Dedovanje

1

- dedovanje je ena ključnih lastnosti OOP
- omogoča uporabo obstoječih razredov v novih razredih
  - novi razred le dopolnimo z dodatnimi lastnostmi in obnašanji
  - to imenujemo **izpeljevanje razredov**
- dedovanje določa relacijo "je vrste" med razredi
  - dobimo hierarhijo razredov

Dedovanje – osnovni pojmi

2

- **izpeljani razred** ali **podrazred**
  - je novi razred, v katerem uporabimo obstoječe razrede
- **bazni razred** ali **nadrazred**
  - je razred, iz katerega smo izpeljali novi razred
- **enkratno** in **večkratno dedovanje**
  - pove, iz koliko razredov izpeljujemo novi razred
- **specializacija**
  - izpeljani razredi imajo dodatne lastnosti glede na nadrazrede
- **posplošitev**
  - nadrazredi predstavljajo posplošitev izpeljanih razredov

Dedovanje – izpeljani razred

3

- izpeljani razred **podeduje vse** lastnosti in obnašanja nadrazreda
  - ni nujno, da ima dostop do vseh
- izpeljani razred dopolnimo z dodatnimi lastnostmi in obnašanji
- objekt izpeljanega razreda je hkrati objekt nadrazreda
  - to pa zato, ker ima izpeljani razred vse lastnosti nadrazreda
  - obratno ni res: v objektu nadrazreda manjkajo "nove" lastnosti izpeljanega razreda

Primeri nadrazredov in podrazredov

4

nadrazred	podrazred
oseba	študent, profesor
študent	redni študent, izredni študent
delavec	vodja izmene, direktor
objekt v ravnini	točka, premica
štirikotnik	paralelogram, kvadrat, trapez

Sintaksa izpeljevanja razredov

5

- razred *COtrok* izpeljemo iz razreda *CStars* na naslednji način:

```
class COtrok : public CStars {
    // dodatne lastnosti in obnašanja
    // novega razreda
};
```

Izpeljevanje razredov

6

- razredi ne dedujejo konstruktorjev
- če želimo v konstruktorju izpeljanega razreda izvesti konstruktor nadrazreda, lahko to storimo v **seznamu inicializacij** konstruktorja

PRIMER:

```
COtrok::COtrok() : CStars(), ... {
}
```

Primer: razred CTocka3D

7

- napisali smo že razred za točko v 2D ravnini
- v čem se razlikuje taka točka s točko v 3D prostoru?
  - imamo eno dodatno koordinato
  - ostale funkcionalnosti so enake
    - lahko določimo vrednosti koordinat
    - lahko pogledamo vrednosti koordinat
    - lahko izvedemo translacijo za nek vektor v prostoru
    - lahko izpišemo koordinate točk na ekran
  - seveda lahko dodamo dodatne funkcionalnosti

Primer: razred CTocka3D (tocka3d.h)

8

```
#include "tocka2d.h"

class CTocka3D : public CTocka2D {
public:
  CTocka3D();
  CTocka3D(double x, double y, double z);
  ~CTocka3D();
  void doloci(double x, double y, double z);
  double vrniZ();
  void izpis();

private:
  double z;
};
```

Primer: razred CTocka3D (tocka3d.cpp)

9

```
CTocka3D::CTocka3D() : CTocka2D() {
    z = 0;
}

CTocka3D::CTocka3D(double x, double y, double z) :
CTocka2D(x,y) {
    this->z = z;
}

CTocka3D::~CTocka3D() {
}
```

Primer: razred CTocka3D (tocka3d.cpp)

10

```
void CTocka3D::doloci(double x, double y, double z) {
    this->CTocka2D::doloci(x,y);
    this->z = z;
}

double CTocka3D::vrniZ() {
    return z;
}

void CTocka3D::izpis() {
    cout << "(" << vrniX() << ", " << vrniY() << ", " << z << ")";
}
```

Primer uporabe razredov za točke

11

```
#include "tocka3d.h"

int main() {
  CTocka2D a(1,3.7), b; // b = (0, 0)
  b.doloci(1,-1);
  b.izpis();
  CTocka3D c(1,1,1), d;
  c.izpis();
  d.izpis();
  return 0;
}
```