|  |
| --- |
| **2019년 11월 14일 실습보고서** |
| **7조 : 송재원, 조윤직, 양석준, 박진영** |
| **실습자료1 : arraydata 이동연산자 복사연산자** |
| **소스코드** |
| #include "ArrayData.h"  using namespace std;  int main()  {  cout << "7조 박진영 양석준 송재원 조윤직\n";  ArrayData data(10);  data.addElement(10.0);  data.addElement(10.1);  data.addElement(10.2);  data.addElement(10.3);  data.showData();    ArrayData data2;  data2 = data;  data2.showData();  ArrayData data3;  data3 = ArrayData(5);  data3.showData();  }  ----------------------------------------------------------------------------  ArrayData.h  #pragma once  class ArrayData  {  private:  double\* data;  int capacity;  int used;  static int count;  public:  static int getCount();  ArrayData(const int& capacity);  ArrayData(const ArrayData& arr);  ArrayData(ArrayData&& arr);    ArrayData();  ~ArrayData();  void addElement(double num);  bool full() const;  int getCapacity() const;  int getUsed() const ;  void emptyArray();  void showData() const;  void operator=(const ArrayData& copy);  void operator=( ArrayData&& copy);  friend double getArraySum(const ArrayData& arr);  ArrayData getObject() {  return \*this;  }  static ArrayData getCopyInstance(const ArrayData&copy) ;  };  ----------------------------------------------------------------------------  ArrayData.cpp  #include "ArrayData.h"  #include<iostream>  using namespace std;  int ArrayData::getCount()  {  return count;  }  ArrayData::ArrayData(const int& capacity):capacity(capacity), used(0)  {  cout << this->capacity << "배열 생성\n";  count++;  this->data = new double[capacity];  }  ArrayData::ArrayData(const ArrayData& arr) : capacity(arr.capacity),used(arr.used)  {  cout << this->capacity << "배열 복사\n";  count++;  this->data = new double[this->capacity];  for (int i = 0; i < used; i++)  data[i] = arr.data[i];  }  ArrayData::ArrayData(ArrayData&& copy) :capacity(copy.capacity), used(copy.used), data(copy.data)  {  cout << this->capacity << "이동 생성자\n";  copy.data = nullptr;  }  ArrayData::ArrayData() :ArrayData(5)  {  }  int ArrayData::count = 0;  ArrayData::~ArrayData()  {    cout << this->capacity << "배열 삭제\n";  count--;  if (data != nullptr) {  delete[] this->data;  data = nullptr;  }    }  void ArrayData::addElement(double num)  {  if (!full())  this->data[used++] = num;  else  cout << "빈 공간이 없음\n";    }  bool ArrayData::full() const  {  return capacity==used;  }  int ArrayData::getCapacity() const  {  return this->capacity;  }  int ArrayData::getUsed() const  {  return this->used;  }  void ArrayData::emptyArray()  {  this->used = 0;  }  void ArrayData::showData() const  {  double\* p = this->data;  double\* endp = p + this->used;  cout << "배열 출력 :";  while (p < endp) {  cout << \*(p++)<<" ";    }  cout << endl;  }  void ArrayData::operator=(const ArrayData& copy)  {  if (capacity != copy.capacity) {  delete data;  capacity = copy.capacity;  data = new double[copy.capacity];  }  emptyArray();  for (int i = 0; i < copy.used;i++) {  addElement(\*(copy.data + i));  }  }  void ArrayData::operator=( ArrayData&& copy)  {  cout << "이동 연산자 수행\n";  if (data != NULL) {  delete data;  }  capacity = copy.capacity;  used = copy.used;  data = copy.data;  copy.data = NULL;  }  ArrayData ArrayData::getCopyInstance(const ArrayData& copy)  {  ArrayData arr(copy);  return arr;  } |
| **실행결과** |
|  |