JAVA programavimo kalba

Objektinis programavimas (OOP – object oriented programming)

Objekto klasė (object class)

Klasė - tai aprašas, nusakantis tam tikro tipo objektų duomenų struktūrą ir manipuliavimo jais taisykles (elgesį). Pavyzdžiui, jei aprašome klasę „Žmogus“, galėsime kurti šios klasės objektus/egzempliorius (instance) kiekvienam konkrečiam asmeniui. Šie objektai turės bendrą duomenų struktūrą, tačiau skirtingas reikšmes (skirsis vardas, pavardė).

Klasė (class)

```
public class Zmogus {
        String vardas;
        String pavarde;
}
```

Čia kintamieji vardas ir pavarde vadinami klasės Zmogus laukais. Jei tiksliau - tai vadinami klasės egzemplioriaus (instance) laukais, nes jie sukuriami kiekvienam duotos klasės objekto egzemplioriui atskirai.

Objekto sukūrimas (object create instance)

```
Zmogus zmogus1 = new Zmogus();
zmogus1.vardas = "Karolis XV";
System.out.println(zmogus1.vardas);
Zmogus zmogus2 = new Zmogus();
zmogus2.vardas = "Ieva";
zmogus2.pavarde = "Ievaitė";
System.out.println(zmogus2.vardas + ' ' + zmogus2.pavarde);
```

Čia zmogus1 ir zmogus2 kintamųjų reikšmės vadinamos klasės Zmogus objektais ar egzemplioriais (instance)

Objekto konstruktorius (object constructor)

- Konstruktorius – specialus klasės metodas, naudojamas naujo objekto sukūrimui.
- Gali būti ir keli konstruktoriai tik reikia kad skirtųsi jų parametrai.

```
public class Zmogus {
        String vardas;
        String pavarde;
Zmogus(String v, String p) {
        vardas = v;
        pavarde = p;
}

Zmogus(String v) {
        vardas = v;
}
```

Objekto konstruktorius (object constructor)

- Jei konstruktoriaus parametro vardas sutampa su klasės lauko vardu, tai norint pabrėžti, kad dirbama su lauku reikia prie lauko vardo naudoti this.

```
class Zmogus {
        String vardas;
        String pavarde;
Zmogus(String vardas, String pavarde) {
        this.vardas = vardas;
        this.pavarde = pavarde;
}
Zmogus(String vardas) {
        this.vardas = vardas;
}
```

Objekto sukūrimas (object create instance)

- Jei turime kelis konstruktorius, tai galime objekto egzempliorių kurti skirtingais būdais:

```
Zmogus zmogus1 = new Zmogus("Jonas Paulius II");
Zmogus zmogus2 = new Zmogus("Ieva", "Ievaitė");
```

Pratimas nr.1

- Sukurkite masyvą, kuriame būtų bent trys Zmogus klasės objektai ir atspausdinkite visus sukurtus žmones.

Paveldimumas (inheritance)

- Klasė gali išplėsti kitą klasę, t.y. paveldėti vienas savybes ir pridėti savo kitas. Toks reiškinys vadinamas klasės paveldimumu (inheritance), pvz:

```
public class Mokinys extends Zmogus {
        int klase;
        int[] trimestras;
}
```

- Zmogus klasė Mokinys klasės atžvilgiu vadinama tėvine (parent) arba bazine (base) klase, o Mokinys klasė vadinama dukterine (child) arba paveldinčia klase klasei Zmogus.

Paveldimumas (inheritance)

Dukterinės klasės konstruktoriuje galima kviesti tėvinės klasės konstruktorių arba tos pačios klasės kitą konstruktorių su kitokiais argumentais:

```
public class Mokinys extends Zmogus {
        int klase; // pvz 6 ar 9
        int[] trimestras; // masyvas mokinio trimestro pažymių

public Mokinys(String vardas, String pavarde, int klase) {
        super(vardas, pavarde);
        this.klase = klase;
}

public Mokinys(String vardas, String pavarde, int klase, int[] trimestras) {
        this(vardas, pavarde, klase);
        this.trimestras = trimestras;
}

}
```

Paveldimumas (inheritance)

- Yra klasė, kuri yra tėvinė klasė visoms Java klasėms – tai klasė vardu Object
- Mums nereikia nurodyti, kad mūsų klasė praplečia klasę Object. Skaitoma, kad yra nurodyta pagal nutylėjimą, t.y.:

class Zmogus { … } atitinka:

class Zmogus **extends Object** { … }

Pratimas Nr.2

```
Mokinys mokinys1 = new Mokinys("Jonas", "Jonaitis", 3);
mokinys1.trimestras = new int[] {8, 9, 8};

Mokinys mokinys2 = new Mokinys("Ona", "Onaitė", 2, new int[] {9, 10, 10});
```

Pratimas Nr.3

Tarkime Mokinys klasės trimestras laukas turi kažkokias reikšmes (masyvą sveikų skaičių), pvz.: {8, 9, 8, 10}

Parašykite klasei Mokinys metodą kuris grąžintų trimestro vidurkį kaip skaičių su kableliu, t.y. duotu atveju grąžintų 8.75

Atspausdinkite mokinių vardus, pavardes ir vidurkius.

Java paketai (packages)

Problema su klasės pavadinimais - kaip sugalvoti klasėms unikalius pavadinimus

- Problema su projekto failais - kadangi kiekviena java klasė talpinama atskirame faile, tai su laiku tų failų pasidaro labai daug.
- Sprendimas - paketai (packages). Paketas yra kaip failų katalogai kompiuteryje. Kiekvienas katalogas gali turėti subkatalogus. Ir juose gali būti talpinami failai (klasės) net ir tais pačiais vardais kaip ir kitame kataloge.
- Pilnas klasės vardas susideda iš paketo pavadinimo ir klasės pavadinimo.
- Jų kombinacija turi būti unikali.
- Paprastai paketas atitinka katalogą kompiuterio failų sistemoje.
- Labai dažnai paketo pavadinimas prasideda kompanijos atvirkščiu domeno pavadinimu po kurio seka programos/projekto pavadinimas.
- Paketas nurodomas su package direktyva ir ji turi būti pati pirma faile (neskaitant komentarų), pvz.:

```
package lt.baltictalents.demo1;

public class Zmogus { … }
```

Java paketai (packages)

- Jei kitoje vietoje (faile/klasėje) norime panaudoti jau aprašytą kažkokiame pakete klasę tai galima tai padaryti dviem būdais:
- Naudoti pilną klasės pavadinimą, t.y. kartu su jos paketu:
- lt.baltictalents.demo1.Zmogus z = new lt.baltictalents.demo1.Zmogus(...)
- Viršuje failo (iš karto žemiau paketo aprašo) su import direktyva nurodant koks pilnas klasės pavadinimas ir tada klasėje galima naudoti sutrumpintą importuotos klasės pavadinimą:
- **import** lt.baltictalents.demo1.Zmogus;
- …
- Zmogus z = new Zmogus(…)

Klasės pasiekiamumo kontrolė (access)

- Klasės pasiekiamumas kontroliuojamas nurodant atitinkamus pasiekiamumo atributus
- public (vieša) - klasė pasiekiama iš visur
- - nieko nenurodyta - klasė prieinama tik iš einamojo paketo

Klasės laukų ir metodų pasiekiamumo kontrolė (access)

- Klasės laukų ir metodų pasiekiamumas kontroliuojamas nurodant atitinkamus pasiekiamumo atributus:
- public - laukai ir metodai pasiekiami iš visur
- private - laukai ir metodai pasiekiami tik iš jų klasės
- protected - laukai ir metodai pasiekiami tik tame pačiame pakete esančių klasių, o taip pat iš duotos klasės dukterinių klasių.
- - nieko nenurodyta - laukai ir metodai pasiekiami tik iš einamojo paketo

Pratimas Nr.4

- Pataisykite Zmogus klasę taip, kad žmogaus vardo ir pavardės po sukūrimo nebūtų galima keisti.

Geteriai (getter) / seteriai (setter)

- Paprastai klasės laukai yra privatūs, o jų reikšmės nustatomos ir nuskaitomos metodų vadinamų geteriais ir seteriais pagalba:

```
public class Zmogus {
        private String name;

        public String getName() {
                return name;
        }
public void setName(String name) {
        this.name = name;
}
}
```

Geteriai (getter) / seteriai (setter)

- Geteriai ir seteriai pavadinimai paprastai sudaryti iš lauko pavadinimo priekyje pridedant **set** ir **get** ir lauko pavadinimą užrašant iš didžiosios raidės. Galima išimtis, kai laukas yra boolean tipo, tai tada tokio lauko geteris pradedamas žodeliu **is**.
- Geteris neturi parametrų ir gražiną tokio pat tipo reikšmę kaip atitinkamas laukas, o seteris turi vieną parametrą tokio pat tipo kaip atitinkamas laukas ir negrąžina jokios reikmės, t.y. **void**, pvz.:

- private int age;
- private boolean live;

- public int **getAge**() { return age; }
- publis void **setAge**(int age) { this.age = age; }
- public boolean **isLive**() { return live; }
- public void **setLive**(boolean live) { this.live = live; }

Inkapsuliacija (Encapsulation)

Pagal objektiškai orientuoto programavimo principus tiesioginė prieiga prie objekto laukų turi būti kiek galima labiau apribota, o visos duomenų keitimo ir nuskaitymo operacijos būtų atliekamos iškviečiant jo metodus. Šis principas vadinamas **inkapsuliacija (encapsulation)**.

Užduotys

1. Tarkime turime masyvą objektų Mokinys. Reikia atspausdinti mokinių vardus ir pavardes surūšiuotus pagal klases ir pagal pavardes bei vardus. Pastaba: nustatykite keliems iš vienos klasės mokiniams tas pačias pavardes bet skirtingus vardus, kad patikrinti ar gerai rūšiuoja. *string tipo kintamųjų palyginimui naudokite metodą pvz: vardas.compareToIgnoreCase();