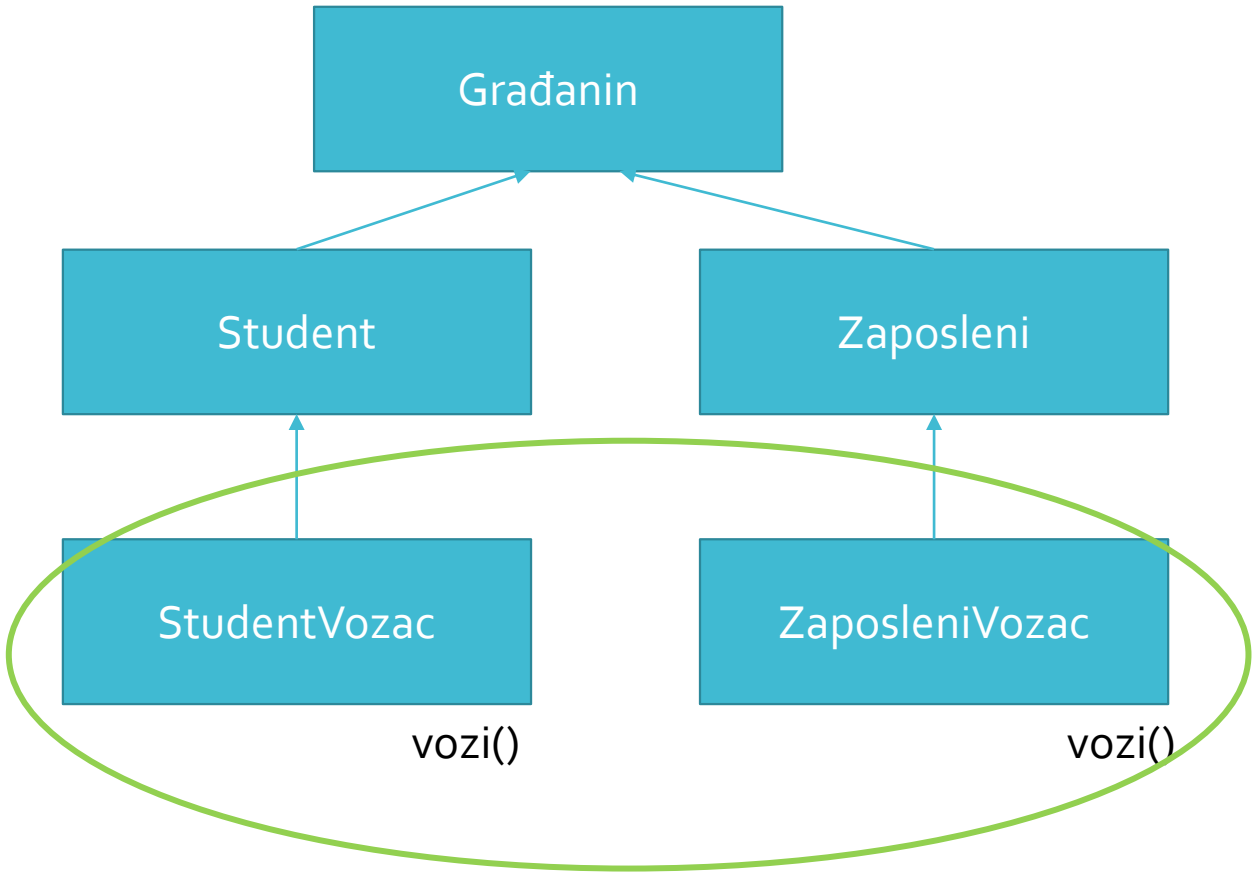
APSTRAKTNI TIPOVI - INTERFEJSI

2020/21

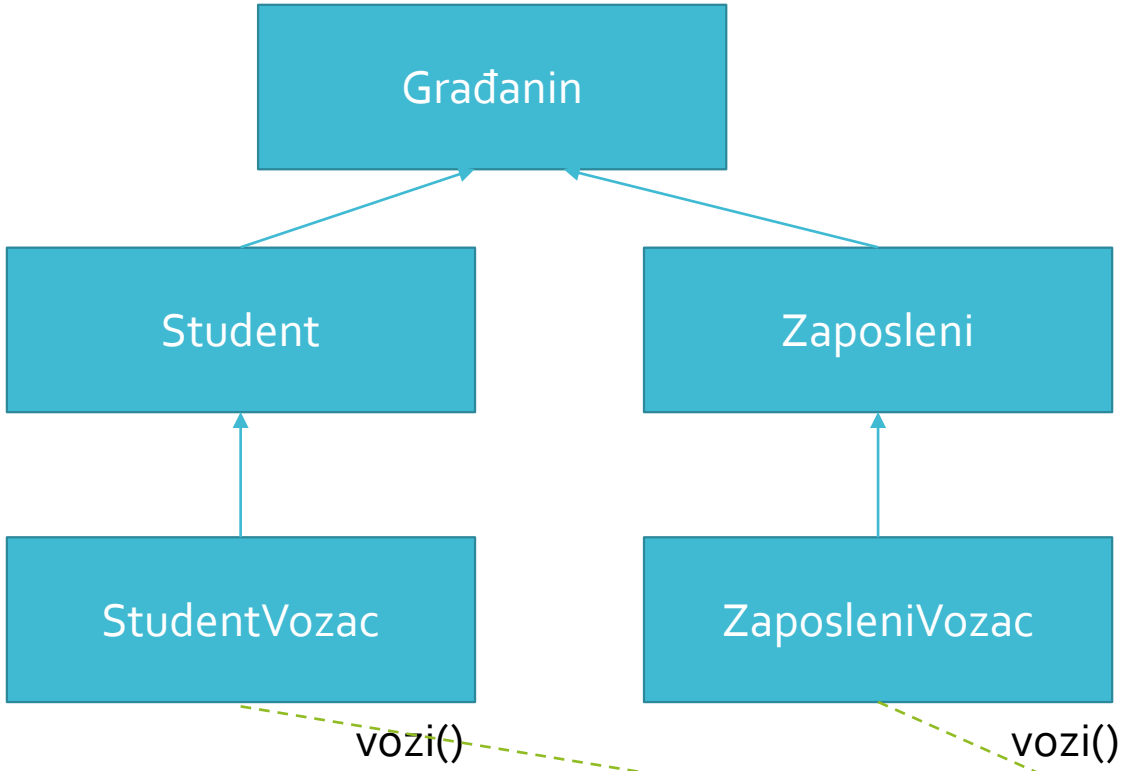
PRIRODNO-MATEMATIČKI FAKULTET, UNIVERZITET U KRAGUJEVCU

Drugačiji problem




```
StudentVozac s = new StudentVozac();
ZaposleniVozac z = new ZaposleniVozac();
```

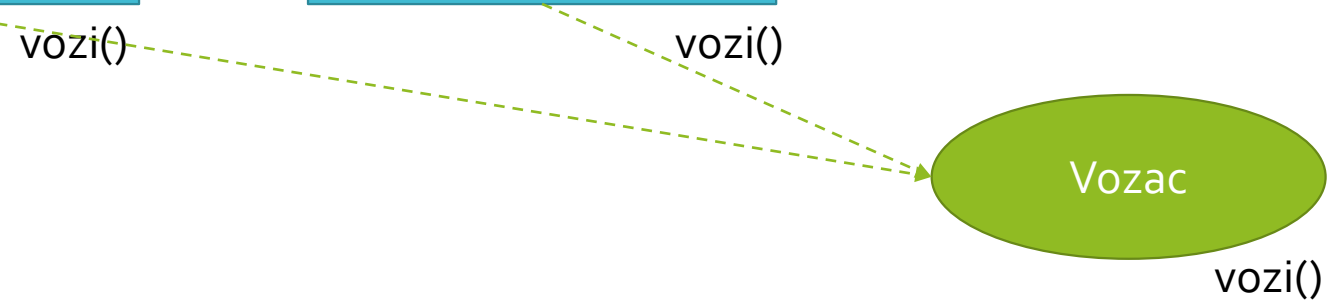
Drugačiji problem



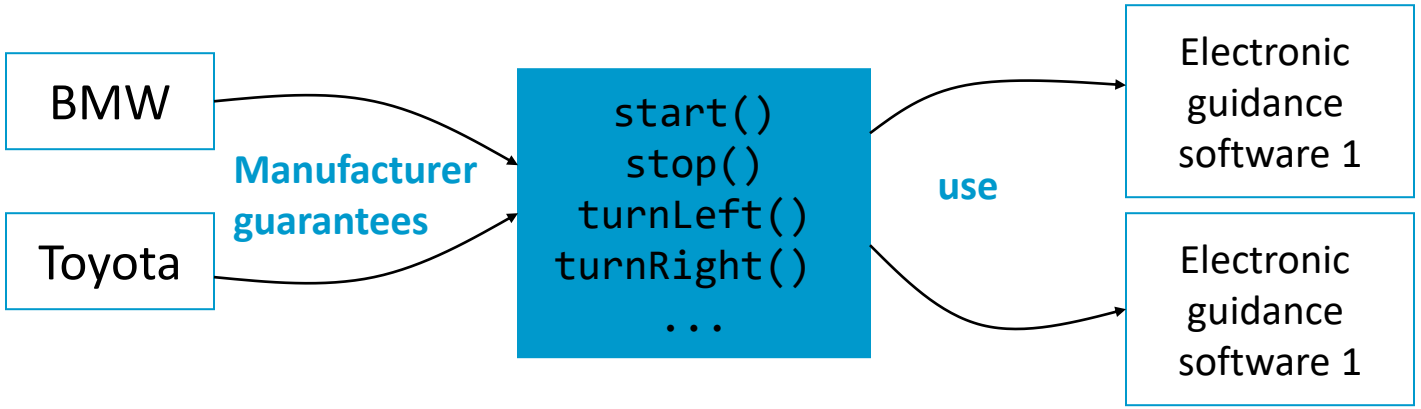

```
StudentVozac s = new StudentVozac();
ZaposleniVozac z = new ZaposleniVozac();
```

Taksi

```
Vozac v;
```



UGOVORI


BMW
Toyota
Manufacturer guarantees
start()
stop()
turnLeft()
turnRight()
...
use
Electronic guidance software 1
Electronic guidance software 1

INTERFEJS

- U Javi interfejs je reference tip (kao i klasa)
  - koji sadrži
    metode koji su implicitno (podrazumevano) `public` i `abstract`,
    podatke koji su implicitno (podrazumevano) `public`, `static` i `final`

```
public interface Merljivo{
      double PI = 3.14;
      int jeJednako(Merljivo u);
      double velicina();
}
```

  - koji se ne može instancirati
  - može biti proširen – tačnije moguće je jedan interfejs izvesti iz drugog

```
public interface MerljivoExt extends Merljivo{
      double obim();
}
```

UPOTREBA

Upotreba:

- Klase implementiraju interfejse
  Nakon navodjenja imena neke podklase i eveltualnog definisanja da je izvedena iz neke nadklase, mogu se opciono implemetirati (implements) **jedan ili više interfejsa** (simulacija višestrukog nasleđivanja)

```
class Circle extends GraphicObject implements MerljivoExt {
    // implemetacija nasledjenih apstraktnih metoda
    void draw() { ... }
    void resize() { ... }
    // implemetacija metoda interfesa
    public int jeJednako() { ... }
    public double velicina() { ... }
    public double obim() { ... }
}
```

- Kada se za neku **konkretnu** klasu definiše da implemetira neki interfejs tada se u njoj moraju implementirati i sve metode interfejsa. Zašto?

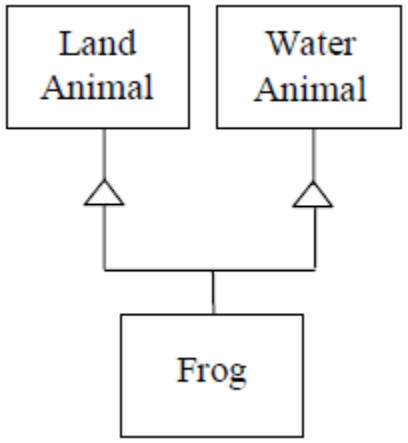
UPOTREBA

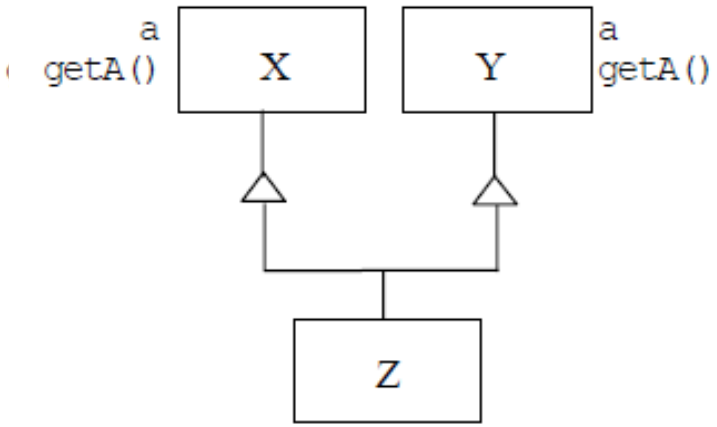
- Definicije interfejsa uvode nova imena tipova
- **Referenca na objekat koji implementira interfejs se može dodeliti referenci tipa tog interfejsa.**

```
class Test {
    public static void main(String[] args){
     MerljivoExt[] nizMerljivih = new MerljivoExt[10];
     nizMerljivih[0] = new Circle();
     nizMerljivih[1] = new Rectangle();
     double suma = 0;
     for(int i=0; i<nizMerljivih.length; i++)
     if (nizMerljivih[i]<>null) suma+=nizMerljivih[i].velicina();
     System.out.println(suma);
    }
}
```

VIŠESTRUKO NASLEĐIVANJE

- Nasleđivanje u OO jezicima može biti:
  - **Jednostruko** – klasa neposredno može da nasledi samo jendu klasu
  - **Višestruko** – klasa može imati dva ili više neposrednih roditeljskih klasa





- Problemi višestrukog nasleđivanja potiču od višestrukog nasleđivanja implementacije.
- Java dozvoljava samo jednostruko.

VIŠESTRUKO NASLEĐIVANJE

- Java dozvoljava samo jednostruko nasleđivanje, ali nudi način da se nasledi samo ugovor i to implementacijom interfejsa.
- Kako jedna klasa može da implementira više interfejsa, to su interfejsi mehanizam kojim je u Javi višestruko nasleđivanje delimično omogućeno.
- Supertipovi neke klase su:
  - klasa čije definicije proširuje
  - interfejsi koji su implementirani

```
class A extends N implements interfejs1, interfejs2 {
    . . .
}

A a = new A();
N n = a;
Object o = a;
interfejs1 i1 = a;
interfejs1 i2 = a;
```