# Inicijalizacija

- Inicijalizacija lokalne statičke varijable vrši se kada bilo koje izvršavanje (u bilo kojoj niti) prvi put naiđe na njenu definiciju; svaki sledeći nailazak preskače inicijalizaciju
- Ako ova inicijalizacija baci izuzetak, statička varijabla se ne smatra inicijalizovanom i inicijalizacija će biti pokušana pri sledećem nailasku izvršavanja na istu definiciju
- Nestatički podatak član može biti inicijalizovan na dva načina:
  - u listi inicijalizatora članova (*member initializer list*) u konstruktoru klase:

```
X::X (int j) : i(j+1) {}
```

Lista inicijalizatora članova

  - podrazumevanim inicijalizatorom člana (*default member initializer*) u definiciji klase, koji se koristi ako se član izostavi iz liste inicijalizatora članova u konstruktoru; ako se član sa podrazumevanim inicijalizatorom pojavi i u listi inicijalizatora članova, ta podrazumevana inicijalizacija se ignoriše:

```
struct X {
  int i = 1;
  string s{'H','e', 'l', 'l', 'o'};

  X () {}

  X (int j) : i(j+1) {}
};
```

Podrazumevani inicijalizatori

Članovi *i* i *s* se inicijalizuju podrazumevanim inicijalizatorima

Član *s* se inicijalizuje podrazumevanim inicijalizatorom, a *i* izrazom *j+1*

# Konstantna inicijalizacija

- Konstantna inicijalizacija se vrši za varijable sa statičkim trajanjem skladištenja i onima vezanim za niti (*thread local*), samo pod uslovom da su inicijalizovani konstantnim izrazom
- *Konstantan izraz* (*constant expression*) je izraz koji se može izračunati za vreme prevođenja i može se koristiti gde god je potreban konstantan izraz (npr. za dimenzije nizova ili inicijalizaciju konstanti)
- Ovakvi su izrazi koji uključuju npr. samo literale, konstante primitivnih tipova i operatore koji se mogu izračunati za vreme prevođenja
- Međutim, od verzije C++11, u konstantnim izrazima mogu učestvovati i varijable, uključujući i objekte klasa, pa čak i pozivi korisničkih funkcija, uključujući i pozive funkcija članica klasa, pod uslovom da su te funkcije deklarisane kao *constexpr*, i da se te funkcije, uključujući i konstruktore, mogu izvršiti za vreme prevođenja, što znači da su im argumenti konstantni izrazi, da su inicijalizatori svih podobjekata konstantni izrazi, da su svi podaci članovi inicijalizovani i slično
- Ovakve klase i funkcije moraju zadovoljiti niz definisanih uslova, pri čemu su se ti uslovi menjali kroz verzije jezika, i dalje se menjaju u novijim verzijama jezika u pravcu relaksacije ograničenja; na primer, od verzije C++20, ove funkcije mogu da budu čak i virtuelne
- Jedno od osnovnih ograničenja jeste to da se te funkcije ne mogu oslanjati na pozive funkcija koje nisu označene kao *constexpr* ili vrednosti varijabli koje nisu označene kao *constexpr* ili podataka članova koji nemaju inicijalizatore
- Funkcije koje su označene kao *constexpr*, uključujući i konstruktore, mogu se pozivati i van konstantnih izraza, odnosno za vreme izvršavanja
- Objekti deklarisani kao *constexr* su implicitno konstantni i moraju biti inicijalizovani konstantnim izrazima; funkcije koje su označene kao *constexpr* su implicitno *inline*