Name _____ Section Number _____

CS110 Exam #2 *** PLEASE TURN OFF ALL CELLPHONES *** Practice
Sections All Bob Wilson

You may only use your crib sheet (one HANDWRITTEN 8-1/2 x 11 page both sides). You will have all 90 minutes until the start of the next class period. Spend only about one minute per point on each question to complete the exam on time.

1. Operations on String Objects and Characters (20 Points)

Given the following local variable declarations:

```
String s = new String("abc");
String a = "Did Hannah see bees? Hannah did. ";
String t;
```

What is the value of the following expressions (or ERROR with reason why)?

```
s.length()                          // a. _____

t.length()                          // b. _____

a.charAt(5)                         // c. _____

s.toUpperCase()                     // d. _____

s+2                                 // e. _____

a.indexOf('H')                      // f. _____

"Tomorrow".lastIndexOf('o')         // g. _____

"Tomorrow".substring(2,4)           // h. _____

s.substring(1,3).equals("bc")       // i. _____

(s.length() + s).startsWith("a")    // j. _____
```

2. Operations with Packages, Classes and Objects (20 Points)

Explain each line of code in the left column with a number from the right column:

a. ___ `import java.util.Scanner;`

b. ___ `Scanner s = new Scanner(System.in);`

c. ___ `int i = s.nextInt();`

d. ___ `public static final double PI = …;`

e. ___ `public static void main(String[] args)`

f. ___ `int n = Integer.parseInt(string);`

g. ___ `n++;`

h. ___ `Scanner scan = s;`

i. ___ `scan = null;`

j. ___ `public String toString()`

1. Defines a class constant or variable
2. Defines an instance constant or variable
3. Defines a class method
4. Defines an instance method
5. Instantiates an object / saves reference
6. Creates an alias of an object reference
7. Deletes an object reference
8. Uses a class constant or variable
9. Uses an instance constant or variable
10. Uses a class method
11. Uses an instance method
12. Uses a class from a package
13. Doesn't involve any package, class, or object

3. Arrays of Objects and ArrayList class (20 Points)

```
import java.util.ArrayList;

public class Arrays
{
  public static void main(String [] args)
  {
    // Part a.
    int [] nums = {3, 6, 4, 2, 1, 5};
    String [] strings = new String [nums.length];

    for (int i = 0; i < nums.length ; i++)
        strings[i] = "#" + nums[i];
                                                  // what prints?
    System.out.println( strings.length);          //_________
    System.out.println(strings[3]);               //_________
    System.out.println( strings[3].length() );    //__________
    System.out.println(strings[1]  + strings[5]); //__________
    System.out.println(strings[1]. charAt(0));    //__________


    // Part b.
    ArrayList<String>  myList = new ArrayList<String>();
    boolean status;           // for return value when needed
    String removed;           // for return value when needed

    // write code to add "foobar" and "huh" to the end of myList

    ____________________________________________________

    ____________________________________________________

    // write a line of code to add "never" at beginning of myList

    ____________________________________________________

    // write a line of code to remove the item at index 2 from myList

    ____________________________________________________

    // print the contents of myList
    for (String s : myList)
       System.out.println(s);
  }                           // what prints?
}
```

4. Writing Classes (20 Points)

Write a class Man with a private String Attribute “name” and a private int attribute “age”. Include a constructor method that sets the name and the age based on arguments passed. Include accessor (“getter”) and mutator (“Setter”) methods for both attributes.
You do NOT need to write any main method for this class and should not.

5. Writing JUnit Test Cases (20 Points)

Write a JUnit test case named TestMan for the class Man you wrote in the last problem. In the test case method named testConstructor, instantiate one Man object and verify that each accessor ("getter") method returns the value that was passed to the constructor.

Answer Key:

1. Operations on Strings

a. 3
b. ERROR – null pointer exception
c. ‘a’
d. “ABC”
e. “abc2”
f. 4
g. 6
h. “mo”
i. true
j. false

2. Operations with Packages, Classes, and Objects

a. 12
b. 5
c. 11
d. 1
e. 3
f. 10
g. 13
h. 6
i. 7
j. 4

3. Arrays of Objects and ArrayList class

```
import java.util.ArrayList;

public class Arrays
{
  public static void main(String [] args)
  {
    // Part a.
    int [] nums = {3, 6, 4, 2, 1, 5};
    String [] strings = new String [nums.length];

    for (int i = 0; i < nums.length  ; i++)
        strings[i] = "#" + nums[i];
                                                  // what prints?
    System.out. println(strings .length);         //__6________
    System.out.printl n(strings[3]);              //__#2_______
    System.out.println( strings[3].length() );    //__2________
    System.out.println(strings[1]  + strings[5]); //__#6#5_____
    System.out.println(strings [1].charAt(0));    //__#________

    // Part b.
    ArrayList<String>  myList = new ArrayList<String>();
    boolean status;           // for return value when needed
    String removed;           // for return value when needed

    // write code to add "foobar" and "huh" to the end of myList

    status = myList.add("foobar");
    status = myList.add("huh");

    // write a line of code to add "never" at beginning of myList

    myList.add(0 , "never");

    // write code to remove the item at index 2 from myList

    removed = myList.remove(2);

    // print the contents of myList
    for (String s : myList)
       System.out.println(s);
  }                             // what prints
}                               never
                                foobar
```

4. Writing Classes

```
public class Man
{
     private String name;
     private int age;

     public Man (String name, int age)
     {
          this.name = name;
          this.age = age;
     }

     public void setName(String  name)
     {
          this.name = name;
     }

     public void setAge(int  age)
     {
          this.age = age;
     }

     public String getName()
     {
          return this.name;    // this. is optional here
     }
     public int getAge()
     {
          return this.age;     // this. is optional here
     }
}
```

5. Writing JUnit Test Cases

```
import junit.framework.TestCase;

public class TestMan extends TestCase
{
     public void testConstructor  ()
     {
          Man cut = new Man(“John  Smith”, 45);
          assertEquals(“John  Smith”, cut.getName());
          assertEquals(45,  cut.getAge());
     }
}
```