GYMNÁZIUM A STŘEDNÍ ODBORNÁ ŠKOLA ROKYCANY

ROČNÍKOVÁ PRÁCE

SUDOKU

Autor práce:

Marek Babiar, 4.I

Rokycany 2018

29-3

Prohlášení

Prohlašuji, že jsem svoji ročníkovou práci z programování Java vytvořil samostatně s využitím legálného programového vybavení.

……..
podpis

Obsah

Prohlášení....2
Anotace....4
Úvod....5
Rozvržení aplikace....6
GUI.java....6
Statistiky.java....6
Potvrzovani.java....6
Generovani.java....6
Stats.java....7
Skore.java....7
GUI.java....8
Importy....9
Hlavní proměnné....9
Vlastnosti/vytvoření okna....10
Tlačítko....10
Vytvoření prázdné hrací plochy 9x9....11
Omezení na pouhé čísla 1-9....12
Menu – statistiky....12
Statistiky.java....13
Načítání statistik....13
„nálepky“....14
Reset tlačítko....14
Stats.java....15
Reset()....15
Nacitani()....16
Vyhra()....17
Generování pole....19
Potvrzovani.java....21
Závěr:....22
Použité zdroje....23

Anotace

V této ročníkové práci jsem dostal za úkol vytvořit aplikaci Sudoku, která obsahuje různé obtížnosti hry, nápovědu pro další tah a tvorbu statistik hráčů a to vše v programovacím jazyce Java.

Annotation

In this seminar work I have got a task to create a Sudoku application that includes different game difficulty, hint for the next move and creation of player statistics and all this in Java language.

Úvod

Pro moji ročníkovou práci jsem si vybral aplikaci Sudoku. Tato aplikace zahrnuje můj vlastní generátor pole čísel 9x9 za využití podmínek Sudoku, obsahuje menu ve kterém je vytváření nových her, ukládání jedné rozehrané hry a její zpětné načítání, vybrání tří různých obtížností(začátečník, pokročilý, expert), statistiky a ukončení aplikace. Dále tato aplikace obsahuje tlačítko pro nápovědu a kontrolu. A též nesmíme zapomenout na časovač, který je podstatnou částí pro tvorbu statistik.

Statistiky jsou samostatné okno s jedním tlačítkem Reset, který vymaže statistiky hráčů(jména, časy) a dále statistiky obsahují spoustou „nálepek“ (labels), ve kterých jsou vloženy jména a časy společně s nadpisy (začátečník, pokročilý, expert).
Většina grafického rozhraní je vytvořená pomocí swing designeru.

Rozvržení aplikace

Aplikace obsahuje 6 souboru typu java, kde GUI a Statistiky jsou graficky rozvržená okna.

GUI.java

– metoda main(), hlavní okno – menu, hrací plocha, tlačítka, časovač

Statistiky.java

– vedlejší okno, spouštění pomocí menu v GUI nebo po dokončení hry

Potvrzovani.java

– 3 metody volané z menu v GUI (nová hra, uložení, načítání), jestli si uživatel opravdu přeje udělat danou operaci

Generovani.java

– 7 metod spuštené z GUI nebo z Potvrzovani
- ulozeni(), nacitani(), novaHra() jsou metody volané z Potvrzovani
- napoveda(), kontrola() jsou metody volané z GUI po zmáčknutí příslušných tlačítek
- generatorPole() je metoda volaná po stisknutí Nové hry v GUI
vytvoří pole čísel 9x9 naplněné vhodnými čísly
- genObtiznost() vezme pole čísel vytvořené generátorem, vymaže náhodný počet čísel tak, aby uživatel mohl hrát a následně toto pole s chybějícími čísli vloží do hrací plochy v GUI

Stats.java

– zde se nachází 3 metody, nacitani(), vyhra() a reset()

- metoda nacitani() načte statistiky, je volaná na začátku vytvoření okna pro Statistiky a kdy se mažou statistiky

- vyhra() je volaná po stisku tlačítka Kontrola nebo Nápověda v GUI tehdy, pokud je hrací pole správně zaplněno čísly

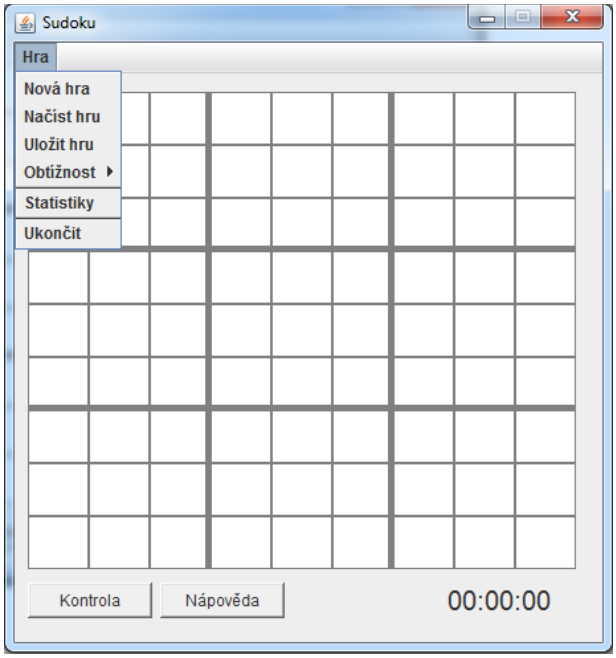
- reset() je volaná po stisku tlačítka Reset ve Statistikách, kde se vytvoří prázdné pole statistik hráčů a přepíše stávající statistiky

Skore.java

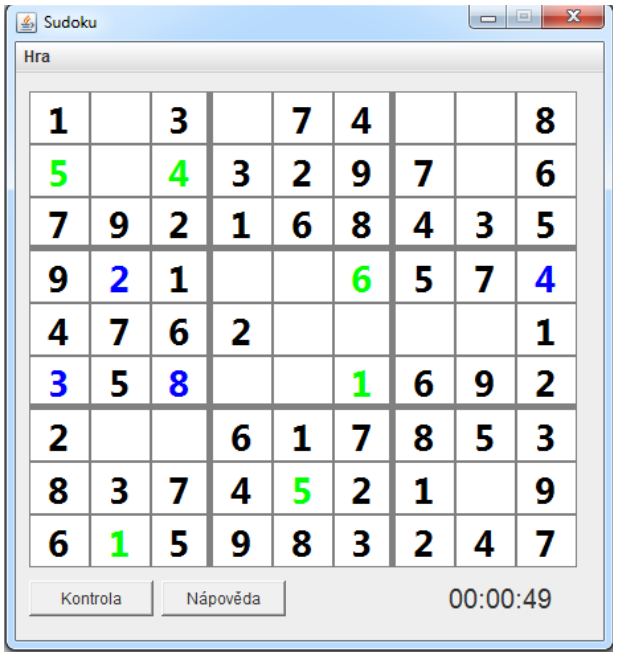
– vytvoření třídy Skore, obsahuje proměnné jmeno, hour, min, sec

- důležité pro statistiky, pro jejich zobrazení

GUI.java



Zde je základní okno aplikace,
po stisknutí tlačítka „Hra“
se vysune menu od tlačítka Nová hra
po tlačítko Ukončit.
Časovač vpravo dole se aktivuje
pomocí Nové hry nebo po načtení
hry, jestli nějaká uložená hra existuje.



Hrací plocha může obsahovat 3 typy
čísel nebo žádné číslo.
Černá čísla značí vygenerovanou
předlohu pomocí generátoru- nejde
změnit.
Zelené číslo se připíše po stisknutí
tlačítka Nápověda, kde se z
vygenerované předlohy přidá
1 číslo – nejde změnit.
Modré čísla jsou napsaná od
uživatele, tyto čísla editovat jdou.

Prázdná políčka jsou editovatelná též, uživatel do nich ale může vložit čísla od 1- 9, nulu a jiné znaky vložit nejdou.

Importy

```
1  import java.awt.Button;
2  import java.awt.Color;
3  import java.awt.EventQueue;
4  import java.awt.Font;
5  import java.awt.GridLayout;
6  import java.awt.Label;
7  import java.awt.event.ActionEvent;
8  import java.awt.event.ActionListener;
9
10 import javax.swing.BorderFactory;
11 import javax.swing.ButtonGroup;
12 import javax.swing.JFrame;
13 import javax.swing.JMenu;
14 import javax.swing.JMenuBar;
15 import javax.swing.JMenuItem;
16 import javax.swing.JPanel;
17 import javax.swing.JRadioButtonMenuItem;
18 import javax.swing.JSeparator;
19 import javax.swing.JTextField;
20 import javax.swing.text.AttributeSet;
21 import javax.swing.text.BadLocationException;
22 import javax.swing.text.Document;
23 import javax.swing.text.DocumentFilter;
24 import javax.swing.text.PlainDocument;
```

Jsou zde použity základní knihovny od vytváření menu, různých textových panelů, barev a fontů až po knihovny potřebné pro omezení vkládání písmen do textových polí.

Hlavní proměnné

```
26 public class GUI {
27
28
29     private JFrame frmSudoku;
30     private final ButtonGroup btObtiznost = new ButtonGroup();
31     private JPanel pBunky;
32
33     static JRadioButtonMenuItem rdbtZacatecnik;
34     static JRadioButtonMenuItem rdbtPokrocily;
35     static JRadioButtonMenuItem rdbtExpert;
36
37     static JTextField poleBunek[]=new JTextField[81];
38     static Integer[][] predloha = new Integer[9][9];
39     static Integer[][] hraciPole = new Integer[9][9];
40
41
42     static int sec=0;
43     static int min=0;
44     static int hour=0;
45     static boolean status = false;
46     static Label lbTimer;
```

Pro snazší a přehlednější práci je většina hlavních proměnných „static“,
takže s nimi mohu přímo pracovat v různých metodách, kde je třeba pouze porovnat různé proměnné či je vzít a uložit.

Vlastnosti/vytvoření okna

```
private void initialize(){
    frmSudoku = new JFrame();
    frmSudoku.setTitle("Sudoku");
    frmSudoku.setBounds(100, 100, 450, 480);
    frmSudoku.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frmSudoku.setResizable(false);
    frmSudoku.getContentPane().setLayout(null);
    frmSudoku.setLocationRelativeTo(null);

    pBunky = new JPanel();
    pBunky.setBounds(10, 11, 414, 367);
    frmSudoku.getContentPane().add(pBunky);
    pBunky.setLayout(new GridLayout(9, 9, 0, 0));
```

Nastavení GUI - jeho šířka a výška, že se nemá dát zvětšovat či zmenšovat, že se má po otevření objevit uprostřed monitoru a po kliknutí na „křížek“ se má okno zavřít.

Vytvoření panelu pro hrací plochu 9x9.

Tlačítko

```
Button btNapoveda = new Button("Nápověda");
btNapoveda.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {

        Generovani.napoveda();

    }
});
btNapoveda.setBounds(110, 384, 94, 27);
frmSudoku.getContentPane().add(btNapoveda);
```

Zde je vytváření tlačítka Nápověda, jeho velikost, souřadnice a též to, že po stisknutí se spustí metoda napoveda().

Vytvoření prázdné hrací plochy 9x9

```
int poziceJTextField=0;
for (int i = 0; i < hraciPole.length; i++) {
    for (int j = 0; j < hraciPole.length; j++) {

        poleBunek[poziceJTextField] = new JTextField();

        PlainDocument doc = (PlainDocument) poleBunek[poziceJTextField].getDocument();
        doc.setDocumentFilter(new IntFilter());

        pBunky.add(poleBunek[poziceJTextField]);
        poleBunek[poziceJTextField].setColumns(1);
        poleBunek[poziceJTextField].setHorizontalAlignment(JTextField.CENTER);
        poleBunek[poziceJTextField].setBackground(new Color(255, 255, 255));
        if(j==2 || j==5) {
            if(i==2 || i==5) {
                poleBunek[poziceJTextField].setBorder(BorderFactory.createMatteBorder(1, 1, 4, 4, Color.gray));
            }else {
                poleBunek[poziceJTextField].setBorder(BorderFactory.createMatteBorder(1, 1, 1, 4, Color.gray));
            }
        }else if(i==2 || i==5){
            poleBunek[poziceJTextField].setBorder(BorderFactory.createMatteBorder(1, 1, 4, 1, Color.gray));
        }else {
            poleBunek[poziceJTextField].setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.gray));
        }
        poleBunek[poziceJTextField].setFont(new Font("Segoe UI", Font.BOLD, 28));


            poleBunek[poziceJTextField].setEditable(false);
            poleBunek[poziceJTextField].setForeground(Color.blue);


        poziceJTextField++;
    }
}
```

Zde se vytváří hrací pole 9x9 (81 buněk), kde jsou všechny buňky prázdné, ale needitovatelné. Zároveň se zde též vytváří šedé rámečky, aby se vizuálně oddělily kontejnery(3x3).

Omezení na pouhé čísla 1-9

```
class IntFilter extends DocumentFilter {

       public void insertString(FilterBypass fb, int offset, String string,

       private boolean povoleni(String text) {

           try {
              if(text.trim().isEmpty()) {
                  return true;
              } else {
                 Integer.parseInt(text);
                 if(text.length() == 1 && !text.equals("0")) {
                     return true;
                 } else {
                     return false;
                 }
              }
           } catch (NumberFormatException e) {
              return false;
           }
       }

       public void replace(FilterBypass fb, int offset, int length, String text,

       public void remove(FilterBypass fb, int offset, int length)
}
```

Zde je použit kód z internetu, který omezí uživateli do poleBunek zadávání na pouhé čísla od 0 do 10, což bylo pro mě nedokonalé, takže prostřední část je pozměněná, díky které se může použít backspace a nemůže se zadat 0.

Menu – statistiky

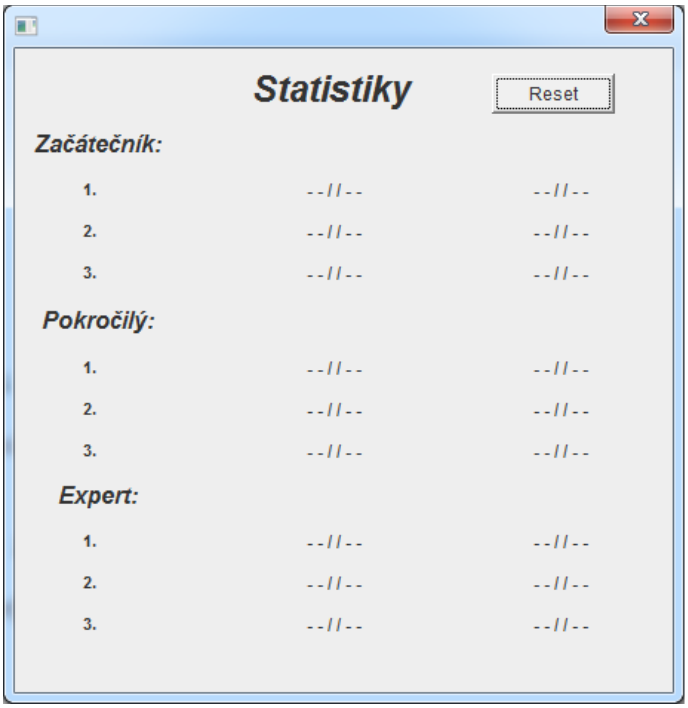
```
JMenuItem mntmStatistiky = new JMenuItem("Statistiky");
mntmStatistiky.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {

        Statistiky stat = new Statistiky();
        stat.setModal(true);
        stat.setVisible(true);

    }
});
mnHra.add(mntmStatistiky);
```

V této části jde vidět vytváření tlačítka „Statistiky“ jeho funkce, že po stisknutí se vytvoří proměnná stat (okno statistik) a následné vložení tlačítka do menu.

Statistiky.java



Statistiky je okno vytvořené po stisknutí tlačítka statistiky v GUI nebo po dohrání hry. Je vytvořeno z 1 tlačítka Reset a 31 „nálepek“ kde nálepky s „- - / / - -“ se přepisují. Do prvního přepisovatelného sloupce se vkládá po načtení statistik jméno daného hráče a k němu do druhého sloupce, se vkládá jeho čas.

Načítání statistik

```
public Statistiky() {

    Skore pole[] = new Skore[9];
    String casy[] = new String[9];
    String jmena[] = new String[9];

    pole = Stats.nacitani();

    for(int i=0; i<pole.length; i++) {
        if(pole[i]!=null) {
            jmena[i] = pole[i].getJmeno();

            String seconds="";
            String minutes="";
            String hours="";
            if(pole[i].getSec()<=9) {
                seconds="0"+pole[i].getSec();
            }else {
                seconds = Integer.toString(pole[i].getSec());
            }
            if(pole[i].getMin()<=9) {
                minutes="0"+pole[i].getMin() ;
            }else {
                minutes=Integer.toString(pole[i].getMin() );
            }
            if(pole[i].getHour()<=9) {
                hours="0"+pole[i].getHour();
            }else {
                hours=Integer.toString(pole[i].getHour());
            }
            casy[i] = hours + " : " + minutes + " : "+ seconds;
        }else {
            jmena[i]="- - / / - -";
            casy[i]="- - / / - -";
        }
    }
```

Po otevření statistik se vytvoří 3 pole, kde se do pole jménem pole načtou pomocí metody nacitani() statistiky. Pole se poté rozpulí na jména a časy. Čas(hod,min, sec) se předělá do typu String pro lehčí výpis.

„nálepky“

```
lbJmeno12 = new JLabel(jmena[1]);
lbJmeno12.setHorizontalTextPosition(SwingConstants.CENTER);
lbJmeno12.setHorizontalAlignment(SwingConstants.CENTER);
lbJmeno12.setBounds(159, 112, 112, 22);
contentPanel.add(lbJmeno12);

lbJmeno13 = new JLabel(jmena[2]);
lbJmeno13.setHorizontalTextPosition(SwingConstants.CENTER);
lbJmeno13.setHorizontalAlignment(SwingConstants.CENTER);
lbJmeno13.setBounds(159, 140, 112, 22);
contentPanel.add(lbJmeno13);

lbCas11 = new JLabel(casy[0]);
lbCas11.setHorizontalTextPosition(SwingConstants.CENTER);
lbCas11.setHorizontalAlignment(SwingConstants.CENTER);
lbCas11.setBounds(312, 84, 112, 22);
contentPanel.add(lbCas11);

lbCas12 = new JLabel(casy[1]);
lbCas12.setHorizontalTextPosition(SwingConstants.CENTER);
lbCas12.setHorizontalAlignment(SwingConstants.CENTER);
lbCas12.setBounds(312, 112, 112, 22);
contentPanel.add(lbCas12);
```

Zde jde vidět, že „nálepky“ mají absolutní rozvržení a že každá má svoje přidělení -jméno/čas

Reset tlačítko

```
Button btResetovat = new Button("Reset");
btResetovat.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        Stats.reset();
    }
});
btResetovat.setBounds(322, 17, 83, 27);
contentPanel.add(btResetovat);
```

Vytváření tlačítka Reset, jeho funkce a umístění.

Stats.java

Zde jsou metody spojené se statistikami, resetování statistik, jejich načítání a též jejich ukládání v případě rychlejšího dohrání bez použití nápovědy.

Reset()

```
public static void reset() {
    Skore pole[]= new Skore[9];
    String cesta = System.getenv("APPDATA");
    try {
        FileOutputStream fileOut = new FileOutputStream(cesta+"\\statistiky.obj");
        ObjectOutputStream objectOut = new ObjectOutputStream(fileOut);

        for (int o = 0; o < 9; o++) {
                objectOut.writeObject(pole[o]);
        }

        objectOut.close();

    } catch (Exception ex) {
        ex.printStackTrace();
    }

    String nic = "- - / / - -";

    Statistiky.lbJmeno11.setText(nic);
    Statistiky.lbJmeno12.setText(nic);
    Statistiky.lbJmeno13.setText(nic);
    Statistiky.lbJmeno21.setText(nic);
    Statistiky.lbJmeno22.setText(nic);
    Statistiky.lbJmeno23.setText(nic);
    Statistiky.lbJmeno31.setText(nic);
    Statistiky.lbJmeno32.setText(nic);
    Statistiky.lbJmeno33.setText(nic);

    Statistiky.lbCas11.setText(nic);
    Statistiky.lbCas12.setText(nic);
    Statistiky.lbCas13.setText(nic);
    Statistiky.lbCas21.setText(nic);
    Statistiky.lbCas22.setText(nic);
    Statistiky.lbCas23.setText(nic);
    Statistiky.lbCas31.setText(nic);
    Statistiky.lbCas32.setText(nic);
    Statistiky.lbCas33.setText(nic);

}
```

Metoda na resetování statistik. Na začátku se vytvoří prázdné pole typu Skore a pak se uloží na místo, kde jsou statistiky uloženy, takže se zapíše prázdné pole. Kvůli absolutnímu rozvržení „nálepek“ v statistikách se musí každá nálepka jednotlivě přepsat(aktualizovat).

Nacitani()

```
public static Skore[] nacitani() {

    Skore pole[] = new Skore[9];
    String cesta = System.getenv("APPDATA");
    try {

         FileInputStream fileIn = new FileInputStream(cesta + "\\statistiky.obj");
         ObjectInputStream objectIn = new ObjectInputStream(fileIn);

         for(int i=0; i<9; i++) {
             pole[i] = (Skore) objectIn.readObject();
         }

         objectIn.close();
    }catch(FileNotFoundException ex) {


    }catch (Exception ex) {
         ex.printStackTrace();
    }
    return pole;
}
```

V této metodě probíhá načítání statistik. Vytvoří se prázdné pole typu Skore do kterého se načtou statistiky, pokud existují. Pokud ne, tak metoda vrátí prázdné pole a statistiky se zobrazí prázdné, takže jakákoliv výhra bez použití nápovědy se uloží.

Vyhra()

```
public static void vyhra() {
    GUI.status=false;
    boolean hint = false;
    String cesta = System.getenv("APPDATA");

    int poziceBunky=0;

    for (int i = 0; i < 9; i++) {
        for (int j = 0; j < 9; j++) {
            if(GUI.poleBunek[poziceBunky].getForeground().equals(Color.GREEN)) {
                hint=true;
                break;
            }
            poziceBunky++;
        }
        if(hint==true) {
            break;
        }
    }

    if(hint==false) {
        Skore porovnavani[] = Stats.nacitani();

        if(GUI.rdbtZacatecnik.isSelected()) {
            for (int i = 0; i < 3; i++) {
                if(porovnavani[i]==null) {
                    UIManager.put("OptionPane.cancelButtonText", "zrušit");
                    String jmeno="";
                    jmeno = jmeno + JOptionPane.showInputDialog(null, "Zadejte jméno",
                    jmeno = jmeno.trim();
                    if(jmeno.length()==0 || jmeno.equals("null")) {
                        jmeno = "Hráč";
                    }
                    porovnavani[i]=new Skore(jmeno, GUI.hour, GUI.min, GUI.sec);
```

Tato metoda se spustí tehdy, pokud je stisknuto tlačítko Nápověda nebo Kontrola v GUI a je správně vyplněné pole.

V této metodě jsou podstatné 3 kroky.

1. Na začátku metody se pozastaví časovač – > status = false
2. Vytvoří se proměnná hint a pomocí cyklů se zjišťuje, jestli byla použita nápověda. Pokud ano, ukáže se výherní okno ale bez uložení do statistik.

   Pokud nebyla použita, tak dle zaškrtnuté obtížnosti se dále hodnotí, jestli byl uživatel lepší než lidé ve statisticích.
3. Pokud je uživatel nejlepší ze statistik, tak se statistiky přepíší následovně a pak se statistiky uloží.

```
if(i==0) {
    porovnavani[2]=porovnavani[1];
    porovnavani[1]=porovnavani[0];
    porovnavani[0]=new Skore(jmeno, GUI.hour, GUI.min, GUI.sec);
} else if(i==1){
    porovnavani[2]=porovnavani[1];
    porovnavani[1]=new Skore(jmeno, GUI.hour, GUI.min, GUI.sec);
} else if(i==2){
    porovnavani[2]=new Skore(jmeno, GUI.hour, GUI.min, GUI.sec);
}

try {
    FileOutputStream fileOut = new FileOutputStream(cesta+"\\statistiky.obj");
    ObjectOutputStream objectOut = new ObjectOutputStream(fileOut);

    for (int o = 0; o < 9; o++) {
            objectOut.writeObject(porovnavani[o]);
    }

    objectOut.close();

} catch (Exception ex) {
    ex.printStackTrace();
}
break;
```

Toto je přepisování statistik pro začátečníky, kde index 0 je nejlepší a index 2 je třetí nejlepší. A dále následuje uložení statistik.

Generování pole

```
public static Integer[][] generatorPole() {
    Random r=new Random();
    Integer[][] hra=new Integer[9][9];

    for(int radek=0;radek<9;radek++){
        int pocetChyb=0;
        for(int sloupec=0;sloupec<9;sloupec++){

            //generovaní pole čísel 1-9
            Integer poleCisel[]=new Integer[9];
            for (int mozneCislo = 0; mozneCislo < poleCisel.length; mozneCislo++) {
                poleCisel[mozneCislo]=mozneCislo+1;
            }


            // generování čísla
            while(true) {
                boolean obsazeno=false;
                pocetChyb=0;
                //počítání, kolik už není možno využít čísel
                for (int zkouska = 0; zkouska < poleCisel.length; zkouska++) {
                    if(poleCisel[zkouska]==null) {
                        pocetChyb=pocetChyb+1;
                    }
                }

                //pokud nebude možné vložit žádné číslo, jde se od druhého řádku od znova
                if(pocetChyb==9) {

                    for (int i = 1; i < poleCisel.length; i++) {
                        for (int j = 0; j < poleCisel.length; j++) {
                            hra[i][j]=null;
                        }
                    }
                    radek=0;
                    sloupec=8;
                    break;
                }
```

.

.

cca dalších 190 řádků

.

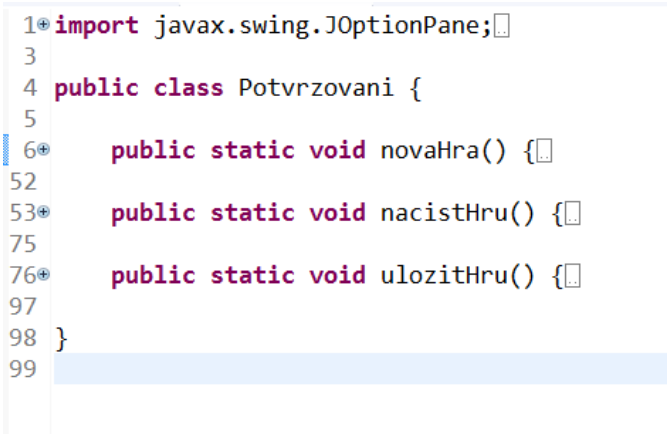
.

V mém generátoru na sudoku je důležité to, že se vytvoří 2rozměrné pole a zbytek se řeší ve vnořeném cyklu for.

Myšlenka byla taková → pole 9x9, které jsem si v hlavě rozdělil na 9 kontejnerů 3x3 a zároveň jsem si to pole 9x9 rozdělil na řádky a sloupce, dle pravidel sudoku.

Takže se první vytvoří první řádka, kde se vygenerují čísla od 1 – 9(lehčí část). Když se dostaneme na druhý řádek, tak se z jedné podmínky(naplnit pouze řádek) stanou 3 podmínky, kde se musí dávat pozor na sloupec, řádek i kontejner (3x3).
V každé buňce může být pouze číslo od 1 do 9. Takže při každé nové pozici se vytvoří pole možných čísel(1-9). Kde se pak spustí cyklus, který kontroluje všechny podmínky a vymazává postupně z pole možných čísel (1- 9)čísla, která už se použít nemohou. Když cyklus skončí s vymazáváním pole možných čísel, a nějaká čísla zůstala, tak se zkusí použít a zároveň se to použité číslo smaže z pole možných čísel a takhle to jde dál, dokud se nenaplní celé herní pole, nebo dokud nenastane situace, kdy pole možných čísel nebude mít žádné volné číslo k použití, tak se pole sudoku 9x9 celé vymaže kromě první řádky, která není potřeba smazat.

Potvrzovani.java



Pro ujištění jsou zde metody, které zajišťují větší logiku tlačítek. Např. metoda nacistHru()

```
public static void nacistHru() {

    if(GUI.predloha[0][0]!=null) {
        UIManager.put("OptionPane.yesButtonText", "Ano");
        UIManager.put("OptionPane.noButtonText", "Ne");
        int moznost = JOptionPane.showConfirmDialog(null, "Opravdu chcete načíst posledně uloženou hru?", "",
                                                    JOptionPane.YES_NO_OPTION, JOptionPane.PLAIN_MESSAGE); //

        if(moznost==0) {

            Generovani.nacitani();

        }

    }else {

        Generovani.nacitani();

    }


}
```

ptá se, jestli predloha[0][0] není prázdná, to znamená, jestli se po spuštění aplikace už spustila nějaká hra, pokud ne, tak není třeba se ptát na načítání a rovnou se hra načte.

Pokud ale hra zahájená už byla a zmáčkne se tlačítko Načíst hru, tak se první aplikace zeptá, jestli opravdu chceme načíst hru, pokud ano - hra se načte, pokud ne - hra se nenačte.

Závěr:

Při vytváření této ročníkové práce jsem se snažil co nejvíce využít zkušenosti ze školy, ale mnohé věci jsem se musel najít a naučit pomocí internetu. Na základní porozumění GUI mi velmi pomohl kanál na youtube jménem Daniel Grissom, je to nejspíše nějaký učitel, který zdokumentoval několik svých vyučujících hodin, čemuž jsem opravdu vděčný.

Dle mého byla nejtěžší část časovač, protože mi Timery zrovna moc nefungovaly, takže jsem byl přinucen k vytvoření Thread a vytvořit si vlastní časovač. Threads nabízí asi 3 metody na ukončení, ale u každé to píše „deprecated“, takže ty způsoby ukončení nefungují, takže zrovna moc nechápu, proč ty metody neodstraní.

Na svoji práci jsem do jisté míry hrdý, vložil jsem do toho spoustu času, ale rozhodně se najde něco, co by se dalo vylepšit, třeba tu „logiku“ aplikace.

Použité zdroje

Omezení pouze na čísla:
https://stackoverflow.com/questions/11093326/restricting-jtextfield-input-to-integers

Základy GUI:
https://www.youtube.com/channel/UCBqFIGiQ5sKA5M8W9GfF1QQ/videos