CS2030/S Programming Methodology
Semester 1 2020/2021

16 September 2020
Problem Set #4

1. Consider a generic class `A<T>` with a type parameter `T` having a constructor with no argument. Which of the following expressions are valid (with no compilation error) ways of creating a new object of type `A`? We still consider the expression as valid if the Java compiler produces a warning.

   (a) `new A<int>()`

   (b) `new A<>()`

   (c) `new A()`

2. In the Java Collections Framework, `List` is an interface that is implemented by both `ArrayList`. For each of the statements below, indicate if it is a valid statement with no compilation error. Explain why.

   (a)
   ```
   void foo(List<?> list) { }

   foo(new ArrayList<String>())
   ```

   (b)
   ```
   void foo(List<? super Integer> list) { }

   foo(new List<Object>())
   ```

   (c)
   ```
   void foo(List<? extends Object> list) { }

   foo(new ArrayList<Object>())
   ```

   (d)
   ```
   void foo(List<? super Integer> list) { }

   foo(new ArrayList<int>())
   ```

   (e)
   ```
   void foo(List<? super Integer> list) { }

   foo(new ArrayList());
   ```

3. In the lecture, we have shown the use of the `Comparator<T>` interface with the abstract method `int compare(T t1, T t2)` that returns zero if `t1` and `t2` are equal, a negative integer if `t1` is less than `t2`, or a positive integer if `t2` is less than `t1`.

   A generic method `T max3(T a, T b, T c, Comparator<T> comp)` is defined below. The method takes in three values of type `T` as well as a `Comparator<T>`, and returns the maximum among the values.

```
<T> T max3(T a, T b, T c, Comparator<T> comp) {
    T max = a;
    if (comp.compare(b, max) > 0) {
        max = b;
    }
    if (comp.compare(c, max) > 0) {
        max = c;
    }
    return max;
}
```

(a) Demonstrate how the `max3` method is called to return the maximum of three integers -1, 2 and -3.

(b) Other than `Comparator<T>`, there is a similar `Comparable<T>` interface with the abstract method `int compareTo(T o)`. This allows one `Comparable` object to compare itself against another `Comparable` object. Now we would like to redefine the `max3` method to make use of the `Comparable` interface instead.

```
<T> T max3(T a, T b, T c) {
    T max = a;
    if (b.compareTo(max) > 0) {
        max = b;
    }
    if (c.compareTo(max) > 0) {
        max = c;
    }
    return max;
}
```

Does the above method work? What is the compilation error?

(c) Now, we further restrict `T` to be `Comparable<T>`

```
<T extends Comparable<T>> T max3(T a, T b, T c) {
    T max = a;
    if (b.compareTo(max) > 0) {
        max = b;
    }
    if (c.compareTo(max) > 0) {
        max = c;
    }
    return max;
}
```

Demonstrate how the method `max3` can be used to find the maximum of three values -1, 2 and -3. Explain how it works now.

(d) What happens if we replace the method header with each of the following:

i. `<T> Comparable<T> max3(Comparable<T> a, Comparable<T> b, Comparable<T> c)`

ii. `<T> T max3 (Comparable<T> a, Comparable<T> b, Comparable<T> c)`

iii. `Comparable max3(Comparable a, Comparable b, Comparable c)`