Sum12 cs401 MIDTERM EXAM 6/27/2012

YOUR NAME:

YOUR PITT ID:

Q#	PTS	RUBRIK	**Your Score**
1	3	all/nothing	
2	2	all/nothing	
3	2	all/nothing	
4	3	all/nothing	
5	6	3 pts ea.	
6	10	all/nothing	
7	8	2 pts ea.	
8	5	all/nothing	
9	9	3 pts ea.	
10	2	all/nothing	
11	2	all/nothing	
12	5	all/nothing	
13	5	all/nothing	
14	3	1pt ea.	
15	5	all/nothing	
16	3	all/nothing	
17	8	all/nothing	
18	6	all/nothing	
19	9	6 pts a/b 3pts c	
20	4	2 pts ea.	
	100		

1) Circle any data type that is NOT a primitve:

(a) int (b) char (c) boolean

(d) String (e) Integer (f) Scanner

2) Circle any set of operators that is capable of short circuiting:

(a) + - / * %

(b) || &&

(c) < > <= >= !=

(d) () [] .

3) When I issue the following command from the command line, which method am I implicitly calling inside Program1?

C:\> java Program1

YOUR ANSWER: main

4) When I issue the following command from the command line, where do the Strings "Hi", "my", "name", "is", and "Tim" get stored inside Program1?

C:\> java Program1 Hi my name is Tim

YOUR ANSWER: In the args[] array

5) Here are two methods from the Arrays library.
Both methods will sort an array of objects.
Notice that both methods have the same name.

```
static void sort(Object[] a)
static void sort(Object[] a, int fromIndex, int toIndex)
```

a) How does the compiler tell them apart?

Parameter list

b) Which one assumes the array is full?

The first one

6) Here is a method that is supposed to fill an array with random numbers 1..100 inclusive and is supposed to ensure that there are no duplicates or gaps in the array.

```
// ASSUME: arr has been dimensioned to something like 20 elements or whatever
// No values have been put into the array yet.
// Parameter r is initialized and ready for calls to nextInt().

private static void randomizeArray( int[] arr, Random r )
{
    for( int i=0 ; i< arr.lrngth ; ++i
    {
        int n = r.nextInt(100) + 1; //WARNING it is possible to get a repeat value
        if ( ! linearSearch( arr,i,n ) ) // That's why we search before putting into array
            arr[i] = n;
    }
}
//ASSUME: This search method IS CORRECTLY WRITTEN
private static boolean linearSearch( int[] arr, int count,int target )
{
    for (int i=0 ; i<count; ++i )
        if (arr[i]==target) return true;
    return false;
}
```

a) What is wrong/dangerous/undisciplined about the randomizeArray method?

When dupes are skipped the cell is not initialized

7) Here is a program that appears to work right but it is in fact written very, very wrong.

```
public class Wrong
{
    public static void main( String[] args )
    {
        String[] database= { "foo", "bar", "baz", "fizz", "pop", "gorp"/ "blatz"};
        String[] targets= { "plop", "plaup", "fizz", "fiz", "gorp", "bar" };
        for (int i=O ; i<targets.length; ++ i )
            if ( contains( database, targets[i) ) ) //i.e. if the database array contains target[i)
                System.out.println( "database contains: "+ targets[i] };
            else
            System.out.println( "database does NOT contain: " + targets[i] );
    }
    private static boolean contains( String[] db, String target)
    {
        for (int i=O ; i<db.length ; ++i)
            if ( db[i] ==target)
        return false; // if we make it here it was not found
    }
    return true;

}
```

NOTICE that the output is actually "correct", but the code is WRONG!!

== is the wrong operator to compare strings

```
C;\Users\tlh\Desktop>javac Wrong.java
C:\Users\tlh\Desktop>java Wrong
```

```
database does NOT contain: plop
database does NOT contain: plaup
database contains: fizz
database does NOT contain: fiz
database contains: gorp
database contains: bar
C:\Users\tlh\Desktop >
```

a) Which method contains the bug: main() or contains()?

main

b) What exactly is that bug?

Skipping a dupe misses that cell

c) Why does it produce the correct output anyway?

Strings are pooled

d) What could you change about how the database array was initialized to cause the bug to manifest itself?

new String("foo")

8) What is the output of this program ?

```
public static void main( String[] args)
{
        int[] arr = new int[IO];
        for (int i=O ; i<arr.length ; ++i)
                arr[i] = i;
        relnitArr( arr );
        for (int i=O ; i<arr.length ; ++i)
                System.out.print( arr[i] + " " );
        System.out.println();
}
private static void relnitArr( int [] arr )
{
        arr= new int[ 5 ];
        for (int i=O ; i<arr.length ; ++i)
                arr[i] = i*2;
}
```

0 1 2 3 4 5 6 7 8 9

9) Look at this program:

```
1:  public class Prob9
2:  {
3:    public static void main( String[] args )
4:    {
5:          int[] arr = null;
6:          initArr( arr ); // dimension array & put values in
7:          for (int i=0 ; i<arr.length ; ++i)
8:                System.out.print( arr[i] + " " );
9:          System.out.println();
10:   }
11:   private static void initArr( int [] arr )
12:   {
13:         arr= new int[ 5 ];
14:         for (int i=O ; i<arr.length ; ++i)
15:               arr[i] = i*2;
16:   }
17: }
```

Tell me EXACTLY where it crashes:

(a) On what line number? 7

(b) On what piece of syntax? .length

(c) What is the nature/cause of the crash?
Deref a null pointer

10) Look at this little chunk of code:

```
1: String s = "Hello World";
2: s.toUpperCase();
3: System.out.println( s ); What is the output?
```

Hello World

11) What does it mean that an object is "immutable" ?

Once initialized cant be modified

12) Here is a method to swap two values:

```
private static void swap( int x,int y)
{
        int tmp=x; x=y; y=tmp;
}
```

(a) Why doesn't it work?

x and y are copies.

13) When I pass an array's reference into a method - Why does the compiler hand off a copy of the address in that reference to the method instead of handing off a copy of all the elements in the array?
(the design decision reasoning) way too memory expensive

14) What values does Java by default put into the follwing data types respectively:

(a) int 0

(b) Boolean false

(c) String null

15) When I pass a String's name into a method, does the compiler hand off a copy of the String or does it just hand off a copy of the String's address (reference)?

Just a copy of the reference

```
private static void foo()
{
        int x=5;
        System.out.println( x );
}
```

What happens to the variable x when foo returns to where it was called from ?

It is destroyed when that methods frame is popped off the stack

17) Look at this code segment:

```
1: int[] arr = new int[5];
2: for (int i=0 ; i<arr.length ; ++i )
3:     arr[i] = i*10;
4: arr = new int[10];
```

What happens to the original chunk of 5 ints created by the first line of code once line 4 is finished executing ?

that chunk is orphaned and will get collected by the garb collector

18)

```
String[] words = new String[3];
words[0]="foo";
words[1]="bar";
words[2]="baz";
```

Circle the diagram that most *ACCURATELY* describes the array and the String data.
Hint: Are the Strings contained INSIDE the array or OUTSIDE the array elements?

```
                0      1      2
words: [-]---->["foo"]["bar" ]["baz"]
```

OR

```
words: [-]----> 0 [-]----->"foo"
                1 [-]----->"bar"
                2 [-]----->"baz"
```

19) If you wanted to search a SORTED array of 100,000,000 ints, which search algorithm would be more efficient?

(a) linearSearch
(b) binarySearch

(c) Why is one better than the other for a huge SORTED array?

Does not have to compare against all the elements

20) Name 2 advantages of an Arraylist over a plain array?
Self resizing & mantains own count