CSE 1020 Introduction to Computer Science I

April 9, 2014

1 (6 marks)

Consider the following code snippet.

```
output.print("Enter the daily amount: ");
double da = input.nextDouble();
double wa = da * 7;
output.printf("Weekly amount: %.2f\n", wa);
if (wa == 1000) {
output.println("Weekly amount is 1000!");
}
```

Mention things that can be improved. There are at least six things that can be improved.

Solution

- Replace the cryptic variable name `da` with a descriptive one such as `dailyAmount`.
- Replace the cryptic variable name `wa` with a descriptive one such as `weeklyAmount`.
- Introduce a constant such as

  ```
  final int DAYS_PER_WEEK = 7;
  ```

  and use it instead of 7.
- Introduce a constant such as

  ```
  final int SPECIAL_AMOUNT = 1000;
  ```

  and use it instead of 1000.
- Instead of using == to compare real numbers, use something like

  ```
  final double EPSILON = 0.00001;
  if (Math.abs(weeklyAmount - SPECIAL_AMOUNT) < EPSILON) {
  ```

- Indent the body of the if-statement.

Also considered reasonable suggestions for improvement are

- Instead of

  ```
  output.println("Weekly amount is 1000!");
  ```

  use

  ```
  output.printf("Weekly amount is %d!", SPECIAL_AMOUNT);
  ```

- Declare and initialize `input` and `output`.
- Handle incorrect input.

Marking scheme
One mark for each improvement (up to a maximum of six marks).

2 (3 marks)

Explain why the expression `2 + 1 / 2` evaluates to `2`.

Solution
`/` binds stronger than `+`, so we first compute `1 / 2`. Since both `1` and `2` are of type `int`, `/` in `1 / 2` is integer division. Hence, `1 / 2` evaluates to `0`. Therefore, `2 + 1 / 2` evaluates to `2`.

Marking scheme
Two marks for the observation that integer division is used. One mark for mentioning that `/` binds stronger than `+`.

3 (6 marks)

You are to develop an app that prompts the user to enter zero or more integers, separated by a space, and prints the average of the entered integers. You may assume that the user always provides valid input.

(a) Give an example of a *compile-time error* that you may encounter during the development of the app.

Solution
There are many correct answers, including

```
intsum = 0;
```

Marking scheme
Two marks for a compile-time error that makes sense within the context of the question.

(b) Give an example of a *run-time error* that you may encounter during the development of the app.

Solution
There are many correct answers, including "NumberFormatException is thrown when the user enters zero integers."

Marking scheme
Two marks for a run-time error that makes sense within the context of the question.

(c) Give an example of a *logic error* that you may encounter during the development of the app.

Solution
There are many correct answers, including "the average is computed as

```
double average = sum * number;
```

Marking scheme
Two marks for a logic error that makes sense within the context of the question.

4 (3 marks)

(a) **Explain** the difference between *pass-by-value* and *pass-by-reference*.

Solution
In pass-by-value the values of the arguments are passed, whereas in pass-by-reference the addresses of the arguments are passed.

Marking scheme
One mark for "in pass-by-value the values of the arguments are passed" and one mark for "in pass-by-reference the addresses of the arguments are passed."

(b) Which of the two does Java use?

Solution
Call-by-value.

Marking scheme
One mark for call-by-value.

5 (2 marks)

What is the main difference between a `String` and a `StringBuffer`?

Solution
A `String` immutable, whereas the state of a `StringBuffer` can be changed.

Marking scheme
One mark for the observation that `String`s are immutable and one mark for the observation that `StringBuffer`s are not.

6 (1 mark)

What is another name for the *inheritance* relation?

Solution
The is-a relationship.

Marking scheme
One mark for "is-a."

7 (5 marks)

What are the five phases of the iterative methodology?

Solution
Requirements (or definition and analysis), design, implementation (or coding and unit testing), testing (or integration testing), evaluation.

Marking scheme
One mark for each correct phase.

8 (6 marks)

Consider the following code snippet.

```
Fraction fraction = new Fraction(1, 2);
long numerator = fraction.getNumerator();
long denominator = fraction.getDenominator();
numerator = numerator * 2;
Fraction twice = new Fraction(numerator, denominator);
```

Draw the diagram representing memory once the execution reaches the end of the above code snippet. Make sure that the variables `fraction`, `numerator`, `denominator` and `twice` are reflected in your diagram. Include blocks for the `main` method and the objects.

Solution

100	main	
	200	fraction
	2	numerator
	2	denominator
	300	twice
⋮		
200	Fraction object	
	1	numerator
	2	denominator
⋮		
300	Fraction object	
	2	numerator
	2	denominator

Marking scheme
One mark for each variable with correct value in the main block. One mark for each of the Fraction object blocks.

9 (6 marks)

In the table below, you find some basic constructs for regular expressions.

`[a-m]`	Range. A character between `a` and `m`, inclusive.
`[a-m[A-M]]`	Union. `a` through `m` or `A` through `M`.
`[abc]`	Set. The character `a`, `b` or `c`.
`[^abc]`	Negation. Any character except `a`, `b` or `c`.
`[a-m&&[^ck]]`	Intersection. `a` through `m` but neither `c` nor `k`.
`.`	Any character.
`\d`	A digit, `[0-9]`.
`\s`	A whitespace character. `[ \t\n\x0B\f\r]`.
`\w`	A word character, `[a-zA-Z_0-9]`.
x`?`	x, once or not at all.
x`*`	x, zero or more times.
x`+`	x, one or more times.
$x\{m, n\}$	x, at least m but no more than n times.

(a) Does the string `axR` match the regular expression `[a-d]?[xyz]+[R]`? **Explain** your answer.

Solution
Yes. `a` matches `[a-d]`. Hence, `a` matches `[a-d]?`. `x` matches `[xyz]`. Therefore, `x` matches `[xyz]+`. `R` matches `[R]`. Combining all, we can conclude that `axR` matches `[a-d]?[xyz]+[R]`.

Marking scheme
One mark for yes. One mark for the explanation.

(b) Does the string `abR` match the regular expression `[a-d]?[xyz]+[R]`? **Explain** your answer.

Solution
No. The string should contain `x`, `y` or `z`. `abR` does not.

Marking scheme
One mark for no. One mark for the explanation.

(c) Does the string `yR` match the regular expression `[a-d]?[xyz]+[R]`? **Explain** your answer.

Solution
Yes. The empty string matches `[a-d]?`. `y` matches `[xyz]`. Therefore, `y` matches `[xyz]+`. `R` matches `[R]`. Combining all, we can conclude that `yR` matches `[a-d]?[xyz]+[R]`.

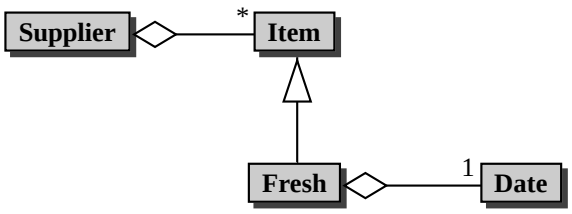
Marking scheme
One mark for yes. One mark for the explanation.

10 (5 marks)

The classes `Supplier`, `Item`, `Fresh` and `Date` are related as follows. A `Supplier` has zero or more `Items`. `Fresh` represents a special kind of `Item`, namely one that is fresh. Each `Fresh` item has a best before `Date`.

Draw a UML diagram of the above classes and their relations.

Solution



Marking scheme
One marking for each correct relation. One mark for $*$ and one mark for 1.

11 (6 marks)

The class `Car` has an attribute named `colour` of type `Colour`. The class `Colour` has attributes named `red`, `green` and `blue`, all of type `int`.

Consider the following code snippet.

```
Car car = new Car();
Colour black = new Colour();
black.setRed(0);
black.setGreen(0);
black.setBlue(0);
car.setColour(black);
Colour colour = car.getColour();
output.println(colour == black);
output.println(colour.equals(black));
```

(a) Assume that the classes `Car` and `Colour` form an ordinary *aggregation*. What output does the above snippet produce? **Explain** your answer.

Solution
`true`
`true`
The method `getColour` returns a reference to the `Colour` object created in the code snippet. Hence, `colour` and `black` refer to the same object.

Marking scheme
One mark for each line of correct output. One mark for the observation that `getColour` returns the same object.

(b) Assume that the classes `Car` and `Colour` form a *composition*. What output does the above snippet produce? **Explain** your answer.

Solution
`false`
`true`
The method `getColour` returns a reference a copy of the `care`'s colour, which is a copy of the `Colour` object created in the code snippet. Hence, `colour` and `black` have the same state but refer to the different objects.

Marking scheme
One mark for each line of correct output. One mark for the observation that `getColour` returns a copy.

12 (6 marks)

The class `CreditCard` contains the method

```
public boolean isSimilar(CreditCard other)
```

The class `RewardCard` extends the class `CreditCard` and contains the method

```
public boolean isSimilar(RewardCard other)
```

Consider the following code snippet.

```
CreditCard mine = new RewardCard(...);
CreditCard yours = new RewardCard(...);
boolean similar = mine.isSimilar((RewardCard) yours);
output.println(similar);
```

(a) During *early binding*, to which method of which class is the invocation `mine.isSimilar((RewardCard) yours)` bound? **Explain** your answer.

Solution
`isSimilar(CreditCard)` of `CreditCard`.
The declared type of `mine` is `CreditCard`. `CreditCard` contains only one matching method.

Marking scheme
One mark for `isSimilar(CreditCard)` of `CreditCard`. One mark for the observation that the declared type of `mine` is `CreditCard`. One mark for the observation that `CreditCard` contains only one matching method.

(b) During *late binding*, to which method of which class is the invocation `mine.isSimilar((RewardCard) yours)` bound? **Explain** your answer.

Solution
`isSimilar(CreditCard)` of `RewardCard`, inherited from `CreditCard`.
The actual type of `mine` is `RewardCard`. The signature stays the same.

Marking scheme
One mark for `isSimilar(CreditCard)` of `RewardCard` (or `CreditCard`). One mark for the observation that the actual type of `mine` is `RewardCard`. One mark for the observation that the signature stays the same.

13 (6 marks)

Consider the following code snippet.

```
try {
   output.println(args[0]);
} catch (IndexOutOfBoundsException e) {
   output.println(e.getMessage());
} catch (ArrayIndexOutOfBoundsException e) {
   e.printStackTrace();
}
```

It gives rise to the following compile-time error.

```
Test.java:5: exception java.lang.
ArrayIndexOutOfBoundsException has already been caught
   catch (ArrayIndexOutOfBoundsException e)
   ^
1 error
```

(a) **Explain** what is wrong with the code snippet.

Solution
The catch blocks are matched in the order in which they appear in the code. However, since an `ArrayIndexOutOfBoundsException` is-a `IndexOutOfBoundsException`, it already matches the first block. Hence, the second block is never reached.

Marking scheme
Two marks for the observation that `ArrayIndexOutOfBoundsException` is-a `IndexOutOfBoundsException`. One mark for the observation that the first block matches the `ArrayIndexOutOfBoundsException`.

(b) **Explain** how to fix it.

Solution
Either change the order of the catch blocks or leave out the second block.

Marking scheme
Three marks for either solution.

14 (2 marks)

What are the two main differences between a `List` and a `Set`?

Solution
A `List` may contains duplicates, whereas a `Set` does not contain duplicates. The elements in a `List` are ordered, whereas the elements in a `Set` are not.

Marking scheme
One mark for each difference.

15 (4 marks)

(a) The precondition of linear (traversal based) search is .....

Solution
true (no precondition)

Marking scheme
One mark for the correct answer.

(b) The precondition of binary search is .....

Solution
The list is sorted.

Marking scheme
One mark for the correct answer.

(c) Linear (traversal based) search has complexity $O(.....)$, where n is the size of the list.

Solution
n

Marking scheme
One mark for the correct answer.

(d) Binary search has complexity $O(.....)$, where n is the size of the list.

Solution
$\log(n)$

Marking scheme
One mark for the correct answer.