Lösningsförslag till tentamen

Kursnamn	**Objektorienterade applikationer**
Tentamensdatum	**2013-08-28**
Program	**DAI2**
Läsår	**2012/2013, lp 3**
Examinator	**Uno Holmer**

Uppgift 1 (6+6 p)

a) (6 p)

```
public class TimerGui extends JFrame implements ActionListener
{
    private JLabel timeLabel;
    private JButton setButton, startButton;
    private int setTime;
    private Timer timer;

    TimerGui() {
        makeFrame();
        timer = new Timer(1000,this);
        setTime = 0;
    }

    private void makeFrame() {
        setLayout(new GridLayout(3,1,2,2));

        timeLabel = new JLabel("0",SwingConstants.CENTER);
        add(timeLabel);

        setButton = new JButton("+1");
        setButton.addActionListener(this);
        add(setButton);

        startButton = new JButton("START");
        startButton.addActionListener(this);
        startButton.setEnabled(false);
        add(startButton);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        pack();
        setVisible(true);
    }
    ...
}
```

b) (6 p)

```
public void actionPerformed(ActionEvent e) {
    if ( e.getSource() == setButton ) {
        timeLabel.setText("" + ++setTime);
        startButton.setEnabled(true);
    } else if ( e.getSource() == startButton ) {
        timer.start();
        setButton.setEnabled(false);
        startButton.setEnabled(false);
    } else if ( e.getSource() == timer ) {
        timeLabel.setText("" + --setTime);
        if ( setTime == 0 ) {
            timer.stop();
            Toolkit.getDefaultToolkit().beep();
            setButton.setEnabled(true);
        }
    }
}
```

Uppgift 2 (12 p)

```
public static void doubleToText(String inFile,String outFile)
{
    int size = length(inFile)/(Double.SIZE/8);
    double[] values = new double[size];
    try {
        DataInputStream in =
            new DataInputStream(new FileInputStream(inFile));
        for ( int i = 0; i < size; i++ )
            values[i] = in.readDouble();
        in.close();
        Arrays.sort(values);
        PrintWriter out = new PrintWriter(new FileWriter(outFile));
        for ( double d : values )
            out.println(d);
        out.close();
    }
    catch (FileNotFoundException e) {
        System.out.println("ConvertFile: Cannot open " + inFile);
    }
    catch (IOException e) {
        System.out.println("ConvertFile: read or write error");
    }
}
```

Uppgift 3 (12+6+6 p)

a) (12 p)

```
public class Account {
    private int balance;

    public Account(int balance) {
        this.balance = balance;
    }

    public synchronized void put(int amount)
                              throws IllegalArgumentException
    {
        if ( amount <= 0 )
            throw new IllegalArgumentException();

        System.out.println(balance + " + " + amount + " = " +
                                (balance+amount));
        balance += amount;
        notify();
     }

    public synchronized void take(int amount)
                              throws IllegalArgumentException
    {
        if ( amount <= 0 )
            throw new IllegalArgumentException();

        while ( balance < amount ) {
            try {
                wait();
            }
            catch(InterruptedException e) {
                return;
            }
        }
        System.out.println(balance + " - " + amount + " = " +
                                (balance-amount));
        balance -= amount;
    }

    public int getBalance() {
        return balance;
    }
}
```

b) (en av b eller c ger 6 p)

```
public class Employer extends Thread {
    private Account account;
    private Random random;

    public Employer(Account account) {
        this.account = account;
        random = new Random();
    }

    public void run() {
        while ( ! interrupted() ) {
            try {
                sleep(30000);
            }
            catch ( InterruptedException e ) {
                break;
            }
            int amount = random.nextInt(10000)+20000;
            account.put(amount);
        }
    }
}
```

c)

```
public class Employed extends Thread {
    private Account account;
    private Random random;

    public Employed(Account account) {
        this.account = account;
        random = new Random();
    }

    public void run() {
        while ( ! interrupted() ) {
            try {
                sleep(1000);
            }
            catch ( InterruptedException e ) {
                break;
            }
            int amount = random.nextInt(1000)+1000;
            account.take(amount);
        }
    }
}
```

Uppgift 4 (12 p)

```
public void actionPerformed(ActionEvent e) {
    if ( e.getSource() == sendButton ) {
         sendDatagram(addressField.getText(),textArea.getText());
    }
}

private void sendDatagram(String address,String text) {
    try {
        InetAddress toAddr = InetAddress.getByName(address);
        DatagramSocket socket = new DatagramSocket();
        byte[] data = text.getBytes();
        socket.send(new DatagramPacket(data,data.length,toAddr,
                                       RECEIVER_PORT));
        waitForAcc();
    }
    catch ( UnknownHostException e ) {}
    catch ( SocketException e ) {}
    catch ( IOException e ) {}
}

private void waitForAcc() {
    DatagramSocket socket = null;
    try {
        socket = new DatagramSocket(ACK_PORT);
        socket.setSoTimeout(10000);
        byte[] buf = new byte[1024];
        DatagramPacket packet = new DatagramPacket(buf,buf.length);
        socket.receive(packet);
        String acc = new String(packet.getData());
        JOptionPane.showMessageDialog(null,acc,null,
            JOptionPane.INFORMATION_MESSAGE);
        socket.close();
    }
    catch (SocketTimeoutException e) {
        JOptionPane.showMessageDialog(null,"message was lost",null,
            JOptionPane.INFORMATION_MESSAGE);
        socket.close();
    }
    catch (IOException e) {}
}
```

Uppgift 5 (6 p)

```
private void makeButtons(String nameFile) {
    try {
        Scanner sc = new Scanner(new File(nameFile));
        while ( sc.hasNextLine() ) {
            String className = sc.nextLine();
            Class c = Class.forName(className);
            final Action obj = (Action)c.newInstance();
            JButton button = new JButton(className);
            add(button);
            button.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    obj.execute();
                }
            });
        }
    }
    catch ( FileNotFoundException e ) {
        System.out.println("Cannot open file");
    }
    catch ( ClassNotFoundException e ) {
        System.out.println("Cannot load class");
    }
    catch ( Exception e ) {
        System.out.println("Cannot proceed");
    }
}
```