Klasa kao realizacija strukture podataka

- Šta ako nam treba stek koji će skladištiti elemente nekog drugog tipa *T* i/ili drugog kapaciteta?

```
// File: stack.h

const int MaxStackSize = 512;

class Stack {
public:
  Stack ();
  int push (T in);
  int pop (T* out);

private:
  T stack[MaxStackSize];  // Stack
  int sp; // Stack pointer
};


// File stack.cpp
#include "stack.h"

Stack::Stack () {
  this->sp = 0;
}

int Stack::push (T in) {
  if (this->sp==MaxStackSize) return -1;
  this->stack[this->sp++] = in;
  return 0;
}

int Stack::pop (T* out) {
  if (this->sp==0) return -1;
  *out = this->stack[--this->sp];
  return 0;
}
```

Klasa kao realizacija strukture podataka

- Zaključujemo:
  - tip *T* može biti bilo koji tip, sve dok su za taj tip *T* definisane operacije koje se u ovoj klasi očekuju:
    - inicijalizacija kopiranjem, zbog prenosa argumenata u operaciju *push* i inicijalizacije niza *stack* (podrazumevano imaju svi ugrađeni tipovi i klase)
    - dodela vrednosti, zbog smeštanja u elemente niza *stack* u operaciji *push* i smeštanja povratne vrednosti u operaciji *pop* (podrazumevano imaju svi ugrađeni tipovi i klase)
    - kapacitet steka može biti bilo koja celobrojna pozitivna vrednost
    - ako želimo nešto od ovoga da promenimo, tj. da napravimo više *klasa* sa promenjenim vrednostima nekog od ovih parametara, moramo da radimo dosadan, pravolinijski, fizički posao proste zamene svih pojava određenog parametra, uz variranje naziva za svaku od tih klasa
- Očigledna potreba za *automatizacijom*: umesto programera, ovaj rutinski posao može da radi *prevodilac*
- Koncept *šablonske klase* (*template class*) ili *generičke klase* (*generic class*): obrazac klase, parametrizovan tipovima i/ili konstantama, po kome će prevodilac *generisati* kod zamenom svih parametara šablona konkretnim, stvarnim parametrima
- Rezultat je isti kao da je ovaj posao urađen ručno: generisane klase su *različite* klase, nemaju nikakve posebne međusobne veze
- Dakle, klasa može realizovati i *apstraktnu strukturu podataka* (*abstract data structure*): strukturu koja skladišti elemente proizvoljnog tipa, pri čemu zahteva od tog tipa samo određena svojstva i usluge, ne i obavezu da bude neki konkretan tip