Konstantni tipovi i funkcije članice

- Prevodilac neće dozvoliti poziv nekonstantne funkcije članice za konstantan objekat (uključujući i indirektan pristup preko pokazivača ili reference na konstantan objekat). Na primer:

```
class X {
public:
  X (int ii) { write(ii); }
  int read () const { return i; }
  void write (int ii) { i = ii; }
private:
  int i;
};
...
X x(0);
const X cx(1);
X* px = &x;
const X* pcx = &cx;
X& rx = x;
const X& rcx = cx;

x.read();
x.write();
cx.read();
cx.write();

px->read();
px->write();
pcx->read();
pcx->write();

rx.read();
rx.write();
rcx.read();
rcx.write();
```

Sve ovo je u redu, jer se za nekonstantan objekat može pozivati i konstantna i nekonstantna funkcija članica

Greške u prevođenju (označeno crvenim): ne može se pozivati nekonstantna funkcija članica za konstantan objekat

Konstantni tipovi i funkcije članice

- Prevodilac zapravo sprovodi ista opšta pravila za pokazivače na (ne)konstantne objekte, jer je:
  - u nekonstantnoj, nestatičkoj funkciji članici klase *X*, pokazivač *this* implicitno deklarisan kao pokazivač tipa *X* const* (konstantan pokazivač na *nekonstantan* objekat)
  - u konstantnoj, nestatičkoj funkciji članici klase *X*, pokazivač *this* je implicitno deklarisan kao pokazivač tipa *const X* const* (konstantan pokazivač na *konstantan* objekat)

```
class X {
public:
  X (int ii) { set(ii); }

  int read () const { return i; }
  void write (int ii) { i = ii; }

private:
  int i;
};

…

X x(0);
const X cx(1);

x.read();
x.write();
cx.read();
cx.write();
```