Klasa kao realizacija apstraktnog tipa podataka

- Ostaje još operacija dodele vrednosti:

```
c3 = c5;
```

- Operacija dodele vrednosti je *različita* od *inicijalizacije*:

```
complex c3 = c5; // Initialization
c3 = c5; // Assignment operation
```

Inicijalizacija; poziva se: `complex::complex(c5)`

Operacija dodele; poziva se: `c3.complex::operator=(c5)`

  - inicijalizacija se vrši kada se objekat kreira; operacija dodele se vrši kao operacija u izrazu
  - inicijalizacija se vrši nad objektom koji tek nastaje i do tada nije postojao; operacija dodele se vrši nad objektom koji već postoji
  - kod inicijalizacije se poziva konstruktor; kod dodele vrednosti se poziva operator dodele kao operatorska funkcija

- U jeziku C nije bilo posebne potrebe da se razlikuju ove dve operacije, jer postoje samo ugrađeni tipovi, za koje je semantika ove dve operacije uvek ista - prosto kopiranje vrednosti
- Operator dodele je nestatička operatorska funkcija klase:

```
complex complex::operator= (complex other) {
  this->re = other.re; this->im = other.im;
  return *this;
}
```

- Zašto vraćamo **this*? Da bi se redefinisana operatorska funkcija ponašala što sličnije onoj ugrađenoj, a ona vraća vrednost, i to baš onu dodeljenu, kako bi se moglo pisati:

```
c1 = c2 = c3;  // Computed as: c1 = (c2=c3)
```

- Svaka klasa ima podrazumevani operator dodele, koji vrši podrazumevanu dodelu član po član, ali se on može i eksplicitno redefinisati

Klasa kao realizacija strukture podataka

- Želimo da realizujemo određenu strukturu podataka, npr. stek kakav smo već skicirali, sa elementima tipa *unsigned int* i kapaciteta *MaxStackSize*:

```
// File: stack.h

const int MaxStackSize = 256;

class Stack {
public:
  Stack ();
  int push (unsigned in);
  int pop (unsigned* out);
private:
  unsigned stack[MaxStackSize];  // Stack
  int sp; // Stack pointer
};


// File stack.cpp
#include "stack.h"

Stack::Stack () {
  this->sp = 0;
}

int Stack::push (unsigned in) {
  if (this->sp==MaxStackSize) return -1;
  this->stack[this->sp++] = in;
  return 0;
}

int Stack::pop (unsigned* out) {
  if (this->sp==0) return -1;
  *out = this->stack[--this->sp];
  return 0;
}
```