**Actividad 3.3: Memoria Dinamica en atributos de clase  
CECS2222 Computer Programming II**

Nombre: Group 6 Sec: \_22

**Instrucciones para el estudiante:**1. Diseñar una clase, donde sus atributos son apuntadores  
2. Implementar la función miembro copy constructor,  
3. Crear un arreglo dinámico unidimensional formado por objetos de la clase.  
4. Tabla Descriptiva  
5. Diagrama UML  
6. Salida de programa con los datos originales del problema.  
7. Envíe su solución en formato PDF.

**Descripción de los problemas:**  
1. (Gaddis) Programming Challenger 10. Number Array Class, pág. 805, Cap 13.

1. Use de ejemplo la clase IntegerList pág. 782 Cap 13.  
   2. Pruebe su programa con el siguiente arreglo <38,64,18,45,29,44,95>

Valor Total = 100 pts.

|  |  |
| --- | --- |
| **Variable** | **Descripción** |
| \*array | Variable tipo puntero que guarda valores de tipo float. |
| ArrSIZE | Variable tipo entero que guarda el valor de elementos del arreglo |
| **Función Miembro** | **Descripción** |
| NumArray() | Constructor por defecto es iniciado cuando no hay parámetros en la definición de objeto de la clase NumArray. |
| NumArray(int SIZE) : | Constructor con parámetros inicial de tipo int. |
| NumArray(obj : const NumArray&) : | Copy Constructor con parámetro inicial de objeto. |
| ~NumArray () : | Función destructor que libera memoria dinámica. |
| setElement(numElem: int, value: float) : void | Función que guarda los elementos del arreglo. |
| getElement(numElem: int) : float | Función que retorna elementos del arreglo. |
| findHighestValue() : float | Función que retorna el valor mas alto del arreglo. |
| findLowestValue() : float | Función que retorna el valor mas bajo del arreglo. |
| findAverage() : float | Función que retorna el valor promedio del arreglo. |
| bubbleSort() : void | Función que sortea el arreglo de menor a mayor. |
| display(n: int) : int | Función que imprime el arreglo. |

|  |
| --- |
| **NumArray** |
| -\*array: float  -ArrSIZE : int |
| +NumArray() :  +NumArray(int SIZE) :  +NumArray(obj : const NumArray&) :  + ~NumArray () :  +setElement(numElem: int, value: float) : void  +getElement(numElem: int) : float  +findHighestValue() : float  +findLowestValue() : float  +findAverage() : float  +bubbleSort() : void  +display(n: int) : int |

## **NumArray.h**

#pragma once

#include <iostream>

using namespace::std;

class NumArray {

private:

float\* array;

int ArrSIZE;

public:

//constructors

NumArray();

NumArray(int SIZE);

NumArray(const NumArray &obj);

//destructor

~NumArray();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Mutators member functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//member function that sets the number of elements in the array

void setElement(int numElem, float value);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Accessors member functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//member function that returns the number of elements in the array

float getElement(int numElem) const;

//member function that return the highest value of the array

float findHighestValue();

//member function that return the lowest value of the array

float findLowestValue();

//member function that sorts the array

void bubbleSort();

//member function tha displays the contents of the array

void display(int n) const;

};

## **NumArray.cpp**

#include <iostream>

#include "NumArray.h"

using namespace::std;

//constructors

NumArray::NumArray() {

//initialize to 0

const int SIZE = 10;

array = new float[SIZE];

ArrSIZE = SIZE;

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

array[i] = 0;

}//end for

}

NumArray::NumArray(int SIZE) {

ArrSIZE = SIZE;

array = new float[SIZE];

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

array[i] = 0;

}//end for

}

NumArray::NumArray(const NumArray &obj) {

ArrSIZE = obj.ArrSIZE;

array = new float[ArrSIZE];

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

array[i] = obj.array[i];

}//end for

}

//destructor

NumArray::~NumArray() {

delete[] array;

array = nullptr;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Mutators member functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//member function that sets the number of elements in the array

void NumArray::setElement(int numElem, float value) {

\*(array + numElem) = value;

}

//member function that returns the number of elements in the array

float NumArray::getElement(int numElem) const {

return array[numElem];

}

//member function that returns the highest value of the array

float NumArray::findHighestValue() {

bubbleSort();

return array[ArrSIZE - 1];

}

//member function that returns the lowest value of the array

float NumArray::findLowestValue() {

bubbleSort();

return array[0];

}

//member function that sorts the array

void NumArray::bubbleSort() {

bool swap;

int temp;

do{

swap = false;

for (int count = 0; count < (ArrSIZE - 1); count++){

if (array[count] > array[count + 1]){

temp = array[count];

array[count] = array[count + 1];

array[count + 1] = temp;

swap = true;

}//end if

}//end for

} while (swap);

}

//member function that displays the contents of the array

void NumArray::display(int n) const {

cout << getElement(n) << " ";

}

## **Main.cpp**

#include <iostream>

#include "NumArray.h"

using namespace::std;

//prototypes

int arraySize();

void storeValuesArray(NumArray&, int);

void displayArr(NumArray, int);

int main() {

int SIZE;

int highestValue;

float value = 0;

//ask user for the number of elements in the array

SIZE = arraySize();

//create NumArray object

NumArray myNumArray(SIZE);

//store values in array

storeValuesArray(myNumArray, SIZE);

cout << endl;

//display the array elements

displayArr(myNumArray, SIZE);

cout << endl;

//find the highest value of the array

cout << "The Highest Value is: " << myNumArray.findHighestValue() << endl;

//find the highest value of the array

cout << "The Lowest Value is: " << myNumArray.findLowestValue() << endl;

//print the sorted array

cout << "The sorted array is: ";

displayArr(myNumArray, SIZE);

cout << endl;

return 0;

}

//function that ask user the size of the array

int arraySize () {

int arrSize;

cout << "Enter the amount of elements for the array." << endl;

cin >> arrSize;

while (arrSize <= 0.0) {

cout << "Invalid entry please try again." << endl;

cin >> arrSize;

}//end while

return arrSize;

}

//function that stores the values entered from the user to the array

void storeValuesArray(NumArray& myNumArray, int SIZE) {

float value;

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

cout << "Enter value #" << i + 1 << ": " << endl;

cin >> value;

myNumArray.setElement(i,value);

}//end for

}

//funtion that displays the array

void displayArr(NumArray myNumArray, int SIZE) {

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

myNumArray.display(i);

}//end for

}

## **Salida**

Text

Description automatically generated

Text

Description automatically generated