ENSF 619- Fall 2020

Lab: 4

Ziad Chemali

October 18, 2020

# Exercise A:

## Code:

/\*

\*MyArray.cpp

\* Lab: 4 Exercise A

\* Completed by Ziad Chemali

\* Submission Date: October 18,2020

\*/

#include "MyArray.h"

#include <iostream>

MyArray::MyArray() {

sizeM = 0;

storageM = new EType[0];

}

MyArray::MyArray(const EType\* builtin, int sizeA) {

if (sizeA >= 0 && builtin != nullptr) {

sizeM = sizeA;

delete[] storageM;

storageM = new EType[sizeM];

for (int i = 0;i < sizeA ;i++) {

storageM[i] = builtin[i];

}

}

else

std::cout << "Error, cant copy an empty array\n";

}

MyArray::MyArray(const MyArray& source) {

if (source.storageM !=nullptr) {

delete[] storageM;

sizeM = source.sizeM;

storageM = new EType[sizeM];

for (int i = 0;i < source.sizeM ;i++) {

storageM[i] = source.storageM[i];

}

}

}

MyArray& MyArray :: operator =(const MyArray& rhs) {

if (this != &rhs)

{

delete[] storageM;

sizeM = rhs.sizeM;

storageM = new EType[sizeM];

for (int i = 0;i < rhs.sizeM ;i++) {

storageM[i] = rhs.storageM[i];

}

}

return \*this;

}

MyArray:: ~MyArray() {

delete[] storageM;

}

int MyArray::size() const {

return this->sizeM;

}

EType MyArray::at(int i) const {

return this->storageM[i];

}

void MyArray::set(int i, EType new\_value)

{

if (i >= 0 && i < this->size()) {

storageM[i] = new\_value;

}

}

void MyArray::resize(int new\_size)

{

if (new\_size >= 0) {

sizeM = new\_size;

EType \*temp = new EType[new\_size];

for (int i = 0;i < new\_size ;i++) {

temp[i] = this->storageM[i];

}

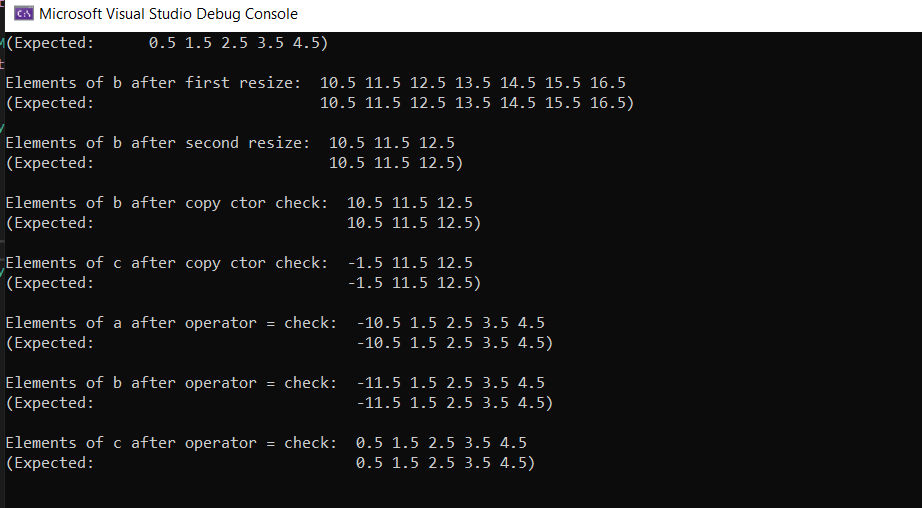
delete[] storageM;

storageM = temp;

}

}

## Output:



# Exercise B:

## Code:

String\_Vector transpose (const String\_Vector& sv) {

String\_Vector vs;

vs.resize(sv.at(0).size());

for (int i = 0; i < sv.size();i++) {

for (int j = 0;j < sv.at(i).size();j++)

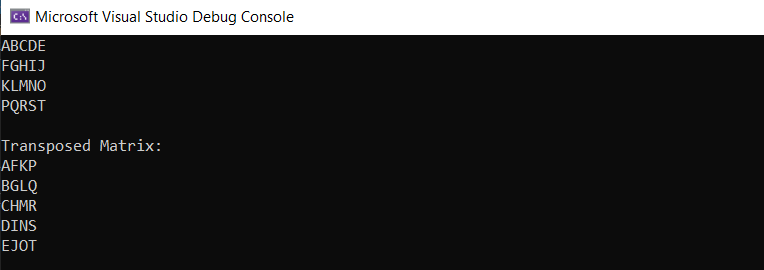
vs.at(j).push\_back(sv.at(i).at(j));

}

return vs;

}

## Output:



# Exercise C:

## Code:

void print\_from\_binary(char\* filename) {

ifstream stream(filename, ios::in | ios::binary);

if (stream.fail()) {

cerr << "failed to open file: " << filename << endl;

exit(1);

}

City temp;

while (!stream.eof()) {

stream.read((char\*)(&temp), sizeof(City));

cout << "x= "<<temp.x <<", y= "<< temp.y << ", City= "<< temp.name << endl;

}

stream.close();}

## Output:

