Konstantni tipovi i funkcije članice

- String-literali imaju tip *const char[]* (niz konstantnih znakova), pa su u skladu sa tim dozvoljene njihove implicitne konverzije u *const char**, ali ne i u *char**:

```
const char* p = "Hello";
char* q = "World";
```

Greška u prevođenju

- Naravno, sve to važi i ukoliko su parametri funkcije pokazivači na konstantan tip: tada funkcija "obećava" da neće izmeniti ono na šta taj parametar ukazuje (jer je opseg važenja parametra lokalna za tu funkciju), pa se funkcija može pozvati i sa pokazivačem na konstantan i sa pokazivačem na nekonstantan objekat (u suprotnom može samo za nekonstantan):

```
void f1 (const char*);
void f2 (char*);
char s1 = …;
const char s2 = …;

void f () {
  f1(&s1);
  f2(&s1);
  f1(&s2);
  f2(&s2);
}
```

Funkcija *f1* "obećava" da neće menjati ono na šta parametar ukazuje

Funkcija *f2* "ne obećava" da neće menjati ono na šta parametar ukazuje

U redu: inicijalizacija argumenta je *const char* arg = &s1* (dozvoljena implicitna konverzija)

U redu: inicijalizacija argumenta je *char* arg = &s1* (isti tipovi)

U redu: inicijalizacija argumenta je *const char* arg = &s2* (isti tipovi)

Greška u prevođenju: inicijalizacija argumenta je *char* arg = &s2*

- Sva navedena pravila konverzija važe potpuno isto i za reference na konstantne/nekonstante tipove

Konstantni tipovi i funkcije članice

- Nestatičke operacije (funkcije članice) klase se generalno mogu posmatrati razvrstane u dve kategorije:
  - operacije koje *ne menjaju* spolja vidljivo stanje objekta (*externally visible state*), odnosno ništa ne upisuju u atribute objekta, već samo čitaju to stanje/vrednosti atribute i vraćaju informacije o tome; ovakve operacije nazivaju se ponekad *inspektori* (*inspector*) ili *selektori* (*selector*)
  - operacije koje *menjaju* spolja vidljivo stanje objekta, odnosno upisuju nešto u atribute objekta; ovakve operacije ponekad se nazivaju *modifikatori* (*modifier*) ili *mutatori* (*mutator*)
- Na jeziku C++, operacije koje ne menjaju stanje objekta nazivaju se *konstantne funkcije članice* (*constant member functions*) i označavaju se specifikatorom *const* iza liste argumenata; operacije koje menjaju stanje objekta ne označavaju se posebno:

```
class Clock {
public:

  Clock (int hh, int mm, int ss);

  void tick ();
  string getTime () const;
  void setTime (int hh, int mm, int ss);

  …

};
```