|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2019\_1\_C++ \_06 | 학번 : | 20145165 | 이름 : | 정균모 |

* **강의 내용**

|  |
| --- |
| 1. 복사 생성자 |
| #include <iostream>  using namespace std;  class Circle {  private:  int radius;  public:  Circle(Circle& c); // 복사 생성자 선언  Circle() { radius = 1; }  Circle(int radius) { this->radius = radius; }  double getArea() { return 3.14\*radius\*radius; }  };  Circle::Circle(Circle& c) { // 복사 생성자 구현  this->radius = c.radius;  cout << "복사 생성자 실행 radius = " << radius << endl;  }  int main() {  Circle src(30); // src 객체의 보통 생성자 호출  Circle dest(src); // dest 객체의 복사 생성자 호출  cout << "원본의 면적 = " << src.getArea() << endl;  cout << "사본의 면적 = " << dest.getArea() << endl;  } |
| **[실행결과]** |

|  |
| --- |
| 1. 깊은 복사 |
| #include <iostream>  #include <cstring>  using namespace std;  class Person { // Person 클래스 선언  char\* name; int id;  public:  Person(int id, const char\* name); // 생성자  Person(Person& person); // 복사 생성자  ~Person(); // 소멸자  void changeName(const char \*name);  void show() { cout << id << ',' << name << endl; }  };  Person::Person(int id, const char\* name) { // 생성자  this->id = id;  int len = strlen(name); // name의 문자 개수  this->name = new char[len + 1]; // name 문자열 공간 핟당  strcpy(this->name, name); // name에 문자열 복사  }  **Person::Person(Person& person) { // 복사 생성자**  **this->id = person.id; // id 값 복사**  **int len = strlen(person.name);// name의 문자 개수**  **this->name = new char[len + 1]; // name을 위한 공간 핟당**  **strcpy(this->name, person.name); // name의 문자열 복사**  **cout << "복사 생성자 실행. 원본 객체의 이름 " << this->name << endl;**  **}**  Person::~Person() {// 소멸자 코드 생략  }  void Person::changeName(const char\* name) { // 이름 변경  if (strlen(name) > strlen(this->name))  return; // 현재 name에 할당된 메모리보다 긴 이름으로 바꿀 수 없다.  strcpy(this->name, name);  }  int main() {  Person father(1, "Kitae"); // (1) father 객체 생성  Person daughter(father); // (2) daughter 객체 복사 생성. 복사생성자호출  cout << "daughter 객체 생성 직후 ----" << endl;  father.show(); // (3) father 객체 출력  daughter.show(); // (3) daughter 객체 출력  daughter.changeName("Grace"); // (4) daughter의 이름을 "Grace"로 변경  cout << "daughter 이름을 Grace로 변경한 후 ----" << endl;  father.show(); // (5) father 객체 출력  daughter.show(); // (5) daughter 객체 출력  return 0; // (6), (7) daughter, father 객체 소멸  } |
| **[실행결과]** |

|  |
| --- |
| 1. 디폴트 매개변수 |
| #include <iostream>  using namespace std;  class MyVector{  int \*p;  int size;  public:  MyVector(int n=100) {  p = new int [n];  size = n;  }  ~MyVector() { delete [] p; }  };  int main() {  MyVector \*v1, \*v2;  v1 = new MyVector(); // 디폴트로 정수 배열 100 동적 할당  v2 = new MyVector(1024); // 정수 배열 1024 동적 할당  delete v1;  delete v2;  } |
| **[실행결과]**  V1, v2 라는 MyVector 형 객체를 만들고 v1 = new MyVector(); 생성자를 불러와 동적 할당을 한다.  100과 1024만큼 동정 할당을 하고 delete 작업을 실행 해 준다. |

|  |
| --- |
| 1. static 멤버 사용하기 |
| class Person {  public:  double money; // 개인 소유의 돈  void addMoney(int money) { this->money += money; }  static int sharedMoney; // 공금  static void addShared(int n) {  sharedMoney += n;  }  };  // static 변수 생성. 전역 공간에 생성  int Person::sharedMoney=10; // 10으로 초기화  int main() {  // static 멤버 접근, 공금=60, han 객체가 생기기전부터 static 멤버접근  **Person::addShared(50);**  cout << **Person::sharedMoney** << endl;  Person han;  han.money = 100;  han.sharedMoney = 200; // static 멤버 접근, 공금=200  **Person::sharedMoney = 300;** // static 멤버 접근, 공금=300  **Person::addShared(100);** // static 멤버 접근, 공금=400  cout << han.money << ' ' << **Person::sharedMoney** << endl;  } |
| **[실행결과]** |

* 내용 점검 – 6장 연습문제

|  |  |
| --- | --- |
| 번호 | 문제풀이 |
| 1 | 2 |
| 2 | 3 |
| 3 | 3,4 |
| 4 | 1,2 |
| 5 | 2 |
| 6 | 2 |
| 7 | 4 |
| 8 | 2 |
| 9 | 3 |
| 10 | 2 |
| 11 | 4 |
| 12 | 0 , 3 , 5,hello |
| 13 | 디폴트생성자 때문에 |
| 14 | Int형 정수가 전달될 경우, 매개변수의 형변환이 모호 |
| 15 | 4 static에서는 int형 함수를 사용할 수 없다. |
| 16 | 3 |

* 프로그램 과제

|  |
| --- |
| 1. 교재 274p 문제11번 |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Book {  string s;  int price;  public:  Book(string s, int price);  ~Book();  /\*  Book(Book& b) { 디폴트 복사 생성자  this->title = title;  thie->price = price;  }  \*/  Book(Book &b);  void set(string s, int price);  void show() {  cout << s << ' ' << price << "원" << endl;  }  };  Book::Book(string s, int price) {  this->s = s;  this->price = price;  }  Book::~Book() {  }  void Book::set(string s, int price) {  this->s = s;  this->price = price;  }  Book::Book(Book &b) {  this->s = b.s;  this->price = b.price;  }  int main() {  Book cpp("명품C++", 10000);  Book java = cpp;  java.set("명품자바", 12000);  cpp.show();  java.show();  system("pause");  } |
| **[실행결과]** |

|  |
| --- |
| 1. 교재 275p 문제12번 |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  class Dept {  int size;  int\* scores;  public:  Dept(int size) {  this->size = size;  scores = new int[size];  }  ~Dept();  int getSize() { return size; }  void read();  bool isOver60(int index);  };  Dept::~Dept() {  delete[] scores;  }  void Dept::read() {  cout << size << "개 점수 입력>>";  for (int i = 0; i < size; ++i) {  int n;  cin >> n;  scores[i] = n;  }  }  bool Dept::isOver60(int index) {  if (scores[index] > 60) {  return true;  }  else {  false;  }  }  int countPass(Dept& dept) {  int count = 0;  for (int i = 0; i < dept.getSize(); ++i) {  if (dept.isOver60(i) == true) {  count++;  }  }  return count;  }  int main() {  Dept com(10);  com.read();  int n = countPass(com);  cout << "60점 이상은 " << n << "명"<<endl;  system("pause");  } |
| **[실행결과]**  **복사 생성자가 호출되는 부분은 int n = countPass(com); 이 부분이다. 복사 생성자가 없으면 디폴트 복사생성자가 자동으로 호출 되는데 컴파일러는 얕은 복사를 하는 복사생성자에서 동적 메모리를 할당 받은 맴버가 있을 때 에러가 난다.** |

|  |
| --- |
| 1. 교재 313p 문제2번 |
| **[프로그램 소스]**  (1)  #include <iostream>  #include <string>  using namespace std;  class Person {  int id;  double weight;  string name;  public:  Person() { id = 1; weight = 20.5; name = "Grace"; }  Person(int id, string name) { this->id = id; weight = 20.5; this->name = name; }  Person(int id, string name, double w) { this->id = id; weight = w; this->name = name; }  void show() { cout << id << ' ' << weight << ' ' << name << endl; }  };  int main() {  Person grace, ashley(2, "Ashley"), helen(3, "Helen", 32.5);  grace.show();  ashley.show();  helen.show();  system("pause");  }  /////////  (2)  #include <iostream>  #include <string>  using namespace std;  class Person {  int id;  double weight;  string name;  public:  Person(int id = 1, string name = "Grace", double w = 20.5)  {  this->id = id;  weight = w;  this->name = name;  }  void show() { cout << id << ' ' << weight << ' ' << name << endl; }  };  int main() {  Person grace, ashley(2, "Ashley"), helen(3, "Helen", 32.5);  grace.show();  ashley.show();  helen.show();  } |
| **[실행결과]**  **1**    2 |

|  |
| --- |
| 1. 교재 313p 문제3번 |
| **[프로그램 소스]**  **1**  #include <iostream>  using namespace std;  int big(int a, int b) {  if (a > 100 || b > 100)  return 100;  else  {  if (a >= b) return a;  else return b;  }  }  int big(int a, int b, int c) {  if (a >= b) {  if (a > c) return c;  else return a;  }  else {  if (b > c) return c;  else return b;  }  }  int main() {  int x = big(3, 5);  int y = big(300, 60);  int z = big(30, 60, 50);  cout << x << ' ' << y << ' ' << z << endl;  system("pause");  }  2  #include <iostream>  using namespace std;  int big(int a, int b, int c = 100) {  if (a > c || b > c)  return c;  else  {  if (a >= b) {  if (a > c) return c;  else return a;  }  else {  if (b > c) return c;  else return b;  }  }  }  int main() {  int x = big(3, 5);  int y = big(300, 60);  int z = big(30, 60, 50);  cout << x << ' ' << y << ' ' << z << endl;  system("pause");  } |
| **[실행결과]**  **1**    2  결과는 위와 같다. |

|  |
| --- |
| 1. 교재 316p 문제6번 |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  class ArrayUtility2 {  public:  static int\* concat(int s1[], int s2[], int size) {  int\* p = new int[size];  int n = size / 2;  for (int i = 0; i < n; i++) {  p[i] = s1[i];  p[i + n] = s2[i];  }  return p;  }  static int\* remove(int s1[], int s2[], int size, int& retSize) {  for (int i = 0; i < size; i++) {  for (int j = 0; j < size; j++) {  if (s1[i] == s2[j]) s1[i] = NULL;  }  }  for (int i = 0; i < size; i++)  if (s1[i] != NULL) retSize++;  int \*p = new int[retSize];  int n = 0;  for (int i = 0; i < size; i++)  if (s1[i] != NULL) { p[n] = s1[i]; n++; }  if (retSize == 0) return NULL;  else return p;  }  };  int main() {  int x[5], y[5], retsize = 0, size;  cout << "정수를 5 개 입력하라. 배열 x에 삽입한다>>";  cin >> x[0] >> x[1] >> x[2] >> x[3] >> x[4];  cout << "정수를 5 개 입력하라. 배열 y에 삽입한다>>";  cin >> y[0] >> y[1] >> y[2] >> y[3] >> y[4];  cout << "합친 정수 배열을 출력한다." << endl;  size = sizeof(x) / 4 + sizeof(y) / 4;  int \*temp = ArrayUtility2::concat(x, y, size);  for (int i = 0; i < size; i++) cout << temp[i] << ' ';  cout << endl;  size = sizeof(x) / 4;  temp = ArrayUtility2::remove(x, y, size, retsize);  cout << "배열 x[]에서 y[]를 뺀 결과를 출력한다. 개수는 " << retsize << endl;  for (int i = 0; i < retsize; i++) cout << temp[i] << ' ';  cout << endl;  system("pause");  } |
| **[실행결과]** |

|  |
| --- |
| 1. 교재 316p 문제7번 |
| **[프로그램 소스]**  #include <iostream>  #include <cstdlib>  #include <ctime>  using namespace std;  class Random {  public:  static void seed() { srand((unsigned)time(0)); }  static int nextInt(int min = 0, int max = 32767) {  int n = min + (rand() % (max - min + 1));  return n;  }  static char nextAlphabet();  static double nextDouble();  };  char Random::nextAlphabet() {  int n = rand() % 2;  if (n == 0) {  n = 65 + rand() % 26;  return n;  }  else {  n = 97 + rand() % 26;  return n;  }  }  double Random::nextDouble() {  double n = rand() / (double)(RAND\_MAX + 1);  return n;  }  int main() {  Random::seed();  int i;  cout << "1에서 100까지 랜덤한 정수 10개를 출력합니다" << endl;  for (i = 0; i < 10; i++) cout << Random::nextInt(1, 100) << ' ';  cout << endl << "알파벳을 랜덤하게 10개를 출력합니다" << endl;  for (i = 0; i < 10; i++) cout << Random::nextAlphabet() << ' ';  cout << endl << "랜덤한 실수를 10개를 출력합니다" << endl;  for (i = 0; i < 5; i++) cout << Random::nextDouble() << ' ';  cout << endl;  for (i = 0; i < 5; i++) cout << Random::nextDouble() << ' ';  cout << endl;  system("pause");  } |
| **[실행결과]** |

|  |
| --- |
| 1. 교재 318p 문제9번 |
| **[프로그램 소스]**  #include <iostream>  #include <string>  #define ARRAY\_MAX 1000  using namespace std;  class Board {  static string \*textArray;  static int num;  public:  static void add(string text);  static void print();  };  void Board::add(string text) {  textArray[num] = text;  ++num;  }  void Board::print() {  cout << "\*\*\*\*\*\*\*\*\*\* 게시판입니다. \*\*\*\*\*\*\*\*\*\*" << endl;  for (int i = 0; i < num; ++i) {  cout << i << ": " << textArray[i] << endl;  }  }  string \*Board::textArray = new string[ARRAY\_MAX];  int Board::num = 0;  int main() {  Board::add("중간고사는 감독 없는 자율 시험입니다.");  Board::add("코딩 라운지 많이 이용해 주세요.");  Board::print();  Board::add("진소린 학생이 경진대회 입상하였습니다. 축하해주세요");  Board::print();  system("pause");  } |
| **[실행결과]** |