Semantika vrednosti od C++17

- Zapravo je formalna semantika povratnih vrednosti poziva funkcija (uključujući i rezultate operatorskih funkcija koje se pozivaju za preklopljene operatore) značajno promenjena od verzije C++17: ako funkcija vraća rezultat koji nije referenca, taj rezultat *nije privremeni objekat*, kao što je to ranije bilo, nego samo *vrednost* (*value*), i to *čdvrednost* (*prvalue*)
- Čdvrednost predstavlja "potrebu za inicijalizacijom", ali ne obavezno privremenog objekta (koja se opciono može i preskočiti): ta inicijalizacija obaviće se onako kako je definisano operandom naredbe *return* koja vraća tu vrednost, a obaviće se u zavisnosti od konteksta u kom se koristi
- Iz funkcija se tako, formalno, vraćaju samo vrednosti i one se dalje prosleđuju kao argumenti/operandi ili inicijalizatori dalje u izrazima ili inicijalizacijama
- Privremeni objekat se pravi samo u određenim situacijama, odnosno u zavisnosti od konteksta, kada se ta vrednost *materijalizuje* u privremeni objekat (*temporary materialization*)
- Materijalizacija privremenog objekta se maksimalno odlaže do trenutka kada se taj objekat mora napraviti, a sve do tada se samo koristi vrednost

Semantika vrednosti od C++17

- To se u mnogim slučajevima svodi na obavezu izbegavanja kopiranja. Na primer, sledeća inicijalizacija formalno se tumači ovako:

```
X f () { return X(1); }
X x = X(X(f()));
```

  - Funkcija *f* u naredbi *return* vraća vrednost koja je čdvrednost i koja "predstavlja" inicijalizaciju nekog objekta (ali tek kada se on odredi kontekstom i ne obavezno privremenog) pozivom konstruktora *X(1)*
  - Ova čdvrednost prosleđuje se kao "zahtev" za inicijalizaciju ponovo čdvrednosti *X(f())*, dakle čdvrednost koju vraća *X(f())*, predstavlja istu prethodno navedenu inicijalizaciju; slično važi za još jednu čdvrednost *X(X(f()))*
    - najzad, tom čdvrednošću *X(X(f()))* inicijalizuje se objekat *x*, tako da se navedena inicijalizacija pozivom konstuktora *X(1)* konačno "vezuje" za objekat *x*

- Implementaciono, sve se opet svodi na isti mehanizam sprovođenja izostavljanja kopiranja RVO tako što pozvana funkcija, ukoliko vraća čdvrednost, dobija adresu mesta (objekta) koji treba da inicijalizuje u naredbi *return*, i ta adresa prosleđuje se tranzitivno

- Međutim, formalna semantika jezika je promenjena i sada konstruktori kopije i premeštanja ne moraju uopšte da budu definisani da bi ovo bilo moguće:

```
struct X {
  X (int);
  X (const X&) = delete;
  X (X&&) = delete;
};

X f () { return X(1); }

X x = f();
```