**2022 Spring OOP Assignment Report**

과제 번호 : ASSN2\_prob1

학번 : 20210716

이름 : 최대현

Povis ID : daehyeonchoi

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.

독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.

문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

1. **프로그램 개요**

개발환경: Visual Studio 2022

Vector 자료형을 구현하는 코드이다. Vector class를 declaration한 헤더파일 “Vector.h”와 함수를 구현하는 cpp파일 “Vector.cpp”, 그리고 이이 파일들을 컴파일 하기 위한 test tool인 “main.cpp” 의 총 3가지로 구성되어 있다.

1. **프로그램의 구조 및 알고리즘**

.

**벡터 자료형**

C++의 vector는 C++ 표준라이브러리(Standard Template Library)에 있는 컨테이너로 사용자가 사용하기 편하게 정의된 class이다. ctor를 생성하면 메모리 heap에 생성되며 동적할당된다. 적인 측면에서 array에 비해 성능은 떨어지지만 메모리를 효율적으로 관리하고 예외처리가 쉬우며, 템플릿으로 정의되어서 다양한 자료형에 대해 사용할 수 있어서 빈번하게 사용된다.

**사용자 정의 자료형 설명**

Vector이라는 이름을 가진 class를 구현했다. Public과 private 세션으로 나뉘어져 있고, public에는 3가지의 멤버 변수가 있다. 또한, private 세션에는 문제에서 요구하는 멤버함수들을 구현하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

**Private**

**멤버 변수 설명**

int \_size: 벡터 자료형 속 array에 들어있는 데이터의 개수이다.

int \_capacity: 벡터 자료형의 용량이다. 즉, array가 받을 수 있는 데이터의 개수이다.

float\* arr: 벡터 자료형 속의 배열이다. Float 배열로 구현했다.

**Public**

**Datatype definition 설명**

Float 기반 벡터이기 때문에 자주 사용하며 상징성 있는 2가지의 변수를 typedef 선언으로 재정의했다.

**멤버 함수 설명**

1. Vector() : 기본 생성자로, 모든 멤버 변수를 0 혹은 NULL 로 초기화하는 함수이다. Arr은 NULL size와 capacity 는 0으로 초기화했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Vector(int size) : 데이터 크기와 메모리 할당량은 size 로 초기화하고 size 개수만큼 0으로 초기화된 데이터 (arr[i])를 저장했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Vector(int size, const float& init) : float init 으로 초기화된 데이터를 저장하고, 데이터 크기와 메모리 할당량을 매개변수 size로 초기화했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Vector(int size, const float\* init) : float 배열 init 으로 초기화된 데이터를 저장하고, 데이터 크기와 메모리 할당량은 매개변수 size로 초기화했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Vector(const Vector& v) : 복사 생성자로, v에 있는 내용을 deep-copy한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. ~Vector() : 소멸자이다. 할당된 메모리를 삭제한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. int capacity() const : \_capacity를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. int size() const : \_size를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. iterator begin() : 데이터의 맨 처음 주소를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. const\_iterator begin() const : 데이터의 맨 처음 주소를 리턴한다. (const 버전)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. iterator end() : 데이터의 맨 마지막 주소를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. const\_iterator end() const : 데이터의 맨 마지막 주소를 리턴한다. (const 버전)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. float& front() : 맨 처음 데이터를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. const float& front() const : 맨 처음 데이터를 리턴한다. (const 버전)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. float& back() : 맨 마지막 데이터를 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. const float& back() const : 맨 마지막 데이터를 리턴한다. (const 버전)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void allocate(int capacity) : 데이터에 할당된 메모리를 capacity 만큼 동적 할당한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void resize(int size) : size 만큼 데이터에 할당된 메모리를 동적 할당하며, 데이터 개수는 size 만큼 갖는다.

텍스트이(가) 표시된 사진

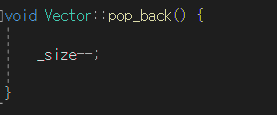
자동 생성된 설명

1. void push\_back(const float& value) : 데이터의 맨 뒤에 새로운 데이터 value 를 추가한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void pop\_back() : 맨 마지막 데이터를 삭제한다. 이 때, 메모리 할당은 유지한다.



1. void insert(iterator position, const float& value) : 데이터의 특정 위치(position)에 value 를 삽입한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void insert(iterator position, const float\* first, const float\* last) : 데이터에서 특정 위치(position)에 float array 를 삽입한다. 이 때 first, last 는 각각 float array 의 맨 처음 주소, 맨 마지막 주소이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void erase(iterator position) : 데이터에서 특정 위치(position)의 값을 삭제하고, 메모리 할당은 유지한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. void clear() : 벡터를 비워주는 함수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. float& operator[](int index) : 연산자 ‘[]’를 오버로딩한다. 즉, 벡터에 대해 []을 통해 배열의 인덱스값으로 접근할 수 있도록 지원한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. const float& operator[](int index) const : 연산자 ‘[]’를 오버로딩한다. 즉, 벡터에 대해 []을 통해 배열의 인덱스값으로 접근할 수 있도록 지원한다. (const 버전)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Vector& operator=(const Vector& v) : 연산자 ‘=’를 오버로딩하는 함수이다. 즉, 벡터 간의 대입연산을 지원해주는 함수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. **토론 및 개선**

이 과제를 시작할 당시 클래스에 대한 개념이 명확히 잡혀있지 않아서 어려움을 겪었고 많이 방황한 흔적이 보인다. 교수님의 수업을 듣고 시험 공부를 하다 보니 클래스에 대한 개념이 어느 정도 잡힌 시점에서 살펴보면, 나의 코드가 효율적이라고 하기는 어려운 것 같다. 반복되는 기능을 멤버함수로 사용할 수 있다던지, 다른 멤버함수를 불러와서 간단히 구현할 수 있다던지 하는 부분이 보여서 이런 부분을 개선해보고 싶다.

1. **참고 문헌**