Spring 2019 CMP 326 Final Exam V1

124 points total

Name:______________________________ Date:___________

(24 points / 2 points each) **Trace the code and write the result (Write ERROR if there is an ERROR)**

	Code	**Method Call**	**Results**

1	`public static int q1(int a, int b) {`
	`    if (a == 0) {`
	`        return b;`
	`    } else if (a < 0) {`
	`        return b + 3;`
	`    } else {`
	`        return q1(a - b, b * 2);`
	`    }`
	`}`
		q1(4, 2);	

		q1(6, 2);	
		q1(14, 6);	
		q1(12, 4);	

2	`public static void q2(boolean a,`
	`                      boolean b,`
	`                      boolean c) {`
	`   if (a && b && !c) {`
	`      System.out.println("right");`
	`   } else if (a && c) {`
	`      System.out.println("high");`
	`   } else if (b && c) {`
	`      System.out.println("low");`
	`   } else if ((a || !b) && c) {`
	`      System.out.println("wrong");`
	`   } else {`
	`      System.out.println("in");`
	`   }`
	`}`
		q2(true, false, true);	

		q2(false, false, true);	
		q2(false, true, true);	
		q2(true, true, false);	

3	`public static void q3(char[] chars,`
	`                      int i,`
	`                      int j,`
	`                      char c) {`
	`   int index = (i * j) % 6;`
	`   try {`
	`      chars[index] = c;`
	`      System.out.println("Success!” + c);`
	`   } catch (IndexOutOfBoundsException e) {`
	`      System.out.println("No Good!");`
	`   } catch (NullPointerException e) {`
	`      System.out.println("Really Bad!");`
	`   }`
	`}`
		char[] a = {'a', 'e', 'i', 'o', 'u'};
		q3(a, 2, 5, ‘c’);	

		char[] a = {'a', 'b', ‘c', 'd'};
		q3(a, 7, 5, ‘o’);	

		char[] a = {'x', ‘y', ‘z'};
		q3(a, 8, 4, ‘w’);	

		char[] a = null;
		q3(a, 8, 4, ‘w’)	

4 **(25 points / 5 points each)** Create the Java class **Animal** that has the following **private** attributes:

Attribute	name : String	numLegs : int	gender : char

a. Create a **default constructor** that initializes all 3 attributes to the following values

Attribute	name	numLegs	gender
Default Value	“unknown”	0	‘u’

b. Create an **overloaded constructor** that takes in values for all 3 attributes and assigns them.

c. Create **getter** and **setter** methods for all 3 attributes.

Getters	Setters

d. Create the **equals** method. Two **Animal** objects are equal when their **numLegs** and **gender** are the same. (Disregard the names)

e. Create the **toString** method so that it returns a well formatted String including all attributes.

5 (20 points / 5 points each) Create the Java class **Giraffe** that inherits from the **Animal** class. The **Giraffe** class has the following **private** attributes:

Attribute	neckLength : int	height : double	foods : String[]

a. Create a **default constructor** that calls the parent constructor and initializes all 3 attributes to the following values

Attribute	neckLength	height	foods
Default Value	0	0.0	Length of 3

b. Create an **overloaded constructor** that takes in values for all 3 attributes as well as the attributes of the parent class' constructor and pass them to the parent constructor or assign them accordingly.

c. Create a **helper method** that determines the equality of two String arrays and returns a boolean, by comparing the value at each index location. Return true if all elements of the arrays matches, return false if there is any mismatch. (Note: this method will be used by the equals method in part e)

```
private boolean doArraysMatch(String[] a, String[] b) {







}
```

d. Create a **helper method** that returns the array of foods as a comma separated String.

```
private String getFoodsAsString() {







}
```

e. Create the **equals** method. Remember to include the super class's **equals** method. Two Giraffe objects are equal when all their attributes are the same. (Note: be sure to use the helper method from part c)

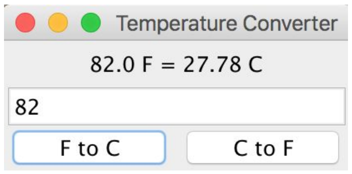
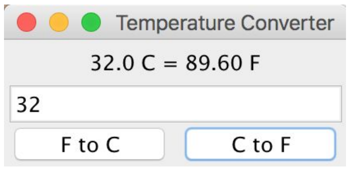
f. Create the **toString** method so that it returns a String including all attributes from both the parent and child classes. (Note: be sure to use the helper method from part d)

6 (10 points) Remember the recursive Searching algorithm BinarySearch. Write a recursive method to search for a target character in the array and return the index location if found or -1 if it is not found.

```
public static int binarySearch(char target,  char[] theValues, int firstIndex,  int lastIndex) {

}
```

7 **(35 points / 7 points each)** Write the code to create a GUI based class **TemperatureConverter** which inherits from **JFrame** and implements the **ActionListerner** interface.

Example: Clicked "F to C" button	Example: Clicked "C to F" button
	

a. Create the **constructor** so that the JButtons have listeners attached, and all components are added to the appropriate JPanel.

b. Create the **getDoubleFromTextField** method that takes in a JTextField and returns the value it contained as a double it should handle exceptions appropriately within the method.

```
public double getDoubleFromTextField(JTextField jtf) throws Exception {






}
```

c. Create the method convertTemp that takes in a temperature as a double, and the unit as a char that we are converting from. It should return the converted temperature as a formatted String (`"%.2f", tempF + " F")` or `("%.2f", tempC + " C")`

```
Example:     convertTemp(100.0, 'C')  returns      "212.00 F"
             convertTemp(87.0, 'F')   returns      "30.56 C"
```

The conversion formulas are:

F = (C * 9.0/5.0) + 32.0
C = (F - 32.0) * 5.0/9.0

```
public String convertTemp(double temperature, char unit) {






}
```

d. Create the **actionPerformed** method so that it determines which button was clicked and updates the lblOut with the conversion sentence shown in the example.

e. Create a method addConversionToFile that appends a conversion to an output file named **temperatures.txt** for the examples given in 7c, the file would have:

```
100.0 C = 212.00 F
87.0 F = 30.56 C
```

```
public void addConversionToFile(String conversion) {









}
```