CS2030 Programming Methodology II

Semester 2 2022/2023

8 & 9 September 2023

Problem Set #3

Abstract Class and Interface

1. Given the following interfaces.

```
interface Shape {
    double getArea();
}

interface Printable {
    void print();
}
```

   (a) Suppose class `Circle` implements both interfaces above. Given the following program fragment,

```
Circle c = new Circle(10);
Shape s = c;
Printable p = c;
```

   Are the following statements allowed? Why do you think Java does not allow some of the following statements?

   i. `s.print();` not allowed, as Shape does not have a print() method

   ii. `p.print();` allowed

   iii. `s.getArea();` allowed

   iv. `p.getArea();` not allowed, as Printable does not have a getArea() method

   (b) Someone proposes to re-implement `Shape` and `Printable` as abstract classes instead? What happens? Circle can only inherit from either Shape or Printable

   (c) Now let's define another interface `PrintableShape` as

```
interface PrintableShape extends Printable, Shape { }
```

   and let class `Circle` implement `PrintableShape` instead.
   Can an interface inherit from multiple parent interfaces? Yes Would the following statements be allowed?

```
Circle c = new Circle(10);
PrintableShape ps = c;
```

   i. `ps.print();` Yes

   ii. `ps.getArea();` Yes

2. Suppose Java allows a class to inherit from multiple parent classes. Give a concrete example why this could be problematic. Why does Java allow classes to implement multiple interfaces then?

As abstract classes have method implementation, it could be troublesome when a child inherits the from two parents which have the same method but different implementations. The child would not know which parent method to follow.

3. Consider the following program.

```
class A {
    protected final int x;

    A(int x) {
        this.x = x;
    }

    A method() {
        return new A(x);
    }
}

class B extends A {
    B(int x) {
        super(x);
    }

    @Override
    B method() {
        return new B(x);
    }
}
```

Overriding method can
override if the children class
is more specific
e.g.
B is a children of A

Yes

Does it compile? What happens if we swap the entire definitions of `method()` between class `A` and class `B`? Does it compile now? Give reasons for your observations.

Swapping definitions of method() would not work

Return type of
overriding method
cannot be more
general