Certamen 3, Programación II

Prof. Rodrigo Olivares
Ayud. Juan Carlos Tapia
Noviembre 30, 2015

Instrucciones:

- El puntaje máximo del certamen es 100 %, siendo el 60 % el mínimo requerido para aprobar.
- Responda cada pregunta en la hoja indicada, agregando su nombre. Si no responde alguna pregunta, debe entregar la hoja con su nombre e indicar que **no responde**.
- El certamen es **individual**. Cualquier intento de copia, será sancionado con nota **1,0**.

1. *20pts.* De las siguentes afirmaciones, evalúe con verdadero ($\mathcal{V}$) o falso ($\mathcal{F}$) las siguientes afirmaciones (*1pt c/u*). **Se descontará una respuesta correcta por cada respuesta incorrecta**.

   a) __V__ Un thread es un flujo de proceso que se ejecuta en memoria.
   b) __V__ Un thread puede ser instanciado como objeto.
   c) __F__ Para crear un thread se debe extender de una sub-clase Thread.
   d) __F__ Para iniciar un thread se utiliza el método *run*.
   e) __F__ Para iniciar un thread se debe sobre-escribir el método *start*.
   f) __F__ En el ciclo de vida de una hebra, el estado *New* es el encargado de crear e instanciar una thread.
   g) __V__ En el ciclo de vida de una hebra, el estado *Dead* es invocado generalmente por el método *stop*.
   h) __V__ Un recurso compartido puede ser una clase.
   i) __V__ Un recurso compartido siempre debe estar sincronizado.
   j) __F__ La API *Swing* sustituye a API AWT.
   k) __F__ AWT incopora los *JComponents* de *Swing*.
   l) __F__ *JPane* es habitualmente utilizada para crear la ventana principal.
   m) __V__ Para crear la ventana (usando *Swing*) se de extender la clase *JFrame*.
   n) __F__ El método *getPaneContent* de la clase *JFrame* obtiene el panel principal.
   ñ) __V__ El método *add* de la clase *JFrame* permite agregar componentes al panel.
   o) __F__ Para agregar funcionalidad a un JButton se debe crear y agregar una clase que implemente un ActionEvent
   p) __F__ Para agregar funcionalidad a una JList se debe sobreescribir el método actionList(ActionPerformed)
   q) __V__ El componente JFileChooser es un componente diálogo de selección de archivos.
   r) __V__ Algunos de componentes que permiten ingresar texto son: JPasswordField, JTextField y JTextArea.
   s) __V__ Algunos de componentes que permiten manipular eventos/acciones son: JList, JButton y JTextField.

2. *40pts.* Simular el proceso de giro y depósito de dinero de una cuenta corriente. La idea es controlar el ingreso y egreso de dinero, de tal manera que no sea factible girar dinero si la cuenta está en valor cero 0 negativo y además no sea factible mantener más de $100.000 (cien mil pesos) en la cuenta. Cada transación debe esperar un número aletorio de segundos (entre 1 y 5) y el monto a depositar o girar también deberá ser aleatorio entre $1.000 (mil) y $10.000 (diez mill). Considere como **región crítica** la clase que gestiona la cuenta corriente (el saldo), por lo cual debe utilizar **bloqueos/sincronización de thread**.

```
public class CuentaCorriente {

    private final int MAX_AHORRO = 100000;
    private int saldo = 0;

    public synchronized void depositar(int dinero) {
        try {
            while (saldo >= MAX_AHORRO) {
                wait();
            }
            saldo += dinero;
            System.out.println("Deposito: " + dinero + " Saldo: " + getSaldo());
            notifyAll();
        } catch (InterruptedException ex) {
        }
    }

    public synchronized void girar(int dinero) {
        try {
            while (saldo <= 0) {
                wait();
            }
            saldo -= dinero;
            System.out.println("Giro: " + dinero + " Saldo: " + getSaldo());
            notifyAll();
        } catch (InterruptedException ex) {
        }
    }

    private int getSaldo() {
        return saldo;
    }
}

public class ThreadCuentaCorrienteDepositar extends Thread {

    private final CuentaCorriente cuentaCorriente;

    public ThreadCuentaCorrienteDepositar(CuentaCorriente cuentaCorriente1) {
        this.cuentaCorriente = cuentaCorriente1;
    }

    @Override
    public void run() {
        while (true) {
            try {
                ThreadCuentaCorrienteDepositar.sleep((long) (1 + (int)(Math.random() * 5)) * 1000);
                cuentaCorriente.depositar((int)(10000 * Math.random()));
            } catch (InterruptedException ex) {
            }
        }
    }
}

public class ThreadCuentaCorrienteGirar extends Thread {

    private final CuentaCorriente cuentaCorriente;

    public ThreadCuentaCorrienteGirar(CuentaCorriente cuentaCorriente1) {
        this.cuentaCorriente = cuentaCorriente1;
    }

    @Override
    public void run() {
        while (true) {
            try {
                ThreadCuentaCorrienteGirar.sleep((long) (1 + (int) (Math.random() * 5)) * 1000);
                cuentaCorriente.girar((int) (10000 * Math.random()));
            } catch (InterruptedException ex) {
            }
        }
    }
}
```

```
public class Principal {

    public static void main(String[] args) {

        CuentaCorriente cc = new CuentaCorriente();

        ThreadCuentaCorrienteDepositar tccd = new ThreadCuentaCorrienteDepositar(cc);
        ThreadCuentaCorrienteGirar tccg = new ThreadCuentaCorrienteGirar(cc);

        tccd.start();
        tccg.start();

        try {
            tccd.join();
            tccg.join();
        } catch (InterruptedException ex) {
        }
    }
}
```

3. *40pts.* Construya una aplicación Java con interfaz gráfica que pemita realizar transformaciones de sistemas números (binario, octal y hexadecimal) de un valor entero ingresado en un campo de texto y muestre el resultado en otro campo de texto no editable. Para la transformación utilice los métodos: *toBinaryString*(), *toOctalString*() y *toHexString*() de la clase *Integer*, para transformar a binario, ocatl y hexadecimal, respectivamente.

```
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

public class Translate extends JFrame {

    JTextField num, result;
    JButton toBin, toOct, toHex;
    private static final String BIN_TEXT = "Binario";
    private static final String OCT_TEXT = "Octal";
    private static final String HEX_TEXT = "Hexadecimal";

    public Translate() {
        num = new JTextField(50);
        result = new JTextField(50);
        toBin = new JButton(BIN_TEXT);
        toOct = new JButton(OCT_TEXT);
        toHex = new JButton(HEX_TEXT);

        result.setEnabled(false);

        toBin.addActionListener(new TranslateActionListener());
        toOct.addActionListener(new TranslateActionListener());
        toHex.addActionListener(new TranslateActionListener());

        setLayout(new FlowLayout(FlowLayout.CENTER, 10, 20));
        add(num);
        add(toBin);
        add(toOct);
        add(toHex);
        add(result);

        setSize(600, 150);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        Translate t = new Translate();
    }

    class TranslateActionListener implements ActionListener {

        @Override
        public void actionPerformed(ActionEvent ae) {
            String n = null;
            try {
                if (BIN_TEXT.equals(ae.getActionCommand())) {
                    n = Integer.toBinaryString(Integer.valueOf(num.getText()).intValue());
                } else {
                    if (OCT_TEXT.equals(ae.getActionCommand())) {
                        n = Integer.toOctalString(Integer.valueOf(num.getText()).intValue());
                    } else {
                        if (HEX_TEXT.equals(ae.getActionCommand())) {
                            n = Integer.toHexString(Integer.valueOf(num.getText()).intValue()).toUpperCase();
                        }
                    }
                }
                result.setText(n);
            } catch (NumberFormatException nfe) {
                num.setText("");
                result.setText("");
                JOptionPane.showMessageDialog(null, "El valor ingresado no es un nmero entero.\n
                    Intentelo nuevamente.", "Error", JOptionPane.ERROR_MESSAGE);
            }
        }
    }
}
```