**СУ “Св. Климент Охридски”,**

**ФМИ – Софтуерно инженерство**

**Курсов проект по Обектно-ориентирано програмиране**

**Вектор**

Веселина Валериева Стоянова, Факултетен № 61794

Съдържание

[1. Въведение 2](#_Toc420525966)

[2. Описание на приложените алгоритми 2](#_Toc420525967)

[3. Описание на програмния код 2](#_Toc420525968)

[4. Използвани технологии 7](#_Toc420525969)

# Въведение

Целта на проекта е да бъде реализиран шаблонен клас за динамичен масив – Vector и да бъде показана примерна употреба на реализирания клас. В тази документация ще разгледаме кратко описание на програмния код, което включва: основните методи използвани в проекта Vector и примерна употреба, и използвани техники за създаване на проекта.

Кодът на проекта е качен и в Github:

# Описание на приложените алгоритми

# Описание на програмния код

Стуктурата на класа е представена по долу:

template<typename T>

class Vector

{

public:

Vector();

Vector(size\_t count);

Vector(size\_t count, const T& value);

Vector(const Vector& other);

Vector& operator=(const Vector& other);

T& operator[](size\_t index)

{

return data[index];

}

size\_t capacity() const

{

return capacity\_;

}

size\_t size() const

{

return size\_;

}

void reserve(size\_t newCapacity);

void resize(size\_t newSize);

T& back();

T& front();

void shrink\_to\_fit();

void swap(Vector<T> &other);

void insert(size\_t index, const T& value);

void erase(size\_t index);

void print() const;

void push\_back(const T& value);

void pop\_back();

bool empty();

void clear();

~Vector();

private:

size\_t capacity\_;

size\_t size\_;

T\* data;

Създаден е шаблонен клас Vector, в който са имплементирани различни методи, чрез които се реализира векторите като изключително удобен и ефективен заместител на стандартните масиви.

Програмният код се състои от .h файл, в които се декларира шаблонен клас Vector и .cpp файл, в който са дефинициите на функциите. Също така има и .h файл, който е създаден клас Animal и който създава животни с техните характеристики, чрез които се показва употребата на методите на вектора.

В класа Vector са дефинирани:

1) Конструктор по подразбиране: Vector();

Създава празен вектор с капацитет 0 и брой елементи 0.

2) Конструктор за запълване: Vector(size\_t count);

Създава вектор с капацитет count и брой елементи count.

3) Конструктор за запълване: Vector(size\_t count, const T& value);

Създава вектор с капацитет cout и брой елементи cout.

(value - стойност с която да се запълни векторът)

4) Конструктор за копиране: Vector(const Vector& other);

Създава вектор, който съдържа копие на всеки от елементите на other.

5)Оператор за присвояващо копиране: Vector& operator=(const Vector& other);

Копира данните от векторът other.

Заделянето на памет се извършва чрез помощна функция: allocateMemory(size\_t capacityAlloc): Заделянето на паметта става чрез operator new като връща указател към нея.

Добавени са и други помощни функции:

copyRange(T\* begin, T\* end, T\* destination): Копира елементите от посочения обхват (опеределн от begin и end) в destination.

constructRange(T\* begin, T\* end): Конструира с конструктор по подразбиране обектите върху паметта , определена от обхвата begin и end.

constructRange(T\* begin, T\* end, const T& fillValue): Конструира с конструктор, приемащ стойност fill\_value ,oбектите върху паметта, определена от обхвата begin и end.

deleteRange(T\* begin, T\* end):Извиква деструкторите на обектите върху паметта, определна от обхвата begin и end.

***Описание на основните функции на класа Vector:***

Функцията push\_back променя размера на вектора, добавяйки един елемент със стойност value след текущия последен елемент .

Друга член-функция pop\_back отстранява последния елемент на вектор, като намалява размера му с единица. Забележете, че функцията pop\_back не връща отстранения елемент.

Функциите могат да модифицират вектор. Могат да изтрият елементите на вектор, да се преоразмеряват или да се модифицират отделни елементи.

Функцията resize() и shrink\_to\_fit() преоразмеряват вектора. Resize променя размера на вектора, така, че да съдържа нов брой елемента, а shrink\_to\_fit свива вектора, така,че размера да се побере в капацитета.

Erase() изстрива елемент на позиция index, а insert вмъква елемент със стойност value на позиция index.

Функцията swap ( ) , разменя стойностите на двата аргумента . Swap ( ) трябва да бъде извикана със адреса на аргументите .

Функцията empty() проверява дали векторът е празен. Функцията не модифицира. За изчистване на съдържанието на вектора се използва функцията clear().

Функциите, които връщат референции са съответно front(),back(), operator[], където съответно: back()-връща референция към елемента в края на вектора,back()-връща елемента в началото на вектора,Operator[]връща референция към елемента на позиция index.

Член-променливи на класа Vector:

size\_t capacity\_-Капацитет на вектора

size\_t size\_-Размер на вектора

***Примерна употреба:***

***int main()***

***{***

***Animal a1("Mammal", "Lion", "Grassland", 5);***

***Animal a2("Bird", "Pengiun", " Antarctica", 1);***

***Animal a3("Reptiles", "Snake", "Desert", 2);***

***Animal a4("Aquatic Animal", "Dolphin", "Sea", 7);***

***Vector<Animal> animals;***

***animals.push\_back(a1);***

***animals.push\_back(a2);***

***animals.push\_back(a3);***

***animals.push\_back(a4);***

***Vector<int> v1;***

***v1.push\_back(1);***

*v****1.push\_back(3);***

***v1.push\_back(5);***

***v1.push\_back(7);***

***v1.push\_back(9);***

***v1.push\_back(11);***

***v1.push\_back(13);***

***v1.pop\_back();***

***v1.push\_back(777);***

***v1.insert(5, 100);***

***v1.erase(6);***

***v1.print();***

***cout << endl;***

***cout << v1[4] << endl;***

***Vector<string> Penguin;***

***Penguin.push\_back("Penguins");***

***Penguin.push\_back("were once");***

***Penguin.push\_back("able to");***

***Penguin.push\_back("fly");***

***for (int i = 0; i < Penguin.size(); i++)***

***{***

***cout << Penguin[i] << " ";***

***}***

***cout << endl;***

***cout << Penguin.front()<<" "<<"don't"<<" "<< Penguin.back() << endl;***

***Vector<string> Lion;***

***Lion.push\_back("Lion");***

***Lion.push\_back("is common symbols");***

***Lion.push\_back(" for royalty and stateliness");***

***Lion.insert(1, "The");***

***for (int i = 0; i < Lion.size(); i++)***

***{***

***cout << Lion[i] << " ";***

***}***

***cout << endl;***

***Lion.erase(1);***

***for (int i = 0; i < Lion.size(); i++)***

***{***

***cout << Lion[i] << " ";***

***}***

***cout << endl;***

***Vector<string> Snake;***

***Snake.push\_back("Snakes");***

***Snake.push\_back("are");***

***Snake.push\_back("dreadful");***

***Vector<string> Dolphin;***

***Dolphin.push\_back("Dolphins");***

***Dolphin.push\_back("are");***

***Dolphin.push\_back("lovely");***

***Snake.swap(Dolphin);***

***for (int i = 0; i < Snake.size(); i++)***

***{***

***cout << Snake[i] << " ";***

***}***

***cout << endl;***

***for (int i = 0; i < Dolphin.size(); i++)***

***{***

***cout << Dolphin[i] << " ";***

***}***

***cout << endl;***

***return 0;***

***}***

# Използвани технологии

Интегрираната среда за разработка, която съм използвала за моя проект е Microsoft Visual Studio 2013. Проектът е писан на програмния език C++.