Name:_____ NetID:_____

Answer all questions in the space provided. Write clearly and legibly, you will not get credit for illegible or incomprehensible answers. Print your name at the top of every page.

This is a closed book exam. However, each student is allowed to bring one page of notes to the exam. Also, you are permitted the use of a "dumb" calculator to perform basic arithmetic.

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	10	14	6	8	16	12	14	10	10	100
Score:										

1. Consider the following classes. For each specified method, does the method in `Child` override, overload, or hide the method in `Parent`?

```
public class Parent {
  protected void method1(int i) { }
  protected void method2(int i) { }
  public static void method3(int i) { }
  public void method4(int i) { }
  public static void method5(int i) { }
}
```

```
public class Child extends Parent {
  public void method1(int i) { }
  public void method2(float i) { }
  public static void method3(float i) { }
  public void method4(int i) { }
  public static void method5(int i) { }
}
```

(a) `method1` (2)

(a) _____

(b) `method2` (2)

(b) _____

(c) `method3` (2)

(c) _____

(d) `method4` (2)

(d) _____

(e) `method5` (2)

(e) _____

2. Indicate if the following statements are true or false.

(a) A class can implement any number of interfaces. (2)

(a) _____

(b) A class can extend any number of classes. (2)

(b) _____

(c) An abstract class cannot contain any method implementations. (2)

(c) _____

(d) It is possible to use an interface as a type. (2)

(d) _____

(e) Any reference type can be assigned to an `Object` variable. (2)

(e) _____

(f) An unchecked exception cannot be caught with a try/catch construct. (2)

(f) _____

(g) Every class has a default no-argument constructor. (2)

(g) _____

3. Why don't the following code snippets compile? (Or do they?)

   Select the single correct answer for each.

(a) (2)

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

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

A. The println method expects a String, not an int.
B. Variable x is out of scope in the main method.
C. Cannot access private variable x from a public method.
D. Cannot access x without an instance of MyClass.
E. Variable x is a constant, so cannot be incremented in main.
F. Some other error.
G. This code will successfully compile.

(b) (2)

```
public static boolean foo(short n) {
  int default = 10;
  return n < default;
}
```

A. Cannot compare int and short variables.
B. Cannot use a keyword as a variable name.
C. Invalid parameter type.
D. Return value does not match method return type.
E. Did not initialize value of n.
F. Some other error.
G. This code will successfully compile.

(c) (2)

```
public static long bar(int n) {
    return (n % 2 == 0) ? n : 1;
}
```

A. Invalid operator % on return line.
B. Invalid operator ? on return line.
C. Return value does not match method return type.
D. Invalid return type.
E. Did not initialize value of n.
F. Some other error.
G. This code will successfully compile.

4. Why don't the following code snippets compile? (Or do they?)

   Select the single correct answer for each.

(a) (2)

```
Collection<Double> numbers = new Set<>();
```

   A. Cannot use Double as generic type parameter.
   B. Cannot assign a Set to a Collection variable.
   C. Cannot use interface Collection as variable type.
   D. Missing type parameter on right hand side of assignment.
   E. Set is an interface and cannot be instantiated.
   F. Some other error.
   G. This code will successfully compile.

(b) (2)

```
List<boolean> truthValues;
```

   A. The list is not initialized.
   B. Cannot use primitive type boolean as generic type parameter.
   C. Cannot use interface List as variable type.
   D. List is an interface and cannot be instantiated.
   E. Some other error.
   F. This code will successfully compile.

(c) (2)

```
List<String> names
```

   A. The list is not initialized.
   B. Cannot use String as generic type parameter.
   C. Cannot use interface List as variable type.
   D. List is an interface and cannot be instantiated.
   E. Some other error.
   F. This code will successfully compile.

(d) (2)

```
Map<Integer,String> idToNameMap = new HashMap<>();
```

   A. Cannot use Integer as generic type parameter.
   B. Cannot use String as generic type parameter.
   C. Cannot use interface Map as variable type.
   D. Missing type parameter on right hand side of assignment.
   E. HashMap is an interface and cannot be instantiated.
   F. Some other error.
   G. This code will successfully compile.

5. For the following questions, select the single *best* answer.

(a) Which methods can access the private members of a class? (2)

A. Only static methods of the same class.
B. Only final methods of the same class.
C. Only private methods of the same class.
D. Only methods defined in the same class.
E. Only methods defined in the same package (including those in the class itself).
F. Only methods within the same class or its children.
G. Only methods within the same class, its children, or in the same package.

(b) What is the value of the following expression? (2)

```
1 + 2 * 3 + "4" + 5
```

A. `16`
B. `18`
C. `"12345"`
D. `"1645"`
E. `"745"`
F. `"945"`
G. The value of this expression is undefined.
H. This expression would result in a compilation error.

(c) Which type could `foo` be in the following code snippet? (2)

```
Object x = foo.get(0);
```

A. `Collection` B. `Deque` C. `List` D. `Set` E. `Queue` F. `SortedSet`
G. Any of these. H. None of these.

(d) Which type could `foo` be in the following code snippet? (2)

```
boolean b = foo.isEmpty();
```

A. `Collection` B. `Deque` C. `List` D. `Set` E. `Queue` F. `SortedSet`
G. Any of these. H. None of these.

(e) Which of the following correctly does *not* declare and instantiate a map that associates String keys with Double values? (2)

A. `Map<String, Double> map = new HashMap<String, Double>();`
B. `HashMap<String, Double> map = new HashMap<String, Double>();`
C. `Map<String, Double> map = HashMap<>();`
D. `Map<String, Double> map = new TreeMap<String, Double>();`
E. `Map<String, Double> map = new HashMap<>();`
F. `Map<String, Double> map = new TreeMap<>();`
G. All of the above.
H. None of the above.

(f) Which interface would be the best choice to associate ISBN values with book objects? (2)

A. `Collection` B. `Deque` C. `List` D. `Map` E. `Set` F. `Queue`

(g) Which interface would be the best choice as a parameter of a method that can accept any collection of student usernames, without assuming anything about order or uniqueness? (2)

A. `Collection` B. `Deque` C. `List` D. `Map` E. `Set` F. `Queue`

(h) Which interface would be the best choice to hold a collection of unique book titles? (2)

A. `Collection` B. `Deque` C. `List` D. `Map` E. `Set` F. `Queue`

6. Consider the following interface.

```
public interface MyInterface {
    int handleInt(int n);
    void doStuff(String s, double x);
}
```

For each of the following classes:

- Does it correctly implement the interface?
- If it does not, what is wrong with it?

(a) (3)

```
public class MyImpl implements MyInterface {
    void doStuff(String s, double x) { }
    int handleInt(int n) { return n; }
}
```

(b) (3)

```
public class MyImpl implements MyInterface {
    public void doStuff(String a, double b) { }
    public int handleInt(int n) { return n*n; }
}
```

(c) (3)

```
public class MyImpl {
    public void doStuff(String s, double x) { }
    public void doStuff(double x) { }
    public int handleInt(int n) { return 42; }
}
```

(d) (3)

```
public class MyImpl implements MyInterface {
    public void handleInt(int n) {}
    public void doStuff(String s, double x) { }
}
```

7. Consider the following classes. What is the output of this code? (14)

```
public class Foo {
    protected String a = "AAA";
    protected int b = 12;

    public Foo() {
        System.out.println(b);
        b *= 2;
    }

    public Foo(String b) {
        this.a = b;
        this.b = b.length();
    }

    public void doStuff() {
        System.out.println(a);
        System.out.println(b);
    }

    public void doStuff(int c) {
        doStuff();
        System.out.println(c);
    }

    public void doStuff(String b) {
        System.out.println(a);
        System.out.println(b);
    }

}
```

```
public class Bar extends Foo {
    protected String c = "BBB";

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

    public void doStuff() {
        super.doStuff();
        System.out.println(c);
    }

    public void doStuff(String a) {
        System.out.println(a);
        doStuff(123);
    }


    public static void main(String[] args) {
        Foo x = new Bar("CCC");
        x.doStuff("DDD");
    }
}
```

8. Consider the following class. What is the output of this code? (10)

```
public class Baz {
    private int x = 10;
    private static int y = 2;

    public Baz(int x) {
        this.x += y;
        y = x;
    }

    public void doStuff() {
        x--;
        y++;
        System.out.println(x);
        System.out.println(y);
    }

    public static void main(String[] args) {
        Baz b1 = new Baz(30);
        b1.doStuff();

        Baz b2 = new Baz(40);
        b2.doStuff();

        b1.doStuff();
        b2.doStuff();
        b1.doStuff();
    }
}
```

9. Write a method that takes two arguments, a `Collection` of `String` objects (this should take any type of collection, not just a specific implementation) and a `String` to search for, and returns a collection containing only the strings found in the input collection that contain the search string somewhere in them.. (10)

   You may assume the arguments will not be null and that the collection argument will not contain any null values.

   Do not modify the input collection.