Java

Q1: What is the output? Choose one. (10 pts)

```
class TestRunner {
  static public int succeeded;
  static public int failed;

  public TestRunner() {
  }

  public TestRunner(int s, int f) {
    succeeded=s; failed=f;
  }

}

class TestStatistics {
  public static void main(String[] args) {
    TestRunner tr = new TestRunner(1, 99);
    TestRunner.succeeded = 99;
    System.out.print(new TestRunner());
  }
}
```

A. **1 1**

B. **1 99**

C. **99 1**

D. **99 99**

When the class is first loaded, the values of `succeeded` and `failed` will be given default initialization to zero values (the same default that is applied to all numeric fields, whether they are static or instance).
Next, the `main` method creates an instance of `TestRunner`, and the constructor overwrites the values with constructor argument values, which are `1` and `99`.
On the second line of the `main` method's body, there's an explicit assignment to `TestRunner.succeeded` with the value `99`. This writes to the single variable called `succeeded`, so at this point, both `succeeded` and `failed` contain the value `99`.

The final step in the `main` method is to invoke the zero-argument constructor to create a new `TestRunner`. This does not modify the values of anything, so the two static fields both still contain `99`. Right after initialization of the object, the `toString` method is invoked implicitly, and the result of that is printed. Because the values of the static variables are both `99`, the resulting output is the output shown in option D. Therefore option D is correct, and options A, B, and C are incorrect.

Q2: Please explain what are the use of the keywords *Private, this, static, final* in Java Programming language. (10 pts)

private is a Java keyword which declares a member's access as private. That is, the member is only visible within the class, not from any other class (including subclasses). The visibility of private members extends to nested classes.

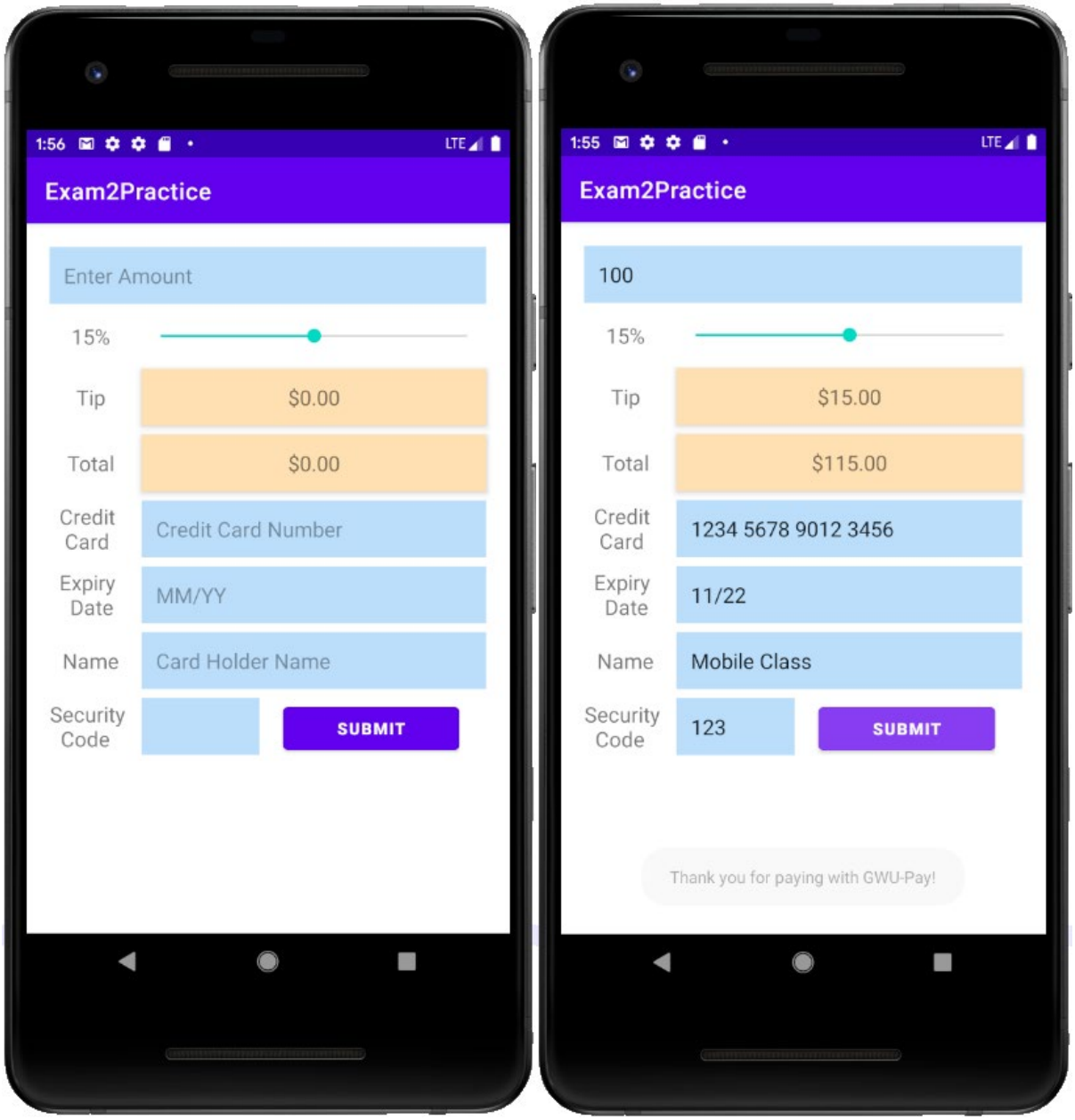
The this keyword refers to the current object in a method or constructor. The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name.

When a member is declared static, it can be accessed before any objects of its class are created, and without reference to any object.

When a variable is declared with final keyword, its value can’t be modified, essentially, a constant. This also means that you must initialize a final variable.

App

Tip Calculator is one of the favorite apps among students. We plan to further develop this app to have the functionality of paying the bill with a credit card. As you may find, we need to add new fields to accept the user inputs for credit card number, credit card expiry date, name on credit card, security code, and finally a submit button to confirm the payment. On the left side, you find the design of the app without any user inputs. On the right side, you observe the possible inputs from the user.



Please complete the app following the requirements below. You may start from the tip calculate example we used in the lecture or in the assignment. You can also find the template code of the original tip calculator on Blackboard.

Requirements:

1. Add labels, edit text fields and button in the grid layout. Please make your design close to the left figure as much as possible. You will be graded based on design, arrangement, margin. (20 pts)
2. Add OnFocusChangeListener on credit card edit text field (2nd blue region). After the credit card numbers are given, you need to add a space character for every 4 digits. (20 pts)
3. Add OnFocusChangeListener on expiry date edit text field (3rd blue region). After the expiry date is given, you need to add slash between month and year. (20 pts)
4. When the submit button is clicked, a Toast will pop up at the bottom of the screen. It shows "Thank you for paying with GWU-Pay!" (20 pts)

Hint: OnFocusChangeListener is an interface definition for a callback to be invoked when the focus state of a view changed.

Usage:

```
((EditText)findViewById(R.id.youredittext)).setOnFocusChangeListener(new OnFocusChangeListener() {

   @Override
   public void onFocusChange(View v, boolean hasFocus) {

    // When focus is lost check that the text field has valid values.

    if (!hasFocus) { {
      // Validate youredittext
    }
   }
});
```

##

activity_main.xml

##

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <GridLayout
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:padding="16dp"
        android:columnCount="2"
        android:useDefaultMargins="true">

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:inputType="numberDecimal"
            android:id="@+id/amountEditText"
            android:hint="Enter Amount"
            android:layout_column="0"
            android:maxLength="18"
            android:ems="20"
            android:padding="12dp"
            android:background="#BBDEFB"
            android:layout_columnSpan="2"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="15%"
            android:id="@+id/percentTextView"
            android:layout_gravity="center_vertical|center"/>

        <SeekBar
            android:layout_width="wrap_content"
            android:layout_height="40dp"
            android:id="@+id/percentSeekBar"
            android:indeterminate="false"
            android:max="30"
            android:progress="15"
            android:layout_gravity="fill_horizontal"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Tip"
            android:id="@+id/tipLabelTextView"
            android:layout_gravity="center"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:id="@+id/tipTextView"
            android:layout_gravity="fill_horizontal"
            android:background="#FFE0B2"
            android:gravity="center"
            android:padding="12dp"
            android:elevation="4dp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
```

```
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Total"
        android:id="@+id/totalLabelTextView"
        android:layout_gravity="center"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/totalTextView"
        android:layout_gravity="fill_horizontal"
        android:background="#FFE0B2"
        android:gravity="center"
        android:padding="12dp"
        android:elevation="4dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Credit\n Card"
        android:id="@+id/creditCardLabel"
        android:layout_gravity="center"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="number"
        android:id="@+id/creditCardEditText"
        android:hint="Credit Card Number"
        android:maxLength="19"
        android:ems="14"
        android:padding="12dp"
        android:background="#BBDEFB"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Expiry\n  Date"
        android:id="@+id/expiredDateLabel"
        android:layout_gravity="center"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="number"
        android:id="@+id/expiryDateEditText"
        android:hint="MM/YY"
        android:maxLength="5"
        android:ems="14"
        android:padding="12dp"
        android:background="#BBDEFB"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Name"
            android:id="@+id/nameLabel"
            android:layout_gravity="center"/>

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:inputType="text"
            android:id="@+id/nameEditText"
            android:hint="Card Holder Name"
            android:maxLength="18"
            android:ems="14"
            android:padding="12dp"
            android:background="#BBDEFB"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Security\n  Code"
            android:id="@+id/securityCodeLabel"
            android:layout_gravity="center"/>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <EditText
                android:layout_width="100dp"
                android:layout_height="wrap_content"
                android:inputType="number"
                android:id="@+id/securityCodeEditText"
                android:hint="CVV"
                android:maxLength="4"
                android:ems="20"
                android:padding="12dp"
                android:background="#BBDEFB"/>

            <Button
                android:id="@+id/submitButton"
                android:layout_width="150dp"
                android:layout_height="wrap_content"
                android:layout_marginLeft="20dp"
                android:textStyle="bold"
                android:text="Submit"
                android:onClick="submitButtonOnClick"/>

        </LinearLayout>


    </GridLayout>

</ScrollView>
```

##

MainActivity.java

##

```
package com.example.exam2sample;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle; // for saving state information
import android.text.Editable; // for EditText event handling
import android.text.TextWatcher; // EditText listener
import android.view.View;
import android.widget.Button;
import android.widget.EditText; // for bill amount input
import android.widget.SeekBar; // for changing the tip percentage
import android.widget.SeekBar.OnSeekBarChangeListener; // SeekBar listener
import android.widget.TextView; // for displaying text
import java.text.NumberFormat; // for currency formatting
import android.view.View.OnFocusChangeListener;
import android.widget.Toast;

// MainActivity class for the Tip Calculator app
public class MainActivity extends AppCompatActivity {

    // currency and percent formatter objects
    private static final NumberFormat currencyFormat =
            NumberFormat.getCurrencyInstance();
    private static final NumberFormat percentFormat =
            NumberFormat.getPercentInstance();

    private double billAmount = 0.0; // bill amount entered by the user
    private double percent = 0.15; // initial tip percentage
    private TextView percentTextView; // shows tip percentage
    private TextView tipTextView; // shows calculated tip amount
    private TextView totalTextView; // shows calculated total bill amount
    private EditText creditCardEditText; // credit card number
    private EditText expiryDateEditText; // expiry date
    private Button submitButton; // submit button

    // called when the activity is first created
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState); // call superclass onCreate
        setContentView(R.layout.activity_main); // inflate the GUI

        // get references to programmatically manipulated TextViews
        percentTextView = (TextView) findViewById(R.id.percentTextView);
        tipTextView = (TextView) findViewById(R.id.tipTextView);
        totalTextView = (TextView) findViewById(R.id.totalTextView);
        tipTextView.setText(currencyFormat.format(0));
        totalTextView.setText(currencyFormat.format(0));

        // set amountEditText's TextWatcher
        EditText amountEditText =
                (EditText) findViewById(R.id.amountEditText);
        amountEditText.addTextChangedListener(amountEditTextWatcher);
```

```
        // set percentSeekBar's OnSeekBarChangeListener
        SeekBar percentSeekBar =
                (SeekBar) findViewById(R.id.percentSeekBar);
        percentSeekBar.setOnSeekBarChangeListener(seekBarListener);

        // set creditCardEditText's FocusChangeListener
        creditCardEditText = (EditText)
findViewById(R.id.creditCardEditText);

creditCardEditText.setOnFocusChangeListener(creditCardEditTextFocusChangeList
ener);

        // set expiryDateEditText's FocusChangeListener
        expiryDateEditText = (EditText)
findViewById(R.id.expiryDateEditText);

expiryDateEditText.setOnFocusChangeListener(expiryDateEditTextFocusChangeList
ener);

    }

    // calculate and display tip and total amounts
    private void calculate() {
        // format percent and display in percentTextView
        percentTextView.setText(percentFormat.format(percent));

        // calculate the tip and total
        double tip = billAmount * percent;
        double total = billAmount + tip;

        // display tip and total formatted as currency
        tipTextView.setText(currencyFormat.format(tip));
        totalTextView.setText(currencyFormat.format(total));
    }

    // listener object for the SeekBar's progress changed events
    private OnSeekBarChangeListener seekBarListener =
            new OnSeekBarChangeListener() {
                // update percent, then call calculate
                @Override
                public void onProgressChanged(SeekBar seekBar, int progress,
                                              boolean fromUser) {
                    percent = progress / 100.0; // set percent based on
progress
                    calculate(); // calculate and display tip and total
                }

                @Override
                public void onStartTrackingTouch(SeekBar seekBar) { }

                @Override
                public void onStopTrackingTouch(SeekBar seekBar) { }
            };

    // listener object for the EditText's text-changed events
    private TextWatcher amountEditTextWatcher = new TextWatcher() {
```

```
        // called when the user modifies the bill amount
        @Override
        public void onTextChanged(CharSequence s, int start,
                                  int before, int count) {

            try { // get bill amount and display currency formatted value
                billAmount = Double.parseDouble(s.toString());
            }
            catch (NumberFormatException e) { // if s is empty or non-numeric
                billAmount = 0.0;
            }

            calculate(); // update the tip and total TextViews
        }

        @Override
        public void afterTextChanged(Editable s) { }

        @Override
        public void beforeTextChanged(CharSequence s, int start, int count,
int after) { }
    };

    // listener object for the EditText's focus-changed events
    private OnFocusChangeListener creditCardEditTextFocusChangeListener = new
OnFocusChangeListener() {
        @Override
        public void onFocusChange(View v, boolean hasFocus) {
            // When focus is lost check that the text field has valid values.
            if (!hasFocus) {
                {
                    //Toast.makeText(getApplicationContext(),
creditCardEditText.getText(), Toast.LENGTH_LONG).show();
                    String oldText = creditCardEditText.getText().toString();
                    String newText = "";
                    for (int i = 0; i < 16; i++) {
                        if (i == 4) newText = newText + " ";
                        if (i == 8) newText = newText + " ";
                        if (i == 12) newText = newText + " ";
                        newText = newText + oldText.charAt(i);
                    }
                    creditCardEditText.setText(newText);
                }
            }
        }
    };

    // listener object for the EditText's focus-changed events
    private OnFocusChangeListener expiryDateEditTextFocusChangeListener = new
OnFocusChangeListener() {
        @Override
        public void onFocusChange(View v, boolean hasFocus) {
            // When focus is lost check that the text field has valid values.
            if (!hasFocus) {
                {
                    String oldText = expiryDateEditText.getText().toString();
                    String newText = "";
```

```
                    for (int i=0; i<4; i++){
                        if (i==2) newText = newText + "/";
                        newText = newText + oldText.charAt(i);
                    }
                    expiryDateEditText.setText(newText);
                }
            }
        }
    };

    public void submitButtonOnClick (View view){
        Toast.makeText(getApplicationContext(), "Thank you for paying with
GWU-Pay!", Toast.LENGTH_LONG).show();
    }

}
```