Name:________________________________________ NetID:________________

Answer all questions in the space provided. Write clearly and legibly, you will not get credit for illegible or incomprehensible answers. This is a closed book exam. However, each student is allowed to bring one page of notes to the exam. Print your name at the top of every page.

| Question: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Total |
|---|---|---|---|---|---|---|---|---|
| Points: | 14 | 8 | 14 | 15 | 10 | 12 | 12 | 85 |
| Score: | | | | | | | | |

1. Multiple choice

   (a) If you wanted to store the value of the square root of 2 in a variable, which of the following types would be best? (2)

   A. boolean
   B. char
   C. double
   D. int
   E. String

   (b) Which of the following is *not* a Java keyword? (2)

   A. for
   B. while
   C. do
   D. if
   E. then
   F. else

   (c) A member declared with a `protected` access modifier is *not* visible to: (2)

   A. the class in which it is declared.
   B. classes in the same package as the class in which it is declared.
   C. classes that extend the class in which it is declared.
   D. parent classes of the class in which it is declared.
   E. classes nested inside the class in which it is declared.

   (d) Which combination of modifiers could *not* be used together to modify a class? (2)

   A. `private static final`
   B. `protected abstract final`
   C. `public static abstract`

   (e) Which combination of modifiers could be used together to modify a member variable? (2)

   A. `private static final`
   B. `protected abstract final`
   C. `public static abstract`

(f) Which code would you use to instantiate a new ArrayList that could *only* hold Strings? (2)

A. `... = <String>ArrayList();`

B. `... = new ArrayList<String>;`

C. `... = ArrayList<String>();`

D. `... = new ArrayList<String>();`

E. `... = String<ArrayList>();`

F. `... = new String[ArrayList];`

(g) Which of the following is true of an unchecked exception? (2)

A. It must be handled at compile time with a try/catch construct.

B. It is thrown because of unavoidable circumstances, such as a file not being found.

C. It extends `RuntimeException`.

D. It cannot be caught at runtime.

2. Consider the following classes.

```
public class ClassA {
  public void method1(int i) {
  }
  public void method2(int i) {
  }
  public static void method3(int i) {
  }
  public static void method4(int i) {
  }
}
```

```
public class ClassB extends ClassA {
  public void method1(float i) {
  }
  public void method2(int i) {
  }
  public static void method3(float i) {
  }
  public static void method4(int i) {
  }
}
```

(a) Does `method1` in `ClassB` override, overload, or hide the method in `ClassA`? (2)

(a) ____________

(b) Does `method2` in `ClassB` override, overload, or hide the method in `ClassA`? (2)

(b) ____________

(c) Does `method3` in `ClassB` override, overload, or hide the method in `ClassA`? (2)

(c) ____________

(d) Does `method4` in `ClassB` override, overload, or hide the method in `ClassA`? (2)

(d) ____________

3. Why do the following lines of code not compile?

   (a) `List<double> values;` (2)

   (b) `Set<Integer> values = new Set<Integer>();` (2)

   (c) `String[] names = String[50];` (2)

   (d) `if (x = 5) System.out.println(x);` (2)

   (e) `boolean break = true;` (2)

   (f) `int x = 5` (2)

   (g) `enum x = 5;` (2)

4. Consider the following interface.

```
public interface TestInterface {
    void doStuff(String s);
    boolean isItTrue(int i, double x);
}
```

For each of the following:

- Does this class implement the interface?
- If it does not, what is wrong with the implementation?

(a) (3)

```
public interface TestImplementation {

    public void doStuff(String s) {
    }

    public boolean isItTrue(int i, double x) {
        return true;
    }
}
```

(b) (3)

```
public class TestImplementation {

    public void doStuff(String s) {
    }

    public boolean isItTrue(int i, double x) {
        return true;
    }
}
```

(c) (3)

```
public class TestImplementation extends TestInterface {

    public void doStuff(String s) {
    }

    public boolean isItTrue(int i, double x) {
        return true;
    }
}
```

(d) (3)

```
public class TestImplementation implements TestInterface {

    public void doStuff(String s) {
    }

    public boolean isItTrue(int i, double x) {
        return true;
    }
}
```

(e) (3)

```
public class TestImplementation implements TestInterface {

    public void doStuff(String s) {
    }

    public boolean isItTrue(double i, int x) {
        return true;
    }
}
```

5. Write a method that takes a `Collection` of `String` objects and returns the length of the shortest one. If the collection is empty, return `Integer.MAX_VALUE` (a constant that represents the largest possible int value). (10)

6. Consider the following classes. What is the output of this code? (12)

```
public class Foo {
    protected int x;
    protected double y;
    protected String z;

    public Foo() {
        this("Exam");
    }

    public Foo(String x) {
        this(x, x.length());
    }

    public Foo(String x, int y) {
        this.x = y;
        this.y = y / 4.0;
        this.z = x;
    }

    public void print(String x) {
        System.out.println(x);
        System.out.println(y);
        System.out.println(z);
    }

    public void print(double z) {
        System.out.println(x);
        System.out.println(y);
        System.out.println(z);
    }
}
```

```
public class Bar extends Foo {

    public Bar(String y) {
        System.out.println(y);
        System.out.println(z);
    }

    public void print(int x) {
        print(x / 2.0);
    }

    public void print(String x) {
        print(x.length() / 3);
        System.out.println(x);
    }

    public static void main(String[] args) {
        Foo test = new Bar("Midterm");
        test.print("CS" + 251);
    }
}
```

7. Consider the following class. What is the output of this code? (12)

```
public class Baz {
    private static String x;
    private String y;

    public Baz(String z) {
        y = x;
        x = z;
    }

    public void printVals() {
        System.out.println(x);
        System.out.println(y);
    }

    public static void main(String[] args) {
        x = "enjoy";
        Baz b1 = new Baz("spring");
        b1.printVals();
        Baz b2 = new Baz("break");
        b1.printVals();
        b2.printVals();
    }
}
```