This exam has 4 additional pages with 4 questions.
1. [25 pts] Consider the following code:

```java
class A {
    abstract public void foo();
}

class B extends A {
    public B() {
        this("");
    }
    public B(String bm) {
        bmsg = bm;
    }

    public String foo() {
        return bmsg;
    }

    private String bmsg;
}

class C extends B {
    public C(String bm, String cm) {
        super(bm);
        cmsg = cm;
    }
    public C() {
        this("", "");
    }

    public String foo() {
        return cmsg + super.foo();
    }

    private String cmsg;
}
```

(a) The compiler is complaining about the implementation of class A. What is the problem and the fix?

(b) Assuming class A is repaired, which of the following lines of code are legal?

- A obj = new A();
- A obj = new B();
- A obj = new C();
- B obj = new C();
- C obj = new B();

(c) What is the output of the following code?

```
A[] items = { new B("foo"), new C("foo", "bar")};
System.out.println(items[0].foo() + items[1].foo());
```
2. [25 pts] Answer each part TRUE or FALSE

(a) All methods in an abstract class must be abstract.
(b) An abstract class may provide constructors.
(c) An abstract class can declare instance data.
(d) An abstract class can extend another abstract class.
(e) An abstract class can extend a non-abstract class.
(f) An interface is an abstract class.
(g) An interface can declare instance data.
(h) Any method in an interface must be public.
(i) All methods in an interface must be abstract.
(j) An interface can have no methods at all.
(k) An interface can extend another interface.
(l) An interface can declare constructors.
(m) A class may extend more than one class.
(n) A class may implement more than one interface.
(o) A class may extend one class and implement one interface.
(p) An interface may implement some of its methods.
(q) Methods in an interface may provide a throws list.
(r) All methods in an interface must have a void return type.
(s) Throwable is an interface.
(t) Object is an abstract class.
3. [25 pts] Consider the following four classes: `WalkupTicket`, `AdvanceTicket`, `StudentAdvanceTicket`, and `Ticket`, which interact as follows:

- A `WalkupTicket` has a seat number and a `price` method that returns a `double`, but I am not telling you the exact price because you do not have to implement `WalkupTicket` on this exam.
- A `AdvanceTicket` has a set number and a `price` method that returns a `double`, but I am not telling you what the `double` is because you do not have to implement `AdvanceTicket` on this exam.
- A `StudentAdvanceTicket` IS-A `AdvanceTicket`. If the `AdvanceTicket`'s `price` method returns `d`, then the `StudentAdvanceTicket`'s `price` method returns `d/2`. Needless to say, if the `AdvanceTicket`'s `price` method changes to return a different price, then the `StudentAdvanceTicket`'s `price` method will automatically be aware of this.
- A `Ticket` has a seat number. Also, a `WalkupTicket` IS-A `Ticket` and a `AdvanceTicket` IS-A `Ticket`. Tickets are not intended to be constructed directly by the client (but of course, a `Ticket` still has a constructor).

For this question, do the following (You do not have to provide any functionality beyond the specifications above.):

(a) The four classes above form an inheritance hierarchy. Draw the hierarchy.
(b) Implement `Ticket`.
(c) Implement `StudentAdvanceTicket`.
(d) Implement the following method:

```java
// Return total price of all tickets
public static double totalPrice( Ticket[] arr )
{
```

```java
```
4. [25 pts] Method **contains** takes an array of integers and returns true if there exists any item in the array that satisfies a specified condition.

For instance, in the following code fragment:

```java
int [] input = { 100, 37, 49 };

boolean result1 = contains( input, new Prime( ) )
boolean result2 = contains( input, new PerfectSquare( ) )
boolean result3 = contains( input, new Negative( ) )
```

The intended result is that **result1** is true because 37 is a prime number, **result2** is true because both 100 and 49 are perfect squares, and **result3** is false because there are no negative numbers in the array.

Implement the following components:

(a) An interface that will be used to specify the second parameter to **contains**.
(b) The **contains** method (which is a static method).
(c) The class **Negative**.