Midterm #2 Sample Questions

Answer 4 of the first 5 questions (#'s 1 through 5)

1. **Short answer questions #1** (10 points)

   (a) What class or classes can access and use **`public`** member variables and methods? What class or classes can access and use **`private`** member variables and methods? (2 points)

   (b) **Why** declare a **member variable `private`**? (2 points)

   (c) What technique or mechanism can you use to **give controlled access to a private member variable**? That is, if you have a private member variable in your class, how can a programmer using the class read the private member variable and/or change it? (2 points)

   (d) Name 3 ways that declaring and using a **constructor** is **different** from declaring or using a **regular method**. What is a constructor used for? (2 points)

   (e) What is **`null`**? When does it come up / when do we see it produced? (2 points)

2. **Short answer questions #2** (10 points)

(a) Describe the **rules** governing what you can do within a **static method** – specifically, its access to instance or static variables, and the ability to call instance or static methods. Briefly describe why. (2 points)

(b) Which is faster, **binary search** or **linear search**? Why…and under what conditions? Are there any assumptions that are made about the Array being searched? (2 points)

(c) What's the difference between a **class** and an **object**? (2 points)

(d) What is the **`StringBuilder`** class? How is it different from using a regular **`String`**? (2 points)

(e) What does the keyword, **`this`**, signify? When should it be used? (2 points)

3. Read the code in the left column. Write the output of the code in the right column. **Compile error**, **nothing** and the ***actual* output** are **all valid answers**. (10 points)

Code (syso is short for System.out.println())	**Output (compile error, nothing and actual output are valid)**

`// imagine you have a stack of ints that`
`// supports push, pop, peek and empty. What is`
`// the output of the code below?`
 	
`syso(stack.empty());`
`stack.push(4);`
`stack.push(2);`
`stack.push(12);`
`stack.push(5);`
`syso(stack.peek());`
`syso(stack.pop());`
`stack.pop();`
`stack.pop();`
`syso(stack.peek());`	

`public class Foo {`
`  public static void main(String[] args) {`
`  Foo f1 = new Foo(4);`
`  Foo f2 = new Foo();`
 	
`  System.out.println(f1.n);`
`  System.out.println(f2.n);`
`  f1 = f2;`
`  addFive(f1);`
`  System.out.println(f1.n);`
`  System.out.println(f2.n);`
`  }`
 	
`  public static void addFive(Foo f) {`
`  f.n += 5;`
`  }`
 	
`  int n;`
 	
`  Foo() {}`
 	
`  public Foo(int n) {`
`  this.n = n;`
`  }`
`}`	

4. Circle the errors below and briefly describe why there are errors. (5 points)

Find 5 of the 6 errors in the code below. They can be compile time, run time or logical errors.

```
import java.lang.reflect.Array;
import java.util.Arrays;

public class Stacky {
      private int[] elements;
      private int size;

      public void Stacky(int capacity) {

            elements = new double[capacity];

            // this controls the size of the stack... so it should
            // *always* accurately determine where the end of the stack is
            size = 0;
      }

      public void push(int x) {
            if (size >= elements.length) {
                  int[] newElements = new int[size * 2];
                  for(int i = 0; i < newElements.length; i++) {
                        elements[i] = newElements[i];
                  }
                  elements = newElements;
            }
            elements[size - 1] = x;
      }

      public int pop() {
            size += 1;
            return elements[size];
      }

      public static int getSize() {
            return this.size;
      }
}
```

(a) _____

_____

(b) _____

_____

(c) _____

_____

(d) _____

_____

(e) _____

_____

5. True or False: (10 points)

(a) _____ You must declare a visibility modifier (`public, private or protected`) before member variables and methods. Without a modifier, you will get a compiler error.

(b) _____ A static method can be invoked both from the class name as well as from an instance:

```
ClassName.myStaticMethod();
ClassName instance = new ClassName();
instance.myStaticMethod();
```

(c) _____ In the following code, the variables `greeting1` and `greeting2` both have a reference that points to the same copy of `"I am a string"` in memory.

```
Greeting1 = "I am a string";
Greeting2 = new String("I am a string");
```

(d) _____ In a program that uses `Processing`, the `setup()` method is called exactly once, and the `draw()` method is called repeatedly after the setup method.

(e) _____ Is the output of the code below `true` or `false`?

```
String s1 = "abc";
String s2 = "adb";
System.out.println(s1.compareTo(s2) < 0);
```

(f) _____ If a local int variable in a method is not explicitly initialized within that method, it receives a default value of 0.

(g) _____ Binary search requires an Array to be presorted.

(h) _____ Assuming that the `Foo` class is defined and contains a public member variable, `n`, both variables, `foo1` and `foo2` are instances of the `Foo` class, and the constructor takes the argument passed in and sets the member variable, n... the code below will print out:

```
// 25
// 12

Foo foo1 = new Foo(12);
Foo foo2 = foo1;
foo1.n = 25;

System.out.println(foo1.n);
System.out.println(foo2.n);
```

(i) _____ If a single constructor is defined in a class, it will have that constructor, as well as the default, no argument constructor available for use.

(j) _____ `(new String("hello")) == (new String("hello"))`

Answer 3 of the last 4 questions (#'s 6 through 9)

6. **Two Dimensional Arrays** (20 points) - Write the following two methods that work on 2D Arrays...

(a)
```
// takes a two dimensional Array, and reverses the columns in the original
// Array passed in (does not return a new Array!)
public static void reverseColumnsInPlace(int[][] arr)
```

Example:

Initial declaration of m	Calling method with m	m is now...

`int[][] m = {`
`  {1, 2, 3, 4},`
`  {2, 3, 4, 5},`
`  {3, 4, 5, 6}`
`};`
	`reverseColumnsInPlace(m);`	`// m looks like:`
		`{3, 4, 5, 6},`
		`{2, 3, 4, 5},`
		`{1, 2, 3, 4}`

(b)
```
// give back the row with the largest sum; it's ok to return a reference
// rather than a new Array
System.out.println(Arrays.toString(getRowWithMaxSum(m)));
```

Example:

Initial declaration of m	Calling method with m	Value returned is

`int[][] m = {`
`  {1, 2, 3, 4},`
`  {2, 3, 4, 5},`
`  {3, 4, 5, 6}`
`};`
	`getRowWithMaxSum(m);`	`// an Array is`
		`// returned`
		`{3, 4, 5, 6},`

7. **Sorting Magical Sticks** (20 points)

(a) Finish the missing implementation of bubble sort method below
(b) Create a class MagicalStick, based on the usage of the main method below

```
public class SortMagicalSticks {
  public static void main(String[] args) {
    MagicalStick[] magicStuff = new MagicalStick[4];
    magicStuff[0] = new MagicalStick(7, "wiggle ears");
    magicStuff[1] = new MagicalStick(10, "levitate");
    magicStuff[2] = new MagicalStick(8, "turn into panda");
    magicStuff[3] = new MagicalStick(9, "lazer beamz");
    bubbleSort(magicStuff);
    System.out.println("Sorted by length...");
    for(MagicalStick s: magicStuff) {
      System.out.println(s.getLength() + " " + s.getPower());
    }
  }
  public static void bubbleSort(MagicalStick[] sticks) {
    // implement bubblesort according to the pseudocode below:
    // keep track of swaps
    // while no swaps have happened
    //   assume no swaps
    //   for every index, up to second to last
    //     compare element at index with next element to sort by length
    //     if element at index is greater
    //       then swap elements
    //       keep track of the fact that a swap happened
}
}
// implement that class that would be in MagicalStick.java
```

8. **My Very Own String Class** (20 points) **-** Create your own version of a String class called MyString. Implement the following methods and constructors (if necessary, see the unicode character chart on the last page):

```
// the constructor, initializes object with the char[] Array supplied
MyString(char[] chars)

// returns the character at the specified index
public char charAt(int index)

// give back the number of characters
public int length()

// return a new MyString object that made from a substring of the original,
// starting at the index, begin (inclusive), going up to, but not including
// the index, end (exclusive)
public MyString substring(int begin, int end)

// gives back a new MyString object that is all lowercase (only letters are
// affected)... when adding an int to a char, you'll have to cast the result
// to char if you're assigning the result to a char variable
public MyString toLowerCase()

//Example Usage
char[] c = {'v', 'o', 'w', 'e', 'l'};
MyString s = new MyString(c);
System.out.println(s.charAt(0)); // v
System.out.println(s.length());  // 5
System.out.println(s.substring(1, 4)); // MyString object with 'o' 'w' 'e'
```

9. **Triangle Maker** (20 points)

(a) Write a program that asks for the coordinates of 3 points, and prints out:
   i. the perimeter of the resulting triangle (the sum of all sides)
   ii. whether or not the resulting triangle is isosceles (two sides are equal)
   iii. example interaction below:

```
Enter point 1 as x,y
> 0,0
Enter point 2 as x,y
> 4,10
Enter point 3 as x,y
> 8,0
The triangle's perimeter is 29.540659228538015
The triangle is isosceles.
```

(b) Requirements
   i. The input must be in the format x,y (hint: there's a method that helps you *extract* x and y)
   ii. Expect that the inputs are integers (hint: `Integer.parseInt` may come in handy)
   iii. You must create 3 classes:
   iv. **`Triangle`** - A class that represents a triangle; it **should contain 3 Point objects in an Array**, and the class should be fully encapsulated (points should not be available by direct data field access)... **as well as a couple of methods for finding the perimeter and determining if a triangle is isosceles**
   v. **`Point`** - A class that represents a point in a 2d plane; it's ok to make the x and y value accessible directly. A **method should exist in this object that calculates distance**
   vi. **`TriangleMaker`** - the entry point into your program, it should be responsible for all input and output

Char	Dec	Char	Dec	Char	Dec	Char	Dec
(nul)	0	(sp)	32	@	64	`	96
(soh)	1	!	33	A	65	a	97
(stx)	2	"	34	B	66	b	98
(etx)	3	#	35	C	67	c	99
(eot)	4	$	36	D	68	d	100
(enq)	5	%	37	E	69	e	101
(ack)	6	&	38	F	70	f	102
(bel)	7	'	39	G	71	g	103
(bs)	8	(	40	H	72	h	104
(ht)	9	)	41	I	73	i	105
(nl)	10	*	42	J	74	j	106
(vt)	11	+	43	K	75	k	107
(np)	12	,	44	L	76	l	108
(cr)	13	-	45	M	77	m	109
(so)	14	.	46	N	78	n	110
(si)	15	/	47	O	79	o	111
(dle)	16	0	48	P	80	p	112
(dc1)	17	1	49	Q	81	q	113
(dc2)	18	2	50	R	82	r	114
(dc3)	19	3	51	S	83	s	115
(dc4)	20	4	52	T	84	t	116
(nak)	21	5	53	U	85	u	117
(syn)	22	6	54	V	86	v	118
(etb)	23	7	55	W	87	w	119
(can)	24	8	56	X	88	x	120
(em)	25	9	57	Y	89	y	121
(sub)	26	:	58	Z	90	z	122
(esc)	27	;	59	[	91	{	123
(fs)	28	<	60	\	92	|	124
(gs)	29	=	61	]	93	}	125
(rs)	30	>	62	^	94	~	126
(us)	31	?	63	_	95	(del)	127