01. Nasljeđivanje. Polimorfizam

Klasa Object

- osnovna klasa za sve dart objekte osim null
- korijen hijerarhije Dart klasa koje nisu null
    - Svaka druga klasa koja nije null je podklasa klase Object i sadrži metode iz Object klase

Zašto nasljeđivanje?

- Želimo napraviti novu klasu, a već postoji klasa koja uključuje nešto što trebamo (kôd, članske varijable, metode...)?
- Umjesto pisanja klase ispočetka, nasljeđivanjem imamo mogućnost korištenja već postojećih metoda
- Novu klasu „obogatimo” svojstvima koja nedostaju - specifičnim samo za tu klasu

Nadjačavanje

- Nadjačavanje – engl. *override*
- Nadjačavanje definira novo ponašanje naslijeđene metode
    - Metoda mora imati isti potpis (naziv, povratni tip, broj i tipovi argumenata)
    - Napomena: novo ponašanje se primjenjuje na klasu u kojoj je došlo do nadjačavanja i na klase ispod u hijerarhiji

Bazna i izvedena klasa

- **Bazna** klasa (osnovna klasa, natklasa, klasa roditelj) je klasa iz koje se izvodi neka druga klasa
  - engl. ***base class, superclass***
- Klasa koja nasljeđuje neku klasu naziva se **izvedena** klasa (potklasa, klasa dijete)
- engl. ***derived class, subclass***
- Izvedena klasa je specijalizacija bazne klase
- Bazna klasa je generalizacija svojih izvedenih klasa

Nasljeđivanje

- Izvedena klasa se sastoji od vlastitih članova i članova bazne klase
- ne može pristupiti privatnim članovima bazne klase
- Konstruktori se ne nasljeđuju, ali se mogu pozivati iz izvedene klase
- U Javi sve klase osim klase *Object* imaju jednu i samo jednu direktnu klasu roditelja.
- Ako to nije neka eksplicitno navedena klasa, onda je to klasa *Object.*

Polimorfizam (engl. Polymorphism)

- „poly“ znači mnogo, „morph“ znači preoblikovanje u različite oblike
  - zajedno znače stvatanje mnogih oblika ili konfiguracija.
- primjer u stvarnom svijetu
  - recimo da kupujemo novu značajku automobila
  - nova značajka je omogućavanje samostalne vožnje
  - tada umjesto osnovne značajke vožnje koja zahtjeva ljudsku podršku, ažuriramo funkcionalnost da se automobil može voziti sam, bez ljudske podrške
- sažetak: Polimorfizam je ažuriranje ili modificiranje značajke,funkcije ili implementacije koje već postoje u roditeljskoj klasi

Zadaci za vježbu

1. Napravi klasu `Dessert` koja ima sljedeće atribute: `name` (String), `weight` (double) i `calories` (int). Dodaj klasi konstruktor koji prima kao parametre vrijednosti za sva 3 atributa. Napravi `get` i set metode za svaki atribut, te nadjačaj metodu `toString`. Napišite i metodu `getDessertType` koja nema argumenata a vraća string „dessert“.
2. Napravi klase `Cake` i `IceCream` koje nasljeđuju `Dessert`. Kolač, uz sve atribute iz klase `Dessert` ima i atribute `containsGluten` (boolean) i `cakeType` (String, može biti „birthday“, „wedding“, „regular“ i sl.). Sladoled ima dodatne atribute `flavour` (String) i `color` (String). Napravi `get` i set metode za svaki atribut, kao i metodu `toString` koja vraća sve što vraća i metoda `toString` iz Klase `Dessert`, a dodatno još i atribute specifične za izvedenu klasu. Napiši metodu `getDessertType` u svakoj od izvedenih klasa, koja će za sladoled vratiti tekst „ice cream“, a za tortu vrijednost atributa `cakeType` i tekst „ cake“. Napiši glavni programs kojim ćete testirati sve zadane funkcionalnosti.
3. Napravi klasu `Person` koja opisuje neku osobu. `Person` sadrži atribute `name` (String) `surname` (String), `age` (int). Napiši konstruktor, get i set metode, te metode `toString` i`equals` (dvije osobe su jednake ako imaju isto ime i prezime te broj godina).
4. Napravi klase `Student` i `Teacher` koje nasljeđuju klasu `Person`, `Student` sadrži atribut studentId (String) i academicYear (int), a `Teacher` sadrži atribute email (String), subject (String) i salary (double). Napiši konstruktore za sve parametre, get i set metode, te metode toString i equals (dva studenta su jednaka ako imaju isti studentId, neovisno o ostalim podacima, a dva nastavnika su jednaka ako imaju isti email, neovisno o ostalim podacima). Dodatno, u klasi `Teacher` napiši metodu `increaseSalary` koja ne vraća ništa, a prima jedan argument tipa int (koji predstavlja postotak). Metoda treba povećati plaću nastavnika za zadani postotak. Također, napiši i statičku metodu `increaseTeachersSalaries` koja prima dva argumenta. Prvi je argument tipa int (koji predstavlja postotak), a drugi je lista učitelja (`Teacher`) kojima jepotrebno povećati plaću za zadani postotak.
5. Napiši glavni program u kojem ćeš kreirati listu od 5 osoba (Person) i u nju staviti 3 nastavnika i dva studenta. Nakon toga program treba u petlji ispisati ime i prezime svake osobe te na kraju petlje prosječnu plaću svih nastavnika koji se pojavljuju u polju.

   Jedan primjer ispisa sa bi trebao izgledati ovako (možeš koristiti bilo koje vrijednosti za atribute):

```
Adrijan Omicevic
Dominik Mesek
Danijel Tolj
Patrik Pralas
Noa Tubic
Average salary of 3 teachers is: 33333.33
```

   Također, za sljedeći isječak koda:

```
final p1 = Person(name: "Adrijan", surname: "Omi", age: 26);
final p2 = Person(name: "Adrijan", surname: "Omi", age: 26);
final p3 = Student(
    name: "Patrik",
    surname: "Pralas",
    age: 24,
    studentId: "0036483352",
    academicYear: 2022);
final p4 = Student(
    name: "Noa",
```

```
      surname: "Tubic",
      age: 25,
      studentId: "0036483352",
      academicYear: 2022);

  print(p1 == p2);
  print(p1 == p3);
  print(p3 == p4);
```

očekuje se ovakav ispis:

```
true
false
true
```

6. Nastavnici se natječu u fakultetskom „Master Chef“ natjecanju, u kojem svaki nastavnik priprema jedan desert, a studenti ih ocjenjuju. Za to ćeš napraviti klasu `CompetitionEntry` koja u konstruktoru sadrži referencu na `Teacher` (osoba koja je pripremila desert) i referencu na `Dessert`. Prilikom generiranja konstruktora napraviti listu studenata koja može primiti najviše 3 elementa studenata,te napraviti listu ratings (lista int-ova) koja isto može primiti najviše 3 elementa. To je potrebno jer jedan student moći ocijeniti jedan desert (3 studenta i 3 ratinga). Napraviti sve gettere(trebat će getter i za studente) te metodu `addRating` koja ima parametre `Student` i broj INT (rating), a vraća boolean ovisno o tome je li uspjela ili ne ubaciti novi zapis u dani `CompetitionEntry` (najviše tri ratinga i studenti se ne smiju ponavljati). Napišite i metodu `getRating` koja vraća prosječnu ocjenu svih studenata koji su ocijenili neki `CompetitionEntry`.
7. Napiši klasu `UniMasterChef` koji u kontruktoru prima broj int (broj prijava na natjecanje). Prilikom generiranja kontruktora napravi listu `CompetitionEntry`, koji može primiti toliko elemenata koliko je definirano u kontruktoru u argumentu za broj prijava za natjecanje. Napiši metodu `addEntry` koji prima `Competitionentry` , a vraća boolean ovisno o tome je li se uspio ubaciti u listu CompetitionEntry (npr. Ako je definirano da lista smije primiti 3 elementa, a želi se ubaciti 4, onda se taj element nece ubaciti i vratit ce false). Isto tako napravi metodu `getBestDessert` koja će vratiti najbolje ocijenjeni desert, kao i statičku metodu `getInvolvedPeople` koja prima argument tipa `CompetitionEntry`, a vraća listu osoba (Person) koji su sudjelovali u izradi ili ocjenjivanju kolača.

   Za isječak koda:

```
void main() {
  Dessert dessert = Dessert(name: "Čokolada", weight: 100, calories: 400);
  Cake cake = Cake(
      name: "Morski vjetar",
      weight: 300,
      calories: 500,
      containsGluten: true,
      cakeType: "birthday");

  final teacher1 = Teacher(
      name: "Adrijan",
      surname: "Omicevic",
      age: 26,
```

```
    email: "adrijan.omicevic@q.agency",
    subject: "Flutter",
    salary: 30000);
final teacher2 = Teacher(
    name: "Dominik",
    surname: "Mesek",
    age: 24,
    email: "dominik.mesek@q.agency",
    subject: "Flutter & Dart",
    salary: 20000);

final student1 = Student(
    name: "Patrik",
    surname: "Pralas",
    age: 24,
    studentId: "AFFJI43838",
    academicYear: 2022);

final student2 = Student(
    name: "Noa",
    surname: "Tubic",
    age: 25,
    studentId: "SFSJI44635",
    academicYear: 2022);

final student3 = Student(
    name: "Marta",
    surname: "Rep",
    age: 23,
    studentId: "SFDJI55635",
    academicYear: 2022);

UniMasterChef competition = UniMasterChef(numberOfEntries: 2);

CompetitionEntry e1 = CompetitionEntry(dessert, teacher1);
competition.addEntry(e1);
print("Entry 1 rating: ${e1.getRating()}");

e1.addRating(student1, 4);
e1.addRating(student2, 5);
print("Entry 1 rating: ${e1.getRating()}");

CompetitionEntry e2 = CompetitionEntry(dessert, teacher2);
competition.addEntry(e2);

e2.addRating(student1, 4);
e2.addRating(student3, 5);
e2.addRating(student2, 5);
```

```
  print("Entry 2 rating: ${e2.getRating()}");

  print("Best dessert is: ${competition.getBestDessert()!.getName}");

  List<Person> e2persons = UniMasterChef.getInvolvedPeople(e2);

  for (Person p in e2persons) {
    print(p);
  }
}
```

Treba ispisati:

```
Entry 1 rating: 0.0
Entry 1 rating: 4.5
Entry 2 rating: 4.67
Best dessert is: Čokolada
name: Dominik, surname: Mesek, age: 24, email: dominik.mesek@q.agency, subject:
Flutter & Dart, salary: 20000.0
name: Patrik, surname: Pralas, age: 24, studentId: AFFJI43838, academicYear: 2022
name: Marta, surname: Rep, age: 23, studentId: SFDJI55635, academicYear: 2022
name: Noa, surname: Tubic, age: 25, studentId: SFSJI44635, academicYear: 2022
```