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:	12	8	12	15	12	12	10	81
Score:								

1. Multiple choice questions

(a) What is the value of the following expression? `"one" + 2 + 3 * 4` (2)

A. `"one234"`

B. `"one212"`

C. `"one14"`

D. `15`

E. This expression would result in a compilation error.

(b) What is displayed when the following code is compiled and executed? (2)

```java
public class EqualsTest {
    public static void main(String[] args) {
        String s1 = "CS251";
        String s2 = "CS";
        s2 += 251;

        if(s1 == s2) System.out.println("Same");
        if(s1.equals(s2)) System.out.println("Equals");
    }
}
```

A. `Same`

B. `Equals`

C. `Same`
`Equals`

D. The code compiles, but nothing is displayed upon execution.

E. The code fails to compile.

(c) Which of the following is *not* a keyword used in exception handling? (2)

A. finally

B. catch

C. final

D. throw

E. try

F. throws

(d) Which code could you use to instantiate a new HashMap that associates String keys with Integer values? (2)

A. `...  = <String, Integer>HashMap();`
B. `...  = new HashMap<Integer, String>();`
C. `...  = new HashMap[String, Integer];`
D. `...  = new HashMap<String, Integer>();`
E. `...  = new HashMap<String, int>();`
F. `...  = new HashMap<String, Integer>;`
G. `...  = HashMap<String, Integer>();`
H. `...  = new HashMap<String>(Integer);`

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

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

(f) A static variable with no access modifier could *not* be accessed by: (2)

A. A static method in the same class.
B. A non-static method in the same class.
C. A final method defined within the same package.
D. A private method defined in a different package.

2. Consider the following classes.

```
public class ClassA {
  public void method1(float i) {
  }
  public void method2(float i) {
  }
  public static void method3(float i) {
  }
  public static void method4(float 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 snippets of code not compile?

(a) (2)

```
boolean do = true;
```

(b) `Set<int> values;` (2)

(c) `List<Integer> values = ArrayList<Integer>();` (2)

(d) `Map<String, Integer> wordsToCounts = new Map<String, Integer>();` (2)

(e) (2)

```
public static String myMethod(int x) {
    if( x > 5 ) {
        return "bigger than five!";
    }
}
```

(f) (2)

```
public class MyClass {
    private int x = 10;

    public static void main(String[] args) {
        System.out.println(x);
    }
}
```

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 implements TestInterface {

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

    public void doStuff(String s) {
    }
}
```

(d) (3)

```
public class TestImplementation implements TestInterface {

    public void doStuff(String s) {
    }

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

(e) (3)

```
public class TestImplementation extends TestInterface {

    public void doStuff(String s) {
    }

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

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

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

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

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

    public Foo(String x, int y) {
        this.x = y * 0.5;
        this.y = y;
        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 / 4.0);
    }

    public void print(String x) {
        print(x.length() / 2);
        System.out.println(x);
    }

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

6. 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(" fall");
        b1.printVals();
        Baz b2 = new Baz(" break");
        b1.printVals();
        b2.printVals();
    }
}
```

7. Write a method that takes a `Collection` of `String` objects (any type of collection, not a specific implementation) and returns the length of the longest one. If the collection is empty, return -1. (10)