NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING
MIDTERM ASSESSMENT FOR
Semester 1 AY2017/2018

CS2030 Programming Methodology II

October 2017 Time Allowed 80 Minutes

INSTRUCTIONS TO CANDIDATES

1. This assessment paper contains 9 questions and comprises 10 printed pages, including this page.
2. The last two pages of this assessment paper is the answer sheet.
3. Write all your answers in the answer sheet. Detach the answer sheet for submission at the end of the assessment.
4. The total marks for this assessment is 40. Answer **ALL** questions.
5. This is an **OPEN BOOK** assessment.
6. All questions in this assessment paper use Java 8 unless specified otherwise.
7. State any additional assumption that you make.

Part I

Multiple Choice Questions (12 points)

- For each of the questions below, select the most appropriate answer and **write your answer in the corresponding answer box on the answer sheet.**. Each question is worth 3 points.
- If multiple answers are equally appropriate, pick one and write the chosen answer in the answer box. Do NOT write more than one answers in the answer box.
- If none of the answers are appropriate, write `X` in the answer box.

1. (3 points) Consider the following definition of class `A`:

```
class A extends P implements I {
}
```

Which of the following statements would result in a compilation error?

(i) `Object a = new A();`
(ii) `I a = new P();`
(iii) `I a = new A();`
(iv) `A a = new I();`

A. Only (i)
B. Only (iii)
C. Only (ii) and (iii)
D. Only (i) and (iv)
E. Only (ii) and (iv)

Write `X` in the answer box if none of the combinations is correct.

Solution: E.

This question assesses if students understand the basic concepts of class and interface, as well as the meaning of keywords `extends` and `implements`.

From the definition of `a`, we know that `P` is the parent class of `A`, and `I` is an interface that `A` implements. (i) is OK (no error), because `A`, like every other class, is also a subclass of `Object`. For (ii), we have no information whether `P` implements `I` or not, and based only on the information given, this will result in an error. (If you state additional assumption that `P implements I`, then you can consider (ii) as no error). (iii) is obviously OK, and (iv) is not OK because we cannot instantiate an interface.

2. (3 points) Consider the program below:

```
class A {
  static void f() throws Exception {
    try {
      throw new Exception();
    } finally {
      System.out.print("1");
    }
  }

  static void g() throws Exception {
    System.out.print("2");
    f();
    System.out.print("3");
  }

  public static void main(String[] args) {
    try {
      g();
    } catch (Exception e) {
      System.out.print("4");
    }
  }
}
```

What is the output of the program?

A. 24

B. 214

C. 23

D. 213

E. 2134

Write `X` in the answer box if none of the answers above is correct.

Solution: B. This question assesses if students understand the control flow of exceptions. Since `main()` calls `g()`, `2` will be printed first. `f()` is then executed, which led to an exception being thrown. Before `f()` returns, it executes the `finally` block, leading to `1` being printed next. After returning to `g()`, since an exception has been thrown, `3` will not be printed. The control returns to `main()`, which catch the exception. The `catch` block is executed and `4` is then printed.

3. (3 points) Consider the program below:

```
class Main {

  static class P {
    public void f(int i, int j) {
      System.out.print(i + j);
    }
  }

  static class A extends P {
    public void f(Integer i, Integer j) {  // Note the type
      System.out.print(i * j);
    }
  }

  static class B extends P {
    public void f(int j, int i) {  // Note the order of i and j
      System.out.print(i - j);
    }
  }

  public static void main(String[] args) {
    P objects[] = { new P(), new A(), new B() };
    for (P p : objects) {
      p.f(1, 2);
    }
  }
}
```

What is the output of the program?

A. 321

B. 331

C. 221

D. 222

E. 333

Write X in the answer box if none of the combinations is correct.

Solution: B. This question assesses if students understand the concept of dynamic binding.

In `main`, instances of `P`, `A` and `B` are created and assigned to variable `p` of compile-time type `P`. When `p.f()` is called, due to polymorphism, the method `f()` will be bound to the version of `f` belonging to the class `P` (i.e., `void f(int, int)`) during compilation.

During run-time, `p` takes on different run-time type, in the order of `P`, `A`, and `B`. The method with the descriptor `void f(int, int)` from `P`, `P`, and `B` will be invoked. Note that for `A`, there is no method with the matching descriptor so the one from its parent will be invoked.

4. (3 points) Consider a method declared as `int foo(double x)`.

   Which of the following statements will NOT result in a compilation error:

   (i) `int cs = foo(2030);`

   (ii) `double cs = foo(2.030);`

   (iii) `int cs = foo(new Double(2.030));`

   (iv) `Integer cs = foo(new Double(2.030));`

   A. (i) and (iv) only

   B. (ii) and (iv) only

   C. (ii) and (iii) only

   D. (i), (ii), and (iii) only

   E. (i), (ii), (iii), and (iv)

   Write `X` in the answer box if none of the combinations is correct.

Solution: E. This question assesses students about type conversion, primitive types, and auto-boxing/unboxing. (i) passes an int value `2030` to a double argument, which is allowed. (ii) again assigns the returned `int` value to a double, again allowed. (iii) a `Double` object to a primitive double, unboxing happens. (iv) assigns an `int` value to an `Integer` reference, autoboxing happens.

Part II

Short Questions (28 points)

Answer all questions in the space provided on the answer sheet. Be succinct and write neatly.

5. (3 points) **LSP.** Consider the following classes: `FormattedText` adds formatting information to the text. We call `toggleUnderline()` to add or remove underlines from the text. A `URL` is a `FormattedText` that is always underlined.

```
class FormattedText {
  public String text;
  public boolean isUnderlined;

  public void toggleUnderline() {
    isUnderlined = (!isUnderlined);
  }
}

class URL extends FormattedText {
  public URL() {
    isUnderlined = true;
  }
  public void toggleUnderline() {} // do nothing
}
```

Does it violate the Liskov Substitution Principle? Explain.

Solution: Yes. `URL` changes the behavior of `toggleUnderline()`, so the property that `toggleUnderline` toggles the `isUnderlined` flag no longer holds for subclass `URL`. Places in a program where `FormattedText` is used cannot be simply replaced by `URL`.

6. (6 points) **Type.** Explain how each of the following language features of Java ensures type safety. You can give an example.
   (a) (3 points) enum
   (b) (3 points) generics

Solution: (a) enum allows a type to be defined and used for a set of predefined constants. Using a constant other than those predefined would lead to a compilation error. In contrast, using `int` is not type safe since `int` values other than those predefined can be accidentally assigned / passed as arguments.

(b) generics allow classes / methods that use any reference type to be defined without resorting to using the `Object` type. It enforce type safety by binding the generic type to a specific given type argument at compile time. Attempt to pass in an incompatible type would led to compilation error.

7. (8 points) **Wildcards.** For each of the statement below, indicate if it is a valid statement (no compilation error). Explain why in one sentence.
   (a) (2 points) `List<?> list = new ArrayList<String>();`
   (b) (2 points) `List<? super Integer> list = new List<Object>();`
   (c) (2 points) `List<? extends Object> list = new LinkedList<Object>();`
   (d) (2 points) `List<? super Integer> list = new LinkedList<>();`

Solution: (a) Yes. `ArrayList<String>` <: `List<String>` <: `List<?>` (b) No. Cannot instantiate an interface `List`. (c) Yes. `LinkedList<Object>` <: `List<Object>` <: `LinkedList<? extends Object>`. (d) Yes. `LinkedList<Integer>` <: `List<Integer>` <: `LinkedList<? super Integer>`. Java infers the type to be `Integer`.

Note that simply stating yes/no without explanation will not get your any marks.

8. (3 points) **Nested.** Consider the program below:

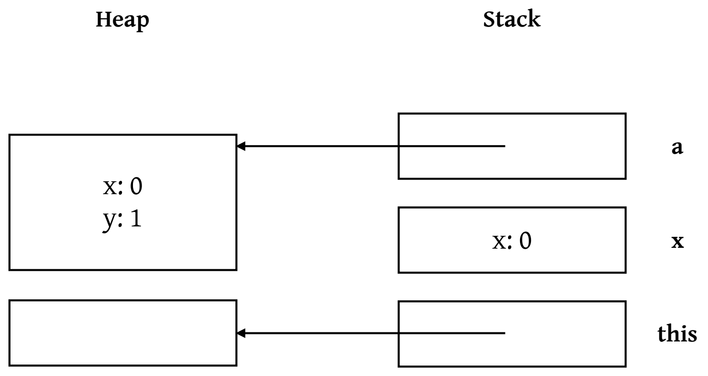
```java
class B {
  void f() {
    int x = 0;

    class A {
      int y = 0;
      A() {
        y = x + 1;
      }
    }

    A a = new A();
  }
}
```

Suppose that a variable `b` is an instance of class `B`, and a program calls `b.f()`. Sketch the content of the stack and heap immediately after the Line `A a = new A()` is executed. Label the values and variables / fields clearly. You can assume `b` is already on the heap and you can ignore all other content of the stack and the heap before `b.f()` is called.

Solution:



This question assesses students understanding how memory is allocated in Java. If you draw the the local variables `a` and `x` on the stack with the object of `A` on the heap, you get 1 mark. If you additional draw the `this` reference (which is strongly hinted), you get another. Finally, realizing the `A` is a nested class and its instance captures the variable `x` would give you another mark.

9. (8 points) **Bus.** You are asked to write a program that prints out, given a bus stop, the list of bus services that serves the bus stop. Bus services and bus stops are identified by names. A name is a string that does not contain any space.

   The query of which bus services serve a bus stop, however, is not readily available. You are given instead, a file that lists which bus stops a bus service serves. The file contains one bus service per line. The first name is the name of the bus service, the rest of the line contains the name of the bus stops. We assume that each bus stop appears only once per bus service (no repeats).

   For example, the following shows the first few lines of a sample input file for NUS internal shuttle buses:

```
A1 PGP KR-MRT LT29 UHall OppUHC YIH CL LT13 AS7 COM2 BIZ2 OppH12 H7
A2 PGP H17 H12 OppHSS OppNUSS COM2 Ventus CCE OppYIH Museum UHC OppUHall S16 OppKR-MRT
B1 KR-Bus CCE OppYIH UTown YIH CL LT13 AS7 BIZ2
C KR-Bus KentVale Museum UHC OppUHall S17 LT29 UHall OppUHC RafflesHall EA
D1 OppHSS OppNUSS COM2 Ventus CCE OppYIH Museum UTown YIH CL LT13 AS7 BIZ2
  :
```

   If you are given an input "`COM2`", the program should print "`A1 D1 A2`", since COM2 is served by these three bus services. The order of the bus services printed is not important. Of course, your program should work for any other bus companies (e.g., SBS or SMRT buses).

   (a) (4 points) Which class or classes from Java Collections Framework would you use to store the information read from the input file so that you can answer the query "which bus services serve this bus stop?" *efficiently*? Explain why you choose them and how you would use them. You only need to choose from `ArrayList`, `LinkedList`, `HashSet`, `HashMap`, and `PriorityQueue`.

   Solution: Use `HashMap<BusStop, HashSet<BusService>>` to map from the bus stop to a set of bus services. Given the name of a bus stop, just look up in the `HashMap` the bus stop object and its list of bus services.

   Some students do it the other way around – `HashMap<BusService, HashSet<BusStop>>` or variations of this. To answer the query, you would then have to scan through all the bus services to see if the current bus stop is in the set. You only get 2 out of 4 marks for such answers.

   Note that despite the question hinted strongly there is no repeat and order is not important, some of you still use a list instead of a set to store the bus services. I accepted such answers anyway. Using a list of lists, however, is too far off. Such answer would get only 1 mark, at most.

   (b) (4 points) Write two classes, in Java syntax, that you would use to encapsulate the information read from the input file and the operations necessary to answer the query "which bus services serve this bus stop?". In writing down the classes, you can skip:

   - the implementation of the methods (just write `{...}`)

- private methods
- methods and fields related to input and output

Solution: Any two of the classes similar to below:

```
class BusSystem {
  private HashMap<BusStop,BusServices> map;
  BusSystem() { ... }
  public void add(BusStop) { ... }
  public void add(BusStop, BusService) { ... }
  public BusServices getBusServices(BusStop) { ... }
}

class BusServices {
  private HashSet<BusService> services;
  BusServices() { ... }
  public add(BusService service) { ... }
  public addStop(BusStop stop) { ... }
  public String toString() { ... }
}

class BusStop {
  private String name;
  private BusServices services;
  BusStop() { ... }
  public void addService(BusService service) { ... }
  public boolean equals() { ... }
  public int hashCode() { ... }
}

class BusService {
  private String name;
  BusServices() { ... }
  public void addStop(BusStop stop) { ... }
  public boolean equals() { ... }
  public int hashCode() { ... }
}
```

A large number of answers does not encapsulate the bus stops and bus services, but instead uses String directly (return a string containing all the bus services). Such answers show lack of understanding and appreciation of encapsulation and type safety.

END OF PAPER