Sa proceduralnog na OO programiranje: klase i objekti

- Pokazivač *this* se može koristiti kao i svaki drugi pokazivač unutar funkcije članice, jer on prosto ukazuje na objekat (čija je funkcija pozvana), na primer, ako je drugom objektu potreba veza ka tom objektu
- Na primer, hoćemo da objekti klase *Clock* imaju vezu ka objektu klase *Lobby* koji ih sadrži:

```
class Clock {
public:
  Clock (Lobby* owner, int hh, int mm, int ss);
  …
private:
  Lobby* myOwner; // The Lobby that contains this Clock
  …
};

Clock::Clock (Lobby* owner, int hh, int mm, int ss) {
  this->myOwner = owner;
}

Lobby::Lobby (unsigned n, string ct[], int lg[]) {
  …
  for (int i=0; i<num; i++) {
    …
    this->clocks[i] = new Clock(this,h,0,0);
  }
}
```

Sa proceduralnog na OO programiranje: polimorfizam

- Rudiment polimorfizma u proceduralnom programiranju — isti kao i za modularnost i enkapsulaciju, isključivo statički, vezan za pisanje i prevođenje programa:
  - deklaracija (potpis, *signature*) nekog potprograma:

```
int printf (char* format, …);
```

  - implementacije (potpune definicije tela) tog potprograma:

```
int printf (char* format, …) {
  …
}
```

- Iza istog interfejsa se može kriti različita implementacija i ona se može i promeniti, ali isključivo promenom koda implementacije i njegovim ponovnim prevođenjem.