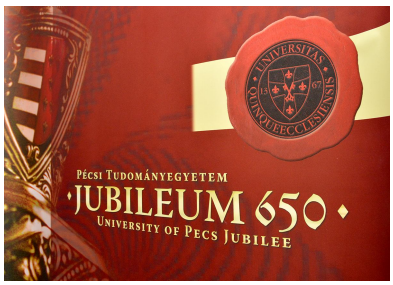
PTE jubileum



A Pécsi Tudományegyetem 2017 szeptemberében ünnepelte fennállásának 650. évfordulóját. A jubileum rendezvényeit szerencsére ingyenesen lehetett látogatni, a feladat kedvéért azonban fizetőssé tesszük őket.

Az ünnepséghez tehát rendezvények tartoznak. Minden egyes rendezvény a címével, időpontjával (jelenleg lehet sima String) és a belépőjegy árával jellemezhető.

Természetesen vannak (lehetséges) résztvevők is, sőt, a PTE azonosítóval rendelkezők még kedvezményt is kapnak. Minden résztvevőnek van neve (a név egyébként sohasem lehet egyedi azonosító, de most az egyszerűség kedvéért feltesszük, hogy minden név más). A PTE-s résztvevőt ezen kívül még egy azonosító is jellemzi.

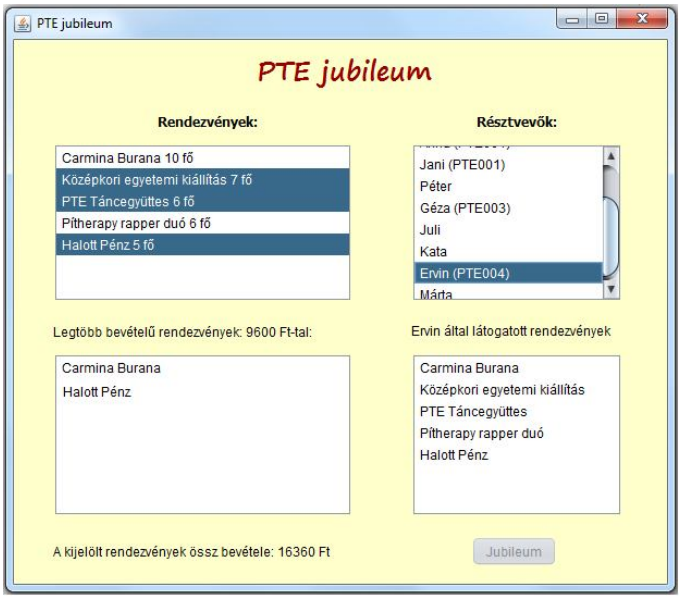
Egy résztvevő akkor vesz részt egy adott rendezvényen, ha kifizeti a rendezvény részvételi díját. A PTE-s résztvevők egységesen 10% kedvezményt kapnak.

Minden egyes résztvevő esetén tudjuk felsorolni azokat a rendezvényeket, amelyeken részt vett az illető, illetve minden rendezvény esetén állapítsuk meg a résztvevők számát és a rendezvény bevételét.

Írjunk programot a rendezvények szimulálására, melynek eredményét grafikus felületen jelenítsük meg:

A képen látható felület belső mérete 600 * 500 pixel.

A program indulásakor már legyenek olvashatóak a rendezvények (cím, időpont, jegyár) és a potenciális résztvevők listájának adatai, illetve adatbeolvasáskor minden résztvevő kapjon véletlenszerűen valamennyi pénzt.

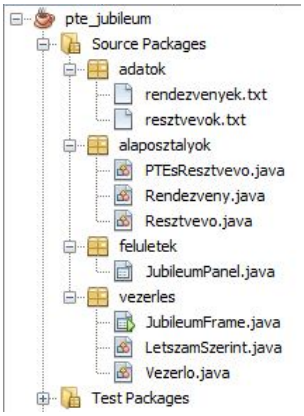


A Jubileum feliratú gomb megnyomásakor fusson le a szimuláció, vagyis minden egyes rendezvény esetén mindegyik résztvevő „döntse el” véletlenül, hogy részt akar-e venni a rendezvényen vagy sem. (A részvételi kedv kb. 80%-os.) A szimuláció lefutása után a gomb váljon inaktívvá.

A gombnyomás hatására jelenjen meg a legtöbb bevételt hozó rendezvények listája is, maguk a rendezvények pedig legyenek résztvevőszám szerint csökkenően kiírva (most úgy, hogy a rendezvény neve és a létszám).

A maximális bevételű rendezvények listája alatt lehessen látni a kijelölt rendezvények összbevételét, a résztvevők listája alatt pedig azt, hogy a kiválasztott résztvevő mely rendezvényeken vett részt. Mindkét felirat csak akkor jelenjen meg, ha már választottunk a megfelelő listából.

Megoldásrészletek:



A feladat a korábbiakhoz képest nem tartalmaz sok újdonságot, ezért különösebb magyarázat nélkül közöljük az egyes kódrészleteket.

A projekt szerkezete:

`Rendezveny` osztály (getterek, setterek nélkül):

```
public class Rendezveny {

    private String cim;
    private String idoPont;
    private int jegyAr;
    private int resztvevokSzama;
    private int bevetel;

    private static boolean rendezett;

    public Rendezveny(String cim, String idoPont, int jegyAr) {
        this.cim = cim;
        this.idoPont = idoPont;
        this.jegyAr = jegyAr;
    }

    /** Résztvevő részt vesz ...5 lines */
    public void resztVesz(Resztvevo resztvevo){
        if(resztvevo.belephet(this)){
            resztvevo.fizet(resztvevo.reszveteliDij(this));
            resztvevo.resztVesz(this);
            resztvevokSzama++;
            bevetel += resztvevo.reszveteliDij(this);
        }
    }

    @Override
    public String toString() {
        if(rendezett) return cim + " " + resztvevokSzama + " fő";
        return cim + ", " + idoPont + ", " + jegyAr + " Ft";
    }
```

Resztvevo:

```
public class Resztvevo {

    private String nev;
    private int penz;
    private List<Rendezveny> rendezvenyek = new ArrayList<>();

    public Resztvevo(String nev) {
        this.nev = nev;
    }

    public boolean belephet(Rendezveny rendezveny){
        return this.reszveteliDij(rendezveny) <= penz;
    }

    protected int reszveteliDij(Rendezveny rendezveny) {
        return rendezveny.getJegyAr();
    }

    public void fizet(int reszveteliDij) {
        if(this.penz >= reszveteliDij) this.penz -= reszveteliDij;
    }

    public void penztKap(int penz) {
        this.penz += penz;
    }

    void resztVesz(Rendezveny rendezveny) {
        if(!rendezvenyek.contains(rendezveny))rendezvenyek.add(rendezveny);
    }

    @Override
    public String toString() {
        return nev;
    }
```

PTE-s résztvevő:

```java
public class PTEsResztvevo extends Resztvevo{

    private String pteAzonosito;
    private static double kedvezmenySzazalek;

    public PTEsResztvevo(String nev, String pteAzonosito) {
        super(nev);
        this.pteAzonosito = pteAzonosito;
    }

    @Override
    protected int reszveteliDij(Rendezveny rendezveny) {
        return (int) (super.reszveteliDij(rendezveny)* (1-kedvezmenySzazalek/100));
    }

    @Override
    public String toString() {
        return super.toString() + " (" + this.pteAzonosito + ")";
    }
```

A frame generált kódok nélküli része:

```java
public class JubileumFrame extends javax.swing.JFrame {

    private final int SZELESSEG = 616;
    private final int MAGASSAG = 538;
    private String CIM = "PTE jubileum";

    public JubileumFrame() {
        initComponents();

        this.setSize(SZELESSEG, MAGASSAG);
        this.setTitle(CIM);
        this.setLocationRelativeTo(null);
    }

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new JubileumFrame().start();
            }
        });
    }
```

```
    private void start() {
        this.setVisible(true);
        Vezerlo vezerlo = new Vezerlo(jubileumPanel1);
        jubileumPanel1.setVezerlo(vezerlo);
        vezerlo.indit();
    }
}
```

A vezérlő osztály ide vonatkozó része:

```
public class Vezerlo {


    private final String RENDEZVENY_ELERES = "/adatok/rendezvenyek.txt";
    private final String RESZTVEVOK_ELERES = "/adatok/resztvevok.txt";
    private final String CHAR_SET = "UTF-8";

    private final double KEDVEZMENY_SZAZALEK = 10;
    private final int ALSO_PENZ = 1000;
    private final int FELSO_PENZ = 10000;
    private final double KEDV_SZAZALEK = 0.8;

    private List<Rendezveny> rendezvenyLista = new ArrayList<>();
    private List<Resztvevo> resztvevoLista = new ArrayList<>();

    private JubileumPanel jubileumPanel;

    public Vezerlo(JubileumPanel jubileumPanel) {
        this.jubileumPanel = jubileumPanel;
    }

    void indit() {
        adatbevitel();
    }
```

```
private void adatbevitel() {

    PTEsResztvevo.setKedvezmenySzazalek(KEDVEZMENY_SZAZALEK);

    try (InputStream ins = this.getClass().getResourceAsStream(RENDEZVENY_ELERES);
            Scanner fajlScanner = new Scanner(ins, CHAR_SET)) {

        String cim, idoPont;
        int ar;
        String sor, adatok[];
        Rendezveny rendezveny;
        while (fajlScanner.hasNextLine()) {
            sor = fajlScanner.nextLine();
            adatok = sor.split(";");
            cim = adatok[0];
            idoPont = adatok[1];
            ar = Integer.parseInt(adatok[2]);
            rendezveny = new Rendezveny(cim, idoPont, ar);
            rendezvenyLista.add(rendezveny);
        }
    } catch (IOException ex) {
        Logger.getLogger(JubileumPanel.class.getName()).log(Level.SEVERE, null, ex);
    }

    try (InputStream ins = this.getClass().getResourceAsStream(RESZTVEVOK_ELERES);
            Scanner fajlScanner = new Scanner(ins, CHAR_SET)) {
        Resztvevo resztvevo = null;
        String sor, adatok[];
        while (fajlScanner.hasNextLine()) {
            sor = fajlScanner.nextLine();
            adatok = sor.split(";");
            if (adatok.length == 1) {
                resztvevo = new Resztvevo(adatok[0]);
            }
            if (adatok.length == 2) {
                resztvevo = new PTEsResztvevo(adatok[0], adatok[1]);
            }
            resztvevo.penztKap((int) (Math.random()
                    * (FELSO_PENZ - ALSO_PENZ + 1) + ALSO_PENZ));
            resztvevoLista.add(resztvevo);
        }
    } catch (IOException ex) {
        Logger.getLogger(JubileumPanel.class.getName()).log(Level.SEVERE, null, ex);
    }

    jubileumPanel.kiir(rendezvenyLista, resztvevoLista);
}
```

A felület kialakítása már biztosan megy önállóan is.

A panel ide vonatkozó része:

```
public class JubileumPanel extends javax.swing.JPanel {


    private DefaultListModel<Rendezveny> rendezvenyModel
                                                = new DefaultListModel<>();
    private DefaultListModel<Resztvevo> resztvevoModel
                                                = new DefaultListModel<>();
    private DefaultListModel<String> resztvevoRendezvenyeModel
                                                = new DefaultListModel<>();
    private DefaultListModel<String> maxRendezvenyekModel
                                                = new DefaultListModel<>();

    private final String MAX_LABEL_SZOVEG = "Legtöbb bevételű rendezvények: ";

    private Vezerlo vezerlo;

    public JubileumPanel() {
        initComponents();
        lstRendezvenyek.setModel(rendezvenyModel);
        lstResztvevok.setModel(resztvevoModel);
        lstResztvevokRendezvenyei.setModel(resztvevoRendezvenyeModel);
        lstMaxBevetel.setModel(maxRendezvenyekModel);
        lblMax.setText(MAX_LABEL_SZOVEG);
    }

    public void kiir(List<Rendezveny> rendezvenyLista,
                     List<Resztvevo> resztvevoLista) {
        for (Rendezveny rendezveny : rendezvenyLista) {
            rendezvenyModel.addElement(rendezveny);
        }
        for (Resztvevo resztvevo : resztvevoLista) {
            resztvevoModel.addElement(resztvevo);
        }
    }

    public void setVezerlo(Vezerlo vezerlo) {
        this.vezerlo = vezerlo;
    }
```

A gombnyomás esemény a panelen:

```
    private void btnJubileumActionPerformed(java.awt.event.ActionEvent evt) {
        vezerlo.jubileum();
        vezerlo.maxKereses();
        vezerlo.rendezve();
        btnJubileum.setEnabled(false);
    }
```

A hivatkozott metódusok:

`Vezerlo` osztály:

```
public void jubileum() {
    for (Rendezveny rendezveny : rendezvenyLista) {
        for (Resztvevo resztvevo : resztvevoLista) {
            if (Math.random() < KEDV_SZAZALEK) {
                rendezveny.resztVesz(resztvevo);
            }
        }
    }
}

public void maxKereses() {
    int max = rendezvenyLista.get(0).getBevetel();
    for (Rendezveny rendezveny : rendezvenyLista) {
        if (rendezveny.getBevetel() > max) {
            max = rendezveny.getBevetel();
        }
    }

    List<Rendezveny> maxBeveteluRendezvenyek = new ArrayList<>();
    for (Rendezveny rendezveny : rendezvenyLista) {
        if (rendezveny.getBevetel() == max) {
            maxBeveteluRendezvenyek.add(rendezveny);
        }
    }

    jubileumPanel.maxKiiras(maxBeveteluRendezvenyek, max);
}

public void rendezve() {

    Collections.sort(rendezvenyLista, new LetszamSzerint());

    Rendezveny.setRendezett(true);
    jubileumPanel.rendezveIr(rendezvenyLista);
}
```

`JubileumPanel` osztály:

```
public void maxKiiras(List<Rendezveny> maxBeveteluRendezvenyek, int max) {
    lblMax.setText(MAX_LABEL_SZOVEG + max + " Ft-tal:");
    maxRendezvenyekModel.clear();
    for (Rendezveny rendezveny : maxBeveteluRendezvenyek) {
        maxRendezvenyekModel.addElement(rendezveny.getCim());
    }
}
```

```
    public void rendezveIr(List<Rendezveny> rendezvenyLista) {

        rendezvenyModel.clear();
        for (Rendezveny rendezveny : rendezvenyLista) {
            rendezvenyModel.addElement(rendezveny);
        }
    }
```

A rendezéshez használt `LetszamSzerint` osztály:

```
public class LetszamSzerint implements Comparator<Rendezveny>{

    @Override
    public int compare(Rendezveny o1, Rendezveny o2) {
        return o2.getResztvevokSzama() - o1.getResztvevokSzama();
    }
}
```

Végül a listakezelő események:

```
    private void lstResztvevokMousePressed(java.awt.event.MouseEvent evt) {
        Resztvevo resztvevo = (Resztvevo) lstResztvevok.getSelectedValue();
        lblResztvevo.setText(resztvevo.getNev() +
                                    " által látogatott rendezvények ");
        resztvevoRendezvenyeModel.clear();
        for (Rendezveny rendezveny : resztvevo.getRendezvenyek()) {
            resztvevoRendezvenyeModel.addElement(rendezveny.getCim());
        }
    }

    private void lstRendezvenyekValueChanged(javax.swing.event.ListSelectionEvent evt) {
        int ossz = 0;
        List<Rendezveny> valasztottak = lstRendezvenyek.getSelectedValuesList();
        for (Rendezveny rendezveny : valasztottak) {
            ossz += rendezveny.getBevetel();
        }
        lblRendezveny.setText(
           String.format("A kijelölt rendezvények össz bevétele: %d Ft", ossz));
    }
```

Most mindegy, hogy egérkattintás vagy a lista elemének megváltozása az esemény. Ha megengednénk, hogy a jubileum gombot többször is megnyomjuk, akkor az első esetben csak az egérkattintás lenne jó, mert ekkor kétszer egymás után ugyanarra kattintva láthatnánk az esetleges változást. Ha nagyon szigorúan akarjuk megoldani, akkor az egérkattintáskor azt is figyelni kellene, hogy a listafelület aktív-e. És a szigorúsághoz az is hozzátartozna, hogy a kiválasztott példányok össz-bevételét is a vezérlő számolná, a panel csak kiírná az eredményt.