Klasa kao realizacija strukture podataka

- Želimo da realizujemo određenu strukturu podataka, npr. stek kakav smo već skicirali, sa elementima tipa *unsigned int* i kapaciteta *MaxStackSize*:

```
// File: stack.h

const int MaxStackSize = 256;

class Stack {
public:
  Stack ();
  int push (unsigned in);
  int pop (unsigned* out);
private:
  unsigned stack[MaxStackSize];  // Stack
  int sp; // Stack pointer
};


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

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

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

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

Konstanta tipa *int*, ne može se menjati nakon inicijalizacije

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;
}
```