Poročilo:

O izvedeni vaji:

21\_lav03\_vaja

Laboratorijske vaje

Vaja 21

Maj Mohar, G3A

Ljubljana, april 2025

***Naloga 1.***

Deklarirajterazred AstralnoTelo*,* ki vsebuje lastnost ‘naziv' in konstruktor.

Iz njega izpeljite razred Zvezda, ki deduje lastnost naziv (pazite na opredelitev kvalifikatorja za dostopne pravice v razredu AstralnoTelo) tako, da je del razreda Zvezda. V razredu Zvezda definirajte še zaščitene(protected) lastnosti: nazivZvezde, radij in oddaljenostOdSonca. Zvezda naj vsebuje tudi konstruktor (lahko tudi več konstruktorjev), ter primerjalne metode:

* razredna metoda vrniSoncuBlizjoZvezdo/2 dobi kot argument 2 zvezdi in vrne tisto, ki je bližje soncu;
* razredna metoda vrniManjsoRazdaljo/2 vrne razdaljo tiste z argumentum podane zvezde, katere razdalja do Sonca je manjša;
* objektno metodo manjsaOdRazdalj/1 primerja oddaljenost zvezde in z argumentum podane in vrne manjso od obeh.

Razred naj vsebuje še metodo, ki zna sešteti in tudi sešteje dve Zvezdi - Zvezdi seštejemo tako, da radiju večje zvezde dodamo 10% radija manjše; oddaljenost postane oddaljenost bližje zvezde; naziv nove zvezde pa se sestavi iz prvega znaka večje in prvega znaka manjše zvezde. **Če sta radija obeh zvezd enaka, zvezdi ne moremo in ju NE seštejemo!**

V zagonski metodi demonstracijskega razreda (lahko je to tudi razred Zvezda) ustvarite 5 zvezd in jih inicializirajte (vpiši podatke). Programsko poiščite zvezdo **najbližjo** Soncu in jo izpišite. Nato prvi definirani zvezdi prištejte vse preostale zvezde, ki jih lahko (različna radija) in izpišite končni radij novonastale zvezde.

Izpis zvezde naj se izvede kot izpis n-členika z nazivom zvezde na začetku, ki mu sledijo vsi preostali atributi oz. lastnosti zvezde.

Koda je v prilogi.

***Naloga 2.***

Zapišite definicijo razreda Racunalnik, ki vsebuje naslednje lastnosti: vrsto CPE, velikost trdega diska, velikost pomnilnika ter cena. Dodajte konstruktor z inicializacijo lasnosti in metode za vnos in vračanje njegovih lastnosti (setterje in getterje).

Iz razreda Racunalnik izpeljite razred Prenosnik, z dodatnimi lastnostmi velikost zaslona ter trajanje(čas avtonomije) baterije. Razred Prenosnik naj vsebuje konstruktor, setterje in getterje ter metodo, ki vrne velikosti zaslona v cm (vrednost lastnosti velikost zaslona je zapisana v palcih).

Zagonska metoda razreda Računalnik naj služi kot ovojnica za preskus definicij in delovanja:

V njej definirajte, skreirajte in inicializirajte dva prenosnika (objekta razreda Prenosnik), ter izvedite demonstracijo rabo vseh metod. Na koncu izpiši, kateri izmed obeh prenosnikov ima boljše razmerje med trajanjem baterije in ceno.

Po potrebi v definicijo razredov lahko dodate še kakšno metodo.

Koda je v prilogi

***Naloga 3.***

Ustvariti bi želeli YesNo gumb za JavaFX aplikacijo. Objekt vrste YesNo pri postavitvi na sceno prikazuje napis Yes, klik na gumbu zamenja vsebino napisa na No, vsak naslednji ponovno zamenja Yes z No oz. No z Yes.

Obstoječi gumb vrste javafx.scene.control.Button je funkcionalno poln gumb, le da ima zgolj eno fiksno lastnost 'text' za prikazovanje vrednosti na gumbu.

Pripravimo test dostopa do lastnosti 'text' gumba ob kliku:

Uporabimo obstoječo JavaFX aplikacijo, ki jo lahko naredimo z čarovnikom v BlueJ. Poiščemo

servisno metodo, ki se izvede ob kliku na gumb in kot zadnjo vrstico dodamo:

Button bb = (Button)event.getSource(); bb.setText("trilili");

1. spišite mehanizem preklapljanja med Yes in No. Gumb naj bo vrste Button.
2. Na osnovi Button kreirajte razred YesNo Button. Konstruktor za novi gumb prejme dve besedi (Yes, No) in pri klikanju naj preklaplja med njima (preklaplja med poljubnima podanima dvema besedama). Dodajte še oba 'fall back' konstruktorja; če gumb naredite zgolj z eno besedo, je druga vedno 'N-e', če ga ustvariite brez, pa je prva 'Da', druga 'Ne'

Koda je v prilogi.

public class AstralnoTelo{

protected String naziv;

public AstralnoTelo(){

naziv = "";

}

public AstralnoTelo(String naziv){

this.naziv = naziv;

}

}

class Zvezda extends AstralnoTelo{

protected String nazivZvezde;

protected double radij, oddaljenostOdSonca;

public Zvezda(){

super();

radij = **0**;

oddaljenostOdSonca=**0**;

}

public Zvezda(String naziv, double radij, double oddaljenostOdSonca){

super(naziv);

this.nazivZvezde = naziv;

this.radij = radij;

this.oddaljenostOdSonca = oddaljenostOdSonca;

}

public String getNaziv(){

**return** nazivZvezde;

}

public double getRadij(){

**return** radij;

}

public double getOddaljenost(){

**return** oddaljenostOdSonca;

}

public Zvezda vrniSoncuBlizjoZvezdo(Zvezda a){

**if** (this.oddaljenostOdSonca> a.getOddaljenost()) {

**return** this;

}

**else** **return** a;

}

public double vrniManjsoRazdaljo(Zvezda a){

**if**(this.radij >a.getRadij())

**return** this.radij;

**else** **return** a.getRadij();

}

public double manjsaOdRazdalij(Zvezda a){

**if**(this.radij <a.getRadij())

**return** this.radij;

**else** **return** a.getRadij();

}

public void sestej(Zvezda a){

**if** (this.radij == a.getRadij()){

System.out.println("Ni možno sešteti zvezd.");

}**else**{

double nRadij = **0**;

**if**(a.getRadij()>this.radij)

nRadij= a.getRadij() + this.radij\***0.1**;

**else**

nRadij = this.radij + a.getRadij()\***0.1**;

this.radij = nRadij;

this.oddaljenostOdSonca = Math.max(this.oddaljenostOdSonca, a.getOddaljenost());

this.nazivZvezde = a.getNaziv().charAt(**0**) + ""+this.nazivZvezde.charAt(**0**);

}

}

public String toString() {

**return** nazivZvezde + ": radij = " + radij + ", oddaljenost od Sonca = " + oddaljenostOdSonca;

}

public static void main(String[] args) {

Zvezda[] zvezde = **new** Zvezda[**5**];

zvezde[**0**] = **new** Zvezda("Sirius", **1.7**, **8.6**);

zvezde[**1**] = **new** Zvezda("Betelgeza", **887.0**, **548**);

zvezde[**2**] = **new** Zvezda("Proxima", **0.1542**, **4.24**);

zvezde[**3**] = **new** Zvezda("Rigel", **78.9**, **860**);

zvezde[**4**] = **new** Zvezda("Vega", **2.36**, **25.04**);

Zvezda najblizja = zvezde[**0**];

**for** (int i = **1**; i < zvezde.length; i++) {

najblizja = zvezde[i].vrniSoncuBlizjoZvezdo(najblizja);

}

System.out.println("Najblizja soncu: "+ najblizja.nazivZvezde);

Zvezda novaZvezda = **new** Zvezda(zvezde[**0**].nazivZvezde, zvezde[**0**].radij, zvezde[**0**].oddaljenostOdSonca);

**for** (int i = **1**; i < zvezde.length; i++) {

**if** (zvezde[i].getRadij() != novaZvezda.getRadij()) {

novaZvezda.sestej(zvezde[i]);

}

}

System.out.println("Sestete zvezde: "+novaZvezda);

}

}

public class Racunalnik{

String vrstaCPE;

double velikostDiska;

double velikostPomnilnika;

double cena;

public Racunalnik(){

vrstaCPE = "Ryzen 7 AMD";

velikostPomnilnika = **16**;

velikostDiska = **512**;

cena = **1099**;

}

public Racunalnik(String vrstaCPE, double velikostDiska, double velikostPomnilnika, double cena){

this.vrstaCPE = vrstaCPE;

this.velikostDiska = velikostDiska;

this.velikostPomnilnika = velikostPomnilnika;

this.cena = cena;

}

public String getVrstaCPE(){

**return** vrstaCPE;

}

public double getVelikostPomnilnika(){

**return** velikostPomnilnika;

}

public double getVelikostDiska(){

**return** velikostDiska;

}

public double getCena(){

**return** cena;

}

public void setVrstaCPE(String vrstaCPE){

this.vrstaCPE = vrstaCPE;

}

public void setVelikostPomnilnika(double velikostPomnilnika){

this.velikostPomnilnika = velikostPomnilnika;

}

public void setVelikostDiska(double velikostDiska){

this.velikostDiska = velikostDiska;

}

public void setCena(double cena){

this.cena = cena;

}

public static void main(String args[]){

Prenosnik a = **new** Prenosnik("Intel i9 1300k", **512**, **16**, **999**, **30**, **4**);

Prenosnik b = **new** Prenosnik("Ryzen 7 AMD", **1024**, **32**, **1299**, **30**, **5**);

System.out.println(a);

System.out.println(b);

a.kateriJeBoljsi(b);

}

}

class Prenosnik extends Racunalnik{

double avtonomija;

int velikostZaslona;

public Prenosnik(){

super();

}

public Prenosnik(String vrstaCPE, double velikostDiska, double velikostPomnilnika, double cena, int velikostZaslona){

super(vrstaCPE, velikostDiska, velikostPomnilnika, cena);

avtonomija = **0**;

this.velikostZaslona = velikostZaslona;

}

public Prenosnik(String vrstaCPE, double velikostDiska, double velikostPomnilnika, double cena, int velikostZaslona, int avtonomija){

super(vrstaCPE, velikostDiska, velikostPomnilnika, cena);

this.avtonomija = avtonomija;

this.velikostZaslona = velikostZaslona;

}

public double getAvtonomija(){

**return** avtonomija;

}

public int getVelikostZaslona(){

**return** velikostZaslona;

}

public void setAvtonomija(double avtonomija) {

this.avtonomija = avtonomija;

}

public void setVelikostZaslona(int velikostZaslona) {

this.velikostZaslona = velikostZaslona;

}

public String toString(){

**return** "Prenosnik "+this.getVrstaCPE()+" z "+this.getVelikostDiska()+"gb spomina, "+this.getVelikostPomnilnika()+"gb pomnilnika, "+this.getVelikostZaslona()+"cm zaslonom, "+this.getAvtonomija()+"h dolgo baterijo in ceno "+this.getCena()+" evrov.";

}

public void kateriJeBoljsi(Prenosnik a){

**if**(this.avtonomija/cena > a.getAvtonomija()/a.getCena() )

System.out.println("Boljsi je: "+ this);

**else**

System.out.println("Boljsi je: "+ a);

}

}

**import** javafx.application.Application;

**import** javafx.event.ActionEvent;

**import** javafx.scene.Scene;

**import** javafx.scene.control.Button;

**import** javafx.scene.control.Label;

**import** javafx.scene.layout.GridPane;

**import** javafx.geometry.Insets;

**import** javafx.stage.Stage;

/\*\*

\* Opis:

\* JavaFX aplikacija števec

\*

\* **@author** Maj

\* **@version** 14/04/2025

\*/

public class Gumb extends Application {

private int count =**0**;

private Button myButton = **new** Button("Yes");

public static void main(String[] args) {

launch(args);

}

@Override

public void start(Stage stage) {

GridPane pane = **new** GridPane();

pane.setPadding(**new** Insets(**10**));

pane.setVgap(**10**);

pane.setHgap(**10**);

myButton.setOnAction(this**::**buttonClick);

pane.add(myButton, **0**, **0**);

Scene scene = **new** Scene(pane, **300**, **100**);

stage.setTitle("Gumb");

stage.setScene(scene);

stage.show();

}

/\*\*

\* Metoda za upravljanje s klikom gumba.

\* Poveča števec in posodobi oznako.

\*/

private void buttonClick(ActionEvent event) {

count++;

**if**(count%**2**==**1**)

myButton.setText("no");

**else**

myButton.setText("yes");

}

}

**import** javafx.application.Application;

**import** javafx.event.ActionEvent;

**import** javafx.scene.Scene;

**import** javafx.scene.control.Button;

**import** javafx.scene.control.Label;

**import** javafx.scene.layout.GridPane;

**import** javafx.geometry.Insets;

**import** javafx.stage.Stage;

/\*\*

\* Opis:

\* JavaFX aplikacija števec

\*

\* **@author** Maj

\* **@version** 14/04/2025

\*/

public class Gumb extends Application {

private int count =**0**;

private Button myButton = **new** Button("Yes");

public static void main(String[] args) {

launch(args);

}

@Override

public void start(Stage stage) {

GridPane pane = **new** GridPane();

pane.setPadding(**new** Insets(**10**));

pane.setVgap(**10**);

pane.setHgap(**10**);

myButton.setOnAction(this**::**buttonClick);

pane.add(myButton, **0**, **0**);

Scene scene = **new** Scene(pane, **300**, **100**);

stage.setTitle("Gumb");

stage.setScene(scene);

stage.show();

}

/\*\*

\* Metoda za upravljanje s klikom gumba.

\* Poveča števec in posodobi oznako.

\*/

private void buttonClick(ActionEvent event) {

count++;

**if**(count%**2**==**1**)

myButton.setText("no");

**else**

myButton.setText("yes");

}

}