Term 2 – Unit 9 – Week 15 Name _____

Everything You Need to Know for the AP Exam

1. **How long a minute is, depends on which side of the bathroom door you're on.**
   ~Zall's Second Law

In other words… save time!

- Do not read a problem top to bottom!
  - As crazy as it sounds, it's true. **Before** you read a long section of code **Read the Question!** No point in reading all that code if you don't know what you are looking for. Plus, they sometimes add a lot of stuff you don't need.
- Think about what they are really trying to get at.
  - Loop problems are often about WHERE a loop ends. You can often just look at the boundary cases (starting and stopping points)
  - if-else problems are testing if you understand nesting and the { } – where one if stops and the next begins. This means you can skip sections of code.

Try It!

Consider the following method:

```
public void conditionalTest (int a, int b) {
      if(( a > 0) && (b > 0)) {
            if  (a>b)
                  System.out.println("A");
            else
                  System.out.println("B");
      } else if ((b < 0) || (a < 0 ))
                  System.out.println("C");
      else
                  System.out.println("D");

}
```

Read this first

What is printed as a result of the call conditionalTest (3, -2)?

a. A
b. B
c. C
d. D
e. Nothing is printed.

Notice if you read the question first then you can skip a lot of this code. Do it right and you only read 4 lines, instead of 10!!

2. **Plug and Play**

- Sometimes the easiest way to solve a problem is to plug in some known values. The trick is picking good values.
- On loop problems try boundary cases…where the loop should stop and start. Does it do what it is supposed to?
- On array problems…make a smaller array of 3 - 4 items:

Try it! Don't forget: Read the question BEFORE you start on the code:

```
private int[] arr;
//precondition : arr.length > 0
public void mystery () {
      int s1 = 0;
      int s2 = 0;
      for (int k = 0; k < arr.length; k++) {
            int num = arr[k];
            if ((num >0) && (num % 2 == 0))
                  s1 += num;
            else if (num <0)
                  s2 += num;
      }
      System.out.println(s1);
      System.out.println(s2);
}
```

Which of the following best describes the value of s1 output by the method *mystery*?

a. The sum of all positive values in arr
b. The sum of all positive even values in arr
c. The sum of all positive odd values in arr
d. The sum of all values greater than 2 in the arr
e. The sum of all negative odd values in arr

Huh? Since the answers all have to do with even/odd & positive/negative try the following array:

```
-9    7    2    8    6    5    1
```

What answer does this give?

Also, if you read the question first, you knew to ignore all that code about s2.

Term 2 – Unit 9 – Week 15

3. **Know the Lingo**

If you don't understand the question, **how are you going to answer?**

- **Compare and contrast Classes and Objects**. How are they related?

- What does **static** do to a:
  - method?
  - variable?

- **Primitive** and **class** data:
  - Give examples of each:
    - primitive:
    - Class:
  - Describe the difference in how these are passed as parameters.

- **Constructors**
  - What are they?
  - What is constructor chaining?
  - With inheritance…what's the gotcha when using a constructor from a super class?

- **Interfaces**
  - Can they contain constructors?
  - Can they be instantiated?
  - While we are on the subject, what the heck is instantiated?
  - All methods in an interface are abstract- true or false?

- **Public** and **Private**
  - What impact do these keywords have on inheritance?

4. **Inherit the Wind!**

- **Fact**: Inheritance is a major topic on the exam
- For Example, consider a set of classes with Fruit, Apple, Orange and Mango.
  - An Apple ***has a*** seed, so you would not use inheritance.
  - An Apple ***is a*** Fruit, so you would use inheritance
  - Fruit is the parent, Apple is the child
- Try it!
  - What does Apple inherit from Fruit?
  - How would Apple access the constructor in Fruit?
  - How does public/private info impact Apple and Fruit?

```
public class Vehicle {
      private int numWheels;

      public Vehicle (int n) {
            numWheels = n;
      }
}
```

Write a class, Scooter, that extends Vehicle. The constructor should set the number of wheels to 2.

Term 2 – Unit 9 – Week 15

5. **History Repeats Itself**

- These kinds of problems are about keeping track of what variables equal when.
- MAKE A CHART!! Writing things down makes them easier to remember.
- **Loops**

```
int num1 = 0;
int num2 = 3;

while ((num2 != 0) && ((num1/num2) >= 0)) {
      num1 = num1 + 2;
      num2 = num2 - 1;
}
```

What are the values of num1 and num2 after the while loop completes its execution?

a. num1 = 0, num2 = 3
b. num1 = 8, num2 = -1
c. num1 = 4, num2 = 1
d. num1 = 6, num2 = 0
e. The loop will never complete its execution because a division by zero will generate an ArithmeticException

- **Recursion**: a function that calls itself

```
private static void recur (int n) {
      if (n != 0) {
            recur (n-2);
            System.out.print(n + " ");
      }
}
```

What will be printed by recur(7) ?

a. -1 1 3 5 7
b. 1 3 5 7
c. 7 5 3 1
d. Many numbers will be printed because of infinite recursion.
e. No numbers will be printed because of infinite recursion

There are always a few recursion problems on the Multiple Choice section.

6. **A few picky details:**

There are a few things **guaranteed** to be in there:

- De Morgan's law
    - This lets you simplify Boolean Expressions
    - Basically, it works the same way as algebraic distribution :
        - ! ((a < b) || (c == a)) → (a >= b) && (c != a)
    - Try it!
        - ! ((y != x) || ((x ==3) && (y < 0)))

- When Chuck Norris does division, there are no remainders.
    - Remember that :
        - int x = 9/2;
        - System.out.println(x); ← displays *4* NOT 4.5

- Know your limit: Integer.MIN_VALUE and Integer.MAX_VALUE
    - Integer.MIN_VALUE = smallest possible integer
    - Integer.MAX_VALUE = largest possible integer

Use the array: int list [] ;

Complete the following method to return the smallest value less than n in the array list.

```
public int smallerThan( int n) {
/*
Return the smallest value in the array less
than n, if no value is smaller return
MAX_VALUE
*/
```

7. **Interfaces** (again)

Lets review: What *is* an interface?

- Picky detail: All methods default to public (unless coded otherwise)

- Example: Comparable:
  - One of the Java standard interfaces
  - Holds one abstract method: compareTo ()
  - Any class that implements the Comparable interface must hold this method.
  - You have used this in the String class.
    - obj1.compareTo(obj2) → What will this return? (see the Quick reference)

- Example: **List interfaces**
  - List: look it up (it's in the Quick Reference). What does it do?
  - What class have we used that implements List?

Term 2 – Unit 9 – Week 15

8. **No problemo boss!**

When Java encounters an error it ***throws*** an ***exception***.

- Types of exceptions: What do each of the following do?
    - ArithmeticException
    - NullPointerExcetion
    - ArrayIndexOutOfBounds

- 2-D Arrays: Drawing inside the lines.
    - Remember 2D Arrays are row-major. That means we look at the rows first, then the columns.

Use the 2D array:

int test [] [] → No, I'm not telling you how big the array is!

Print the sum of only the even values stored in test.

Term 2 – Unit 9 – Week 15

9. **And last but not least…. covering all the bases**

- You should be able to convert between:
  - decimal base _____
  - binary base _____
  - octal base _____
- Fill in the table!

decimal	binary	octal
37		
		230
	1011	

