Do not open until I give a verbal instruction to do so

Name: ____________________________  Section: 001

NetId: ____________________________  Exam Version: A

Algorithm:

→ Collect this exam
→ Proceed to seat
→ Sign the exam
→ Read instructions below carefully
→ When I give the verbal instruction, open and begin the exam
→ Do not leave seat (unless medical emergency)
→ Finish exam
→ If you finish before 10:45 {
    → Raise your hand
    → Exam will be collected
    → Wait until 10:45 (You may use your phone)
} else {
    → Return exam
}
→ Exit

Instructions:

• You have 70 minutes.

• Write all answers on the answer sheet. If you make a mess, clearly indicate your final answer.

• Read the instructions for each section carefully! Be sure to answer all the questions.

• If you have any questions during the exam, raise your hand and I will try to get to you.

• At the end of this exam there is one blank page. Use this as your scrap paper.

• The exam is printed on both sides of the paper. Don't be the person who doesn't answer half the exam.

• There will be zero tolerance for cheating.

• At no point panic, if you get stuck, move on and return to the question later.
Section 1: Boolean (10 points)

Write your answer on the answer sheet.

01. Multiple inheritance allows a subclass to directly extend multiple classes. Java supports multiple inheritance.

02. For both static and instance variables, the variable has class scope regardless of other modifiers (ex. private)

03. A variable of a reference type doesn’t actually contain an object, it contains a reference to the object on the heap.

04. The super keyword cannot be used outside of a constructor.

05. Member variable and method declarations must include an accessibility modifier such as private or protected, without a modifier, the program will not compile.

06. When a constructor of a subclass is execute at runtime, a superclass constructor may optionally be executed by the subclass, but it is not required.

07. When a method or a variable is marked as private, it can be accessed from within the class or from within subclasses.

08. Each time you call a method, Java allocates memory called the static block to hold its variables.

09. Data hiding ensures exclusive data access to class members and protects object integrity by preventing illegal values being set in instance variables.

10. When a class contains another class as an instance variable the latter is said to be abstracted by the former.

Section 2: Multiple Choice (15 points)

Write your answer on the answer sheet.

01. Which of the following are true statements regarding the stack data structure?
   (a) It has 'last-in, last-out' behavior.
   (b) The 'peek' operation removes the top element from the stack.
   (c) The number of elements contained in a stack after some number of operations is called 'capacity'.
   (d) The 'push' operation removes the top element from the stack.
   (e) None of the above.

02. Which of the following are true statements regarding objects and classes?
   (a) An object is like the blueprint, and we can have many instances of it, which are called classes.
   (b) A class is like the blueprint, and we can have many instances of it called, which are called objects.
   (c) There is no difference between a class and an object.
   (d) The object keyword indicates we're defining an object and the class keyword indicates we're defining a class.
   (e) None of the above.

03. Which of the following are false statements regarding methods and variables?
   (a) Instance methods can access instance variables and instance methods directly.
   (b) Instance methods can access static variables and static methods directly.
   (c) Static methods can access static variables and static methods directly.
   (d) Static methods cannot access instance variables or instance methods directly.
   (e) Static methods cannot use the 'this' keyword as there is no instance for this to refer to.
   (f) None of the above.
04. Which of the following are false statements regarding inheritance?
   (a) If no superclass is explicitly declared, the compiler inserts an extends clause for java.lang.Object.
   (b) A subclass has access to all methods of a superclass, regardless of the access modifiers on that method.
   (c) Subclasses can specialize their behaviors by overriding superclass methods.
   (d) The super keyword can be used to explicitly invoke a superclass constructor from a subclass constructor.
   (e) None of the above

05. Which if the following are true statements regarding constructors?
   (a) The name of the constructor must be different than the name of the class.
   (b) The return type must be void.
   (c) The constructor is called automatically when a new operator is used to create an object.
   (d) Constructors may not be overloaded.
   (e) None of the above.

Section 3: Code Reading (15 points)
When executed would the code display any output? If so what would it display? If not, why not?
Write your answer on the answer sheet.

01.
```java
class Value {
    int value = 8;
}

class Mutilator {
    static void mutilate(int[] a, int b, Value c) {
        a[0] = 7;
        b = 7;
        c.value--;  
    }
}

public class Main {
    public static void main(String[] args) {
        int[] a = new int[] { 1, 2, 3, 4 };
        int b = 4;
        Value c = new Value();
        Mutilator.mutilate(a, b, c);
        System.out.print(a[0]);
        System.out.print(b);
        System.out.print(c.value);
    }
}```
02.
```java
class Foo {
    private static int staticVar = 0;

    public Foo() {
        ++staticVar;
    }

    public static int get() {
        return staticVar;
    }
}

public class Main {
    public static void main(String[] args) {
        Foo f1 = new Foo();
        Foo f2 = new Foo();
        Foo f3 = new Foo();
        System.out.println(Foo.get());
    }
}
```

03.
```java
public class TwoDimArray {

    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3, 4 },
            {4, 5, 6, 7 },
            {8, 9, 10, 11 },
            {12, 13, 14, 15 }
        };

        for (int i = 0; i < 4; i++) {
            System.out.print(matrix[1][i] + " ");
        }
    }
}
```
04.
public class SomeClass {
    public SomeClass(String s){
        this();
        System.out.print(s);
        System.out.print(" baz ");
    }

    public SomeClass(int x){
        this(" foo ");
        System.out.print(" bang ");
    }

    public SomeClass(){
        System.out.print(" bar ");
    }

    public static void main(String[] args) {
        SomeClass t = new SomeClass(1);
    }
}

05.
class Super {
    public String m1() {
        return "123";
    }
    public String m2() {
        return "456";
    }
}

class Sub extends Super {
    public String m1() {
        return super.m2();
    }
}

class Base {
    public static void main(String[] args) {
        Sub sub = new Sub();
        System.out.println(sub.m1() + sub.m2());
    }
}
Section 4: Programming (25 points)

Answer the programming questions on the answer sheet.

For each section A-E you only need to define what is specified. Moreover, only provide code for the things that are asked for.

Write a `Die` class that has the following properties:

A. `Die` has the following variables:

- `numSides`  An `int` that holds the number of sides the die has.
- `color`  A `String` that holds the color of dice. It should have the same value for all objects.

Define the variables as they would appear in the class. Initialize the color variable to a value of your choice (ex. red). Make sure to use the correct modifiers.

B. `Die` has the following constructors:

- **2-arg**
  Takes as parameters an `int` called `numSides` and a `String` called `color`.
  For `numSides`, forbid negative values, in that case use 6 as the value.

- **No-arg**
  Takes no arguments. Initialize the class such that you create a six-sided, purple die.

Make sure to initialize instance variables properly. Use of the `this` keyword (both of its applications).

C. `Die` has a setter for `numSides`

Implement a mutator method for the variable `numSides`. Make sure to use the conventional name for this method. Forbid negative values, in that case use 6 as the value.

D. `Die` has a getter for `color`

Implement an accessor for `color`. Make sure to use the conventional name for this method. Think about the modifiers.

E. `Die` has the following method:

- `rollAgainst`  A method that simulates the rolling of two dice.

`rollAgainst` that takes another `Die` object as a parameter, “rolls” both dice using the `roll()` method.

(roll() is provided, see answer sheet)

`rollAgainst` returns 1 if the first die's (the one the method is called on) roll is larger than the opponents (the die that is passed in) roll, returns -1 if the opponent's roll is larger or returns 0 if they are the same.
Section 5: Programming (20 points)
Answer the programming questions on the answer sheet.

The class StackOfCharacters is provided below. Implement the missing portions.

```java
public class StackOfCharacters {
    private static final int DEFAULT_CAPACITY = 16;

    private Character[] list;
    private int capacity;
    private int size = 0;

    public StackOfCharacters(int c) {
        // TODO - Implement this
    }

    public boolean empty() {
        return size == 0;
    }

    public boolean full() {
        return capacity == size;
    }

    public boolean push(Character value) {
        // TODO - Implement this method
    }

    public Character pop() {
        // TODO - Implement this method
    }

    public Character peek() {
        // TODO - Implement this method
    }

    // Some getters/accessors
    public int getSize() {
        return size;
    }

    public int getCapacity() {
        return capacity;
    }
}
```