ZADAĆA 2

Objektno orijentirano programiranje

Upute:

Zadaću je potrebno predati do 13. ožujka u 08:00 na Teamsu. Diskutiranje zadaće u okviru "study group" je dozvoljen, ali predana zadaća mora biti samostalno riješena. Studenti koji predaju zadaću obavezni su prisustvovati vježbama, u suprotnome zadaća neće biti bodovana. Ako se kod ne prevodi (neovisno o tipu compiler-a), ili se događa greška prilikom izvršavanja koda, zadaća neće biti bodovana.

Zadatak 1. (70 bodova) Dinamičko polje

Klasa mojVektor implementirana na predavanju definira neke od metoda po uzoru na klasu std::vector. Proučite klasu std::vector na ovome linku. Klasa std::vector primjer je strukture podataka koja se naziva dinamičko polje (kako je rečeno na predavanju). Na osnovu sljedeće deklaracije klase `MyVector`, implementirajte metode koje nisu implementirane na predavanju po uzoru na istoimene metode iz std::vector. Vodite računa da su članovi P, `_size` i `_capacity` označeni private modifikatorom pristupa, odnosno, njima je moguće pristupiti isključivo unutar same klase.

Proširite funkcionalnost metode `pushBack` koju smo implementirali na predavanju na način da se polje P dinamički alocira (i realocira) u ovisnosti o veličini i kapacitetu (`_size` i `_capacity` atributi) (). Npr. ako je trenutni kapacitet `_capacity` = 4 i trenutna veličina `_size` = 4 i odlučimo se unijeti novi element pomoću push_back metode, novi kapacitet bit će `_capacity` = 8, a veličina `_size` = 5. Ako je npr. naš kapacitet `_capacity` = 16, a veličina `_size` = 9 i ako se odlučimo ukloniti element, nova će veličina biti `_size` = 8, a kapacitet `_capacity` = 8 (Slika 1, na kraju zadaće).

```
class MyVector {
private:
    unsigned int _size, _capacity;
    int* P;
public:
    MyVector();
    MyVector(const MyVector&);
    ~MyVector();

    void pushBack(int);
```

```
    int popBack();

    unsigned int getSize() const;
    unsigned int getCapacity() const;

    void print();

    bool empty() const;
    bool full() const;

    int& at(unsigned int pos);
    int& front();
    int& back();
};
```

Dinamički alocirajte objekt ove klase, popunite ga cijelim brojevima od 1 do 10, ispišite te ponovno dinamički alocirajte drugi objekt kopiranjem prvog alociranog objekta. Taj drugi objekt ispražnjavajte dok ne bude prazan te ga istovremeno ispisujte.

Zadatak 2. (30 bodova) Stog

Po uzoru na klasu predstavljenu na vježbama implementirajte `DynamicStack` – stog koji ne enkapsulira polje već std::vector, a naziva se dinamičkim budući da ne enkapsulira niz fiksne duljine (polje), već niz varijabilne duljine (vektor). Za takav stog nisu potrebni članovi `top` i `size` već je dovoljno koristiti se već enkapsuliranim članovima klase std::vector.

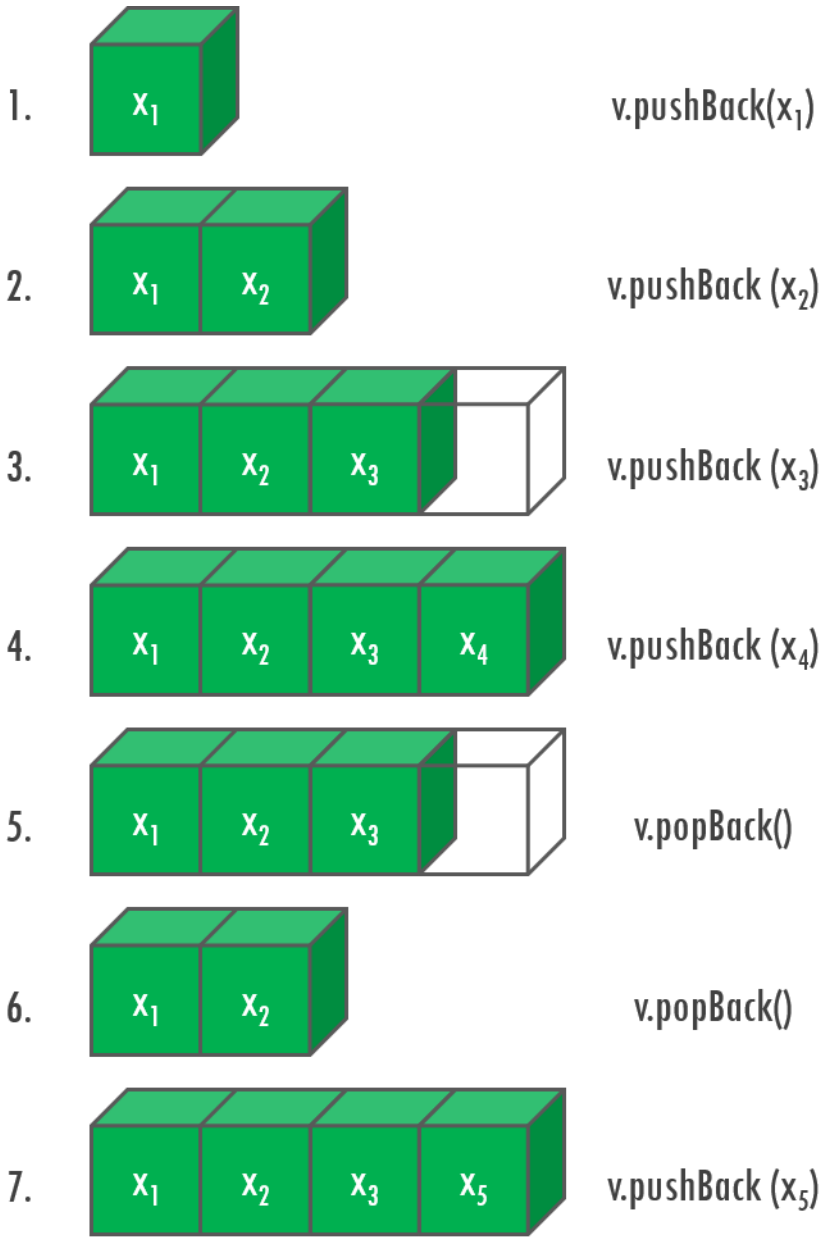
```
class DynamicStack {
private:
    vector<int> container;
public:
    DynamicStack();
    DynamicStack(const DynamicStack& S);

    bool empty() const;

    void push(int x);
    int pop();

    void print();
};
```

Dinamički alocirajte objekt ove klase, popunite ga cijelim brojevima od 1 do 10, ispišite te ponovno dinamički alocirajte drugi objekt kopiranjem prvog alociranog objekta. Taj drugi objekt ispražnjavajte dok ne bude prazan te ga istovremeno ispisujte.



Slika 1