# Nasleđivanje

- Ova relacija naziva se *nasleđivanje* (*inheritance*): klasa *ClockWithDate* nasleđuje sve osobine (atribute, operacije) osnovne klase *Clock*, ali je i *specijalizuje* (*specializes*) ili *proširuje* (*extends*), dodavanjem osobina (atributa, operacija), koje osnovna klasa nema
- Terminologija:
  - *Osnovna klasa* (*base class*), *generalizacija* (*generalization*), *roditelj* (*parent*)
  - *Izvedena klasa* (*derived class*), *specijalizacija* (*specialization*), *dete* (*child*)
- Sada se sa objektima izvedene klase *ClockWithDate* može raditi sve što i sa objektima osnovne klase *Clock*, jer su oni *jedna vrsta* (*kind of*) objekata osnovne klase, pošto poseduju (nasleđuju) sve osobine osnovne klase
- Sa objektima izvedene klase može se raditi i ono što je specifično za tu izvenedu klasu, a nije svojstveno osnovnoj:

```
Clock* simpleClock = new Clock(13,17,0);
ClockWithDate* smartClock = new ClockWithDate(2018,9,13,13,17,0);

simpleClock->setTime(13,20,0);
smartClock->setTime(13,20,0);

simpleClock->setDate(2018,9,14);
smartClock->setDate(2018,9,14);
```

Greška u prevođenju

# Nasleđivanje

- Objekti izvedene klase su (indirektne) instance i osnovne klase!
- *Pravilo supstitucije* (*Substitution rule*, B. Liskov): gde god i kad god se očekuje objekat osnovne klase, može se pojaviti i objekat izvedene klase, jer
  - je objekat izvedene klase takođe i primerak (instanca) osnovne klase, jer
  - se sa njim može raditi sve što i sa objektima osnovne klase, jer
  - objekat izvedene klase ima (nasleđuje) sve osobine osnovne klase.
- Konverzija koju dozvoljava C++ i to implicitno:

  `DerivedClass*` → `BaseClass*`

  `ClockWithDate*` → `Clock*`

```
Lobby::Lobby (unsigned n, string ct[], int lg[], int dateYN[]) {
  …
  for (int i=0; i<num; i++) {
    …
    if (dateYN[i])
      clocks[i] = new ClockWithDate(1970,1,1,h,0,0);
    else
      clocks[i] = new Clock(h,0,0);
  }
}
```