Sa proceduralnog na OO programiranje: klase i objekti

Upravo tako se na jeziku C++ implementiraju klase i objekti:

- Objekti se u vreme izvršavanja implementiraju kao instance obične C strukture koja sadrži samo vrednosti podataka članova date klase
- Za svaku (nestatičku) funkciju članicu klase, prevodilac generiše kod koji izgleda potpuno isto kao za "običnu" C funkciju, s tim što ta funkcija ima još jedan, prvi, skriveni argument *this* koji je pokazivač na ovu strukturu
- Svako neposredno obraćanje podatku članu unutar te funkcije članice:

```
sp = 0;
```

  prevodilac prevodi u implicitan pristup preko pokazivača *this*:

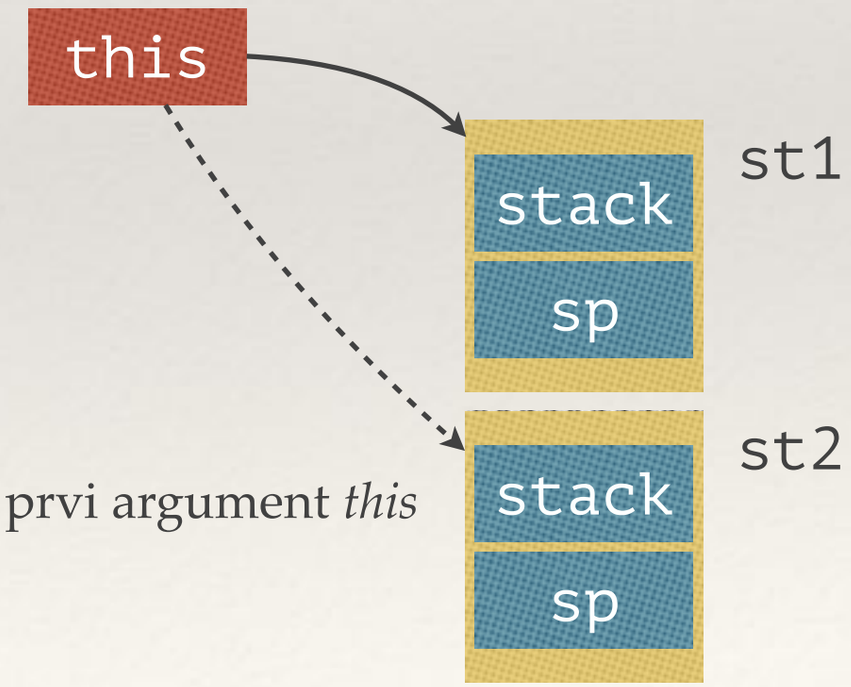
```
this->sp = 0;
```

- Svaki poziv funkcije članice za dati objekat:

```
pSt1->push(in);
```

  prevodilac prevodi u poziv "obične" C funkcije, pri čemu joj kao taj prvi argument *this* prenosi pokazivač na taj objekat:

```
stack_push(pSt1,in);
```



Sa proceduralnog na OO programiranje: klase i objekti

- Na jeziku C++, svaka (nestatička) funkcija članica klase poseduje lokalni, implicitno deklarisan pokazivač *this*; ovaj pokazivač je konstantan (ne može mu se promeniti vrednost)
- Tokom izvršavanja tela ovakve funkcije članice pokazivač *this* pokazuje na objekat za koga je ta funkcija pozvana
- Svaki neposredan pristup članu klase unutar ovakve funkcije smatra se implicitnim pristupom članu onog objekta na koga pokazuje *this* (uvek se može vršiti i eksplicitan pristup, što ponekad može poboljšati čitljivost ili razrešiti višeznačnost):

```
sp = 0;
```

je isto što i:

```
this->sp = 0;
```

- Privid da svaki objekat "poseduje svoju" funkciju članicu je zapravo posledica toga što se funkcija članica poziva za dati objekat, a članovima u tekstu tela funkcije može pristupati neposredno, bez navođenja objekta (podrazumeva se pristup članu onog objekta "čija" se funkcija izvršava). Ovaj privid stvara se jednostavnim opisanim mehanizmom implicitnog pristupa preko pokazivača *this*