

Questions**Name(s)** _____

1. Assuming *seconds* is initially 5, what is the result of *seconds* in the following?

seconds=0	seconds++
	mov ax, seconds
mov ax, 0	
	inc ax
mov seconds, ax	
	mov seconds, ax

2. How can we make the counter stop?
3. Make necessary changes for a *CounterListener* method *onReset()* that is called when *seconds* is reset to 0.

```
package edu.ius.rwisman.Counter5;

import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.TextView;

public class AndroidCounter5Activity extends Activity implements CounterListener {
    TextView count;
    Counter counter;

    public void setCount(final String s) {
        runOnUiThread(new Runnable() {
            public void run() {
                count.setText(s);
            }
        });
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        count = (TextView) findViewById(R.id.count);
        count.setText("0");
        counter = new Counter(this);
        counter.count();
    }

    public void reset(View v) {
        counter.reset();
    }
}
```

```

interface CounterListener {
    public void setCount(final String s);
}

class Counter {
    CounterListener delegate;
    int seconds = 0;

    public Counter(CounterListener delegate) {
        this.delegate=delegate;
    }

    public void reset() {
        synchronized(delegate) {
            seconds = 0;
        }
    }

    public void count() {
        new Thread(new Runnable() {
            public void run() {
                while(true)
                    try {
                        Thread.sleep(1000);
                        synchronized(delegate) {
                            seconds++;
                            delegate.setCount(seconds+"");
                        }
                    }
                catch(Exception e) {
                    Log.w("counter Thread Exception ", e+"");
                }
            }
        }).start();
    }
}

```

Questions

1. What does *extends Activity implements **ParserListener*** mean?
2. What does *new ITunesSAX(url, **this**)* do?
3. What is the purpose of:

```
interface ParserListener {  
    public void setTitle(final String s);  
}
```

4. Which thread is executed?
5. How many times is the delegate method, *setTitle()*, executed?
6. Are there any race-conditions? If so, where?
7. Does the worker thread stop? If so, where and when?
8. Modify to perform a call-back to *ParserListener* method *parseComplete()* when parsing is completed.

```
package edu.ius.rwisman.AndroidITunesListener;  
  
import java.io.*;  
import android.app.Activity;  
import android.os.Bundle;  
import android.widget.LinearLayout;  
import android.widget.TextView;  
import java.net.URL;  
import org.xml.sax.*;  
import org.xml.sax.helpers.DefaultHandler;  
import javax.xml.parsers.SAXParserFactory;  
import javax.xml.parsers.SAXParser;  
  
public class AndroidITunesListenerActivity extends Activity implements ParserListener{  
  
    LinearLayout layout;  
    AndroidITunesListenerActivity activity = this;  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        layout = new LinearLayout(this);  
        layout.setOrientation(1);  
        setContentView(layout);  
  
        try {  
            URL url = new URL("http://ax.phobos....topsongs/limit=10/rss.xml");  
            new ITunesSAX(url, this);  
        }  
        catch(Exception e) { System.out.println("Exception " + e); }  
    }  
}
```

```

public void setTitle(final String s) {
    runOnUiThread(new Runnable() {
        public void run() {
            TextView title = new TextView(activity);
            title.setText(s);
            layout.addView(title);
        }
    });
}

interface ParserListener {
    public void setTitle(final String s);
}

class ITunesSAX extends DefaultHandler {
    Boolean itemFound=false;
    ParserListener delegate;
    String element="";

    public ITunesSAX(final URL url, ParserListener delegate) {
        this.delegate = delegate;

        final ITunesSAX iTunesSAX = this;
        new Thread(new Runnable() {
            public void run() {
                SAXParserFactory factory = SAXParserFactory.newInstance();
                try {
                    InputStream in = url.openStream();
                    SAXParser saxParser = factory.newSAXParser();           // Parse input
                    saxParser.parse( in, iTunesSAX);
                } catch (Throwable t) { t.printStackTrace(); }
            }
        }).start();
    }

    public void startElement(String namespaceURI, String sName,           // simple name
                            String qName,                               // qualified name
                            Attributes attrs) throws SAXException {

        if( sName.equals("item") ) itemFound = true;
        element = "";
    }

    public void endElement(String namespaceURI, String sName,           // simple name
                           String qName                               // qualified name
                           ) throws SAXException {
        if(itemFound && sName.equals("title") )
            delegate.setTitle(element);           // Call-back to user
    }

    public void characters(char[] buf, int offset, int length) throws SAXException {
        if( length <= 0) return;
        element = element + new String(buf, offset, length);
    }

    public void startDocument() throws SAXException {}
    public void endDocument() throws SAXException {}
}

```