```
/*
	Alaya, Anas

	CS A200
	August 28, 2017

	Lab 1
*/

#include "DArray.h"

// Definition function emptyArray
void DArray::emptyArray()
{
	this->numOfElements = 0;
}

// Definition function appendArray
void DArray::appendArray(const DArray& obj)
{
	if (obj.getNumOfElements() != 0)
	{
		const int TOTAL_SIZE = obj.getNumOfElements() + this->getNumOfElements();
		if (this->capacity < TOTAL_SIZE)
		{
			int *temp = new int[TOTAL_SIZE];
			for (int i = 0; i < this->getNumOfElements(); i++)
				temp[i] = a[i];

			int x = this->getNumOfElements();
			for (int i = 0; i < obj.getNumOfElements(); i++)
			{
				temp[x] = obj.a[i];
				x++;
			}
			delete[] a;
			a = temp;
			temp = nullptr;
			numOfElements = TOTAL_SIZE;
		}
		else
		{
			for (int i = 0; i < obj.getNumOfElements(); i++)
```

```
                        {
                                addElement(obj.a[i]);
                        }
                }
        }
        //else
        //        cout << "Param is Empty!!!" << endl;
}

// Definition move constructor
DArray::DArray(DArray&& obj)
{
        capacity = obj.capacity;
        a = obj.a;
        numOfElements = obj.getNumOfElements();

        // free the param obj
        obj.capacity = 0;
        obj.a = nullptr;
        obj.numOfElements = 0;

}

// Definition move assignment operator
DArray& DArray::operator=(DArray&& obj)
{
        if (this != &obj)
        {
                // Free the calling obj.
                delete[] a;

                // Copy from param
                a = obj.a;
                capacity = obj.capacity;
                numOfElements = obj.numOfElements;


                // Release  param
                obj.a = nullptr;
                obj.capacity = 0;
                obj.numOfElements = 0;
        }
        return *this;
```

```
}
```