|  |
| --- |
| **2019년 11월 14일 실습보고서** |
| **7조 송재원 조윤직 박진영 양석준** |
| **실습자료1 : [연산자 오버로딩]** |
| **소스코드** |
| // ArrayData.h  #pragma once  class ArrayData  {  private:  double\* data;  int capacity; // 최대용량  int used; //현재용량  static int count;  public:  static int getCount();  ArrayData();  ~ArrayData();  ArrayData(const int& capacity);  ArrayData(const ArrayData &copy); //반드시레퍼런스고 복사생성자만들어줌 깊은 복사해주려고  //const는 복사된거 안바꿀거라는의미  ArrayData(ArrayData &&copy); // 이동생성자 const이면 곤란하다  void addElement(double num);  bool full() const;  int getCapacity() const;  int getUsed() const;  void emptyArray();  void showData() const;  friend double getArraySum(const ArrayData& arr);  ArrayData getObject() {//this 를 넘기면 포인터타입 this가 가리키는 값을 넘겨줘야함  return \*this;  }  static ArrayData getCopyInstance(const ArrayData& copy) { // 복사생성자 만들어놈  ArrayData arr(copy);  return arr;  }  void operator=(const ArrayData & AD); // 대입연산자  void operator=(ArrayData&& AD); // 이동대입연산자  };  //ArrayData.cpp  #include "ArrayData.h"  #include <iostream>  using namespace std;  int ArrayData::count = 0; // static변수는 cpp에서 이렇게 초기화함  int ArrayData::getCount()  {  return count;  }  ArrayData::ArrayData() //디폴트로 방5개만잡는다고할때  :ArrayData(5) //이니셜라이저  {  }  ArrayData::~ArrayData() //소멸자  {  if (data != nullptr) { // 정상적인 메모리를 가질때만 지우면됨  delete[] data;  data = nullptr; //nullptr먼데  }  count--;  cout << this->capacity << " 배열 삭제" << endl;  }  ArrayData::ArrayData(const int& capacity)  :capacity(capacity), used(0) //맴버 초기화 이니셜~  {  cout << this->capacity << " 배열 생성" << endl;  count++;  this->data = new double[capacity];  }  ArrayData::ArrayData(const ArrayData & copy) //복사생성자 깊은 복사 수행  :capacity(copy.capacity), used(copy.used) //멤버 접근가능  {  cout << this->capacity << " 배열 복사" << endl;  count++;  this->data = new double[this->capacity];  for (int i = 0; i < used; i++) {  data[i] = copy.data[i];  }  }  ArrayData::ArrayData(ArrayData && copy) //얕은 복사 수행  :capacity(copy.capacity), used(copy.used), data(copy.data)  {  copy.data = nullptr;  cout << "이동 생성자" << endl;  }  void ArrayData::addElement(double num) {  if (!full()) {  data[used++] = num;  }  else {  cout << "빈공간이 없음" << endl;  }  }  bool ArrayData::full() const {  return capacity == used;  }  int ArrayData::getCapacity() const {  return this->capacity; //capacity 와 다름 this에해당하는 capacity(내꺼말하는거)거고  //그냥 capacity는 받은거말하는거임  }  int ArrayData::getUsed() const {  return this->used;  }  void ArrayData::emptyArray() {  used = 0;  }  void ArrayData::showData() const  {  for (int i = 0; i < used; i++) {  cout << data[i] << " ";  }  cout << endl;  }  void ArrayData::operator=(const ArrayData & copy)  {  cout << "=대입연산자 실행" << endl;  if (capacity != copy.capacity) {  delete data;  capacity = copy.capacity;  data = new double[copy.capacity];  }  used = 0;  for (int i = 0; i < copy.used; i++) {  addElement(copy.data[i]);  }  return;  }  void ArrayData::operator=(ArrayData && AD)  {  cout << "=이동생성자 실행" << endl;  capacity = AD.capacity;  used = AD.used;  data = AD.data;  AD.data = nullptr;  }  //testmain.cpp  #include <iostream>  #include "ArrayData.h"  using namespace std;  void showArray(ArrayData& arr) { //call by value  arr.showData();  }  double getArraySum(const ArrayData& arr) {  double total = 0;  for (int i = 0; i < arr.getUsed(); i++) {  total += arr.data[i];  }  return total;  }  int main() {  cout << "7조 조윤직 송재원 양석준 박진영 입니다." << endl;  ArrayData arr(10);  ArrayData arr2 = ArrayData::getCopyInstance(arr);    ArrayData data(10);  data.addElement(10.0);  data.addElement(20.0);  data.addElement(30.0);  data.addElement(40.0);  data.addElement(10.0);  data.showData();  ArrayData data2;  data2 = data;  data.showData();  data2.showData();    ArrayData data3;  data3 = ArrayData(10);  system("pause");  return 0;  } |
| **실행결과** |
|  |