    a. True
    b. False

33. **[2 marks]** A competent developer would define and implement some methods which are not named in this class diagram.

    a. True
    b. False

34. **[3 marks]** Which of the following options would be the most appropriate signature for a `doWork()` method which instructs an employee to lead a team?

    a. `String doWork(String task, List<Manageable> team);`
    b. `String doWork(String task, List<Person> team);`
    c. `String doWork(String task, Collection<Manageable> team);`
    d. `String doWork(String task, Collection<Person> team);`

35. **[3 marks]** What visibility should the `advertise()` method have, if a company may engage a contractor to supply the text for its advertisements?

    a. The `advertise` method should be `private`.
    b. The `advertise` method should be `protected`.
    c. The `advertise` method should be `public`.
    d. The OO design should be modified, so that the `Contractor` class is defined in the `corporate` package.
    e. The `advertise` method should have default visibility.

36. **[3 marks]** If the `acceptMessage()` method has default visibility, would a contractor be able to invoke the `acceptMessage()` method of another contractor?

    a. No, this invocation will not compile.
    b. Yes.
    c. No, this invocation will throw a runtime exception.

37. **[3 marks]** Assume the `acceptMessage()` method of `Employee` has signature `void acceptMessage(String msg, Manageable sender)`. Also assume the `myContacts` field of `Employee` is of type `Set<Person>`. Would invoking `myContacts.add(sender)` in the body of the `acceptMessage()` method be a reliable way to update an employee's set of contacts?

    a. No, there would be a compilation error.
    b. Yes, the employee's set of contacts would be updated to include the sender.
    c. No, there would be a runtime error.

# Section E

This section of the test refers to the following description of a children's game.

> Huckle Buckle Beanstalk, also called Hide the Object or Hide the Key, is a childhood game which involves the hiding and seeking of an object. It is a variation of a traditional parlour game which can be played with two or more players, one being the hider, or the person who is "it," and the other person or persons being seekers. …
>
> The seekers must cover their eyes and ears or leave the designated game area while the hider hides a small, pre-selected object. When the hider says to come and find it, or after the seekers have counted to a specific number, usually sixty or one-hundred, the seekers come out and attempt to be the first to find the object. When a seeker has the object in hand, he can alert the other players of his success by yelling "huckle buckle beanstalk!".
>
> [http://en.wikipedia.org/wiki/Huckle_buckle_beanstalk, 23 March 2015 at 14:28]

38. **[3 marks]** Does the following use-case diagram have any major defects?

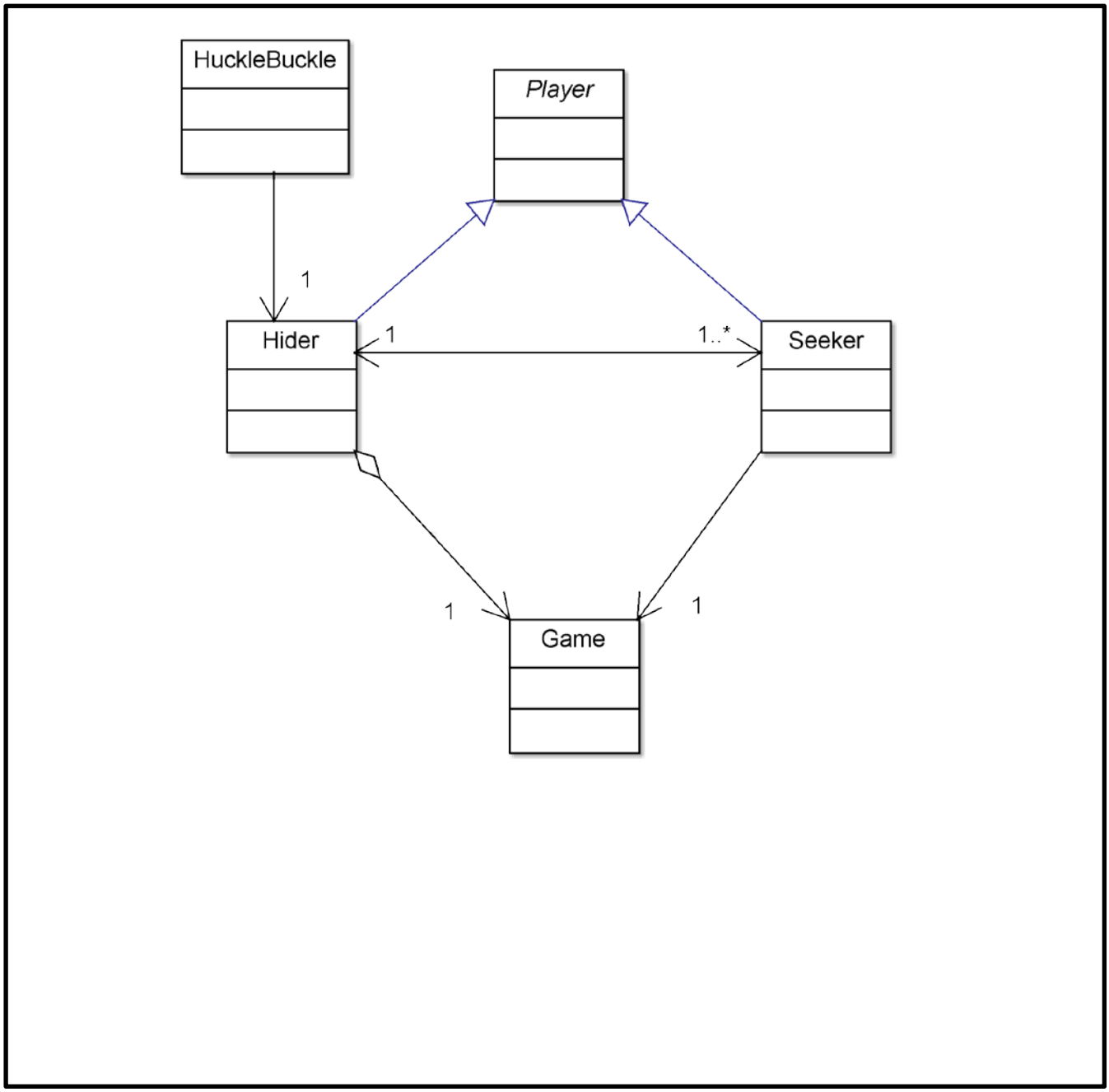

a. Yes, the semantics of this diagram are inaccurate.
b. Yes, this diagram should show more semantic detail.
c. Yes, this diagram has a syntax error.
d. No, this diagram has no major defects.
e. Yes, this diagram should show less semantic detail.

# Section F

39) **[4 marks]** Add a class called `HiddenKey` to the following class diagram, so that it describes an OO design which ***accurately represents*** the Huckle Buckle Beanstalk game described in Section E.

You should not name any variables or methods. You should not add any interfaces. You should add no classes other than `HiddenKey`.

To receive full marks, your class diagram must show the ***multiplicity*** and ***navigability*** of all important associations in your OO design.



40) **[6 marks]** Briefly ***explain*** the major design decision(s) you made when answering the previous question.

ID …….

## Overflow pages – please number answers carefully

## Overflow pages – please number answers carefully