|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2018\_2\_C++ \_10 | 학번 : |  | 이름 : |  |

1. 상속 1

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  class Point {  protected:  int x, y; //한 점 (x,y) 좌표값  public:  void set(int x, int y) { this->x = x; this->y = y; }  void showPoint() {  cout << "(" << x << "," << y << ")" << endl;  }  };  class ColorPoint : public Point {  string color;  public:  void setColor(string color) { this->color = color; }  void showColorPoint();  bool equals(ColorPoint p);  };  void ColorPoint::showColorPoint() {  cout << color << ":";  showPoint(); // Point 클래스의 showPoint() 호출  }  bool ColorPoint::equals(ColorPoint p) {  if(x == p.x && y == p.y && color == p.color) // ①  return true;  else  return false;  }  int main() {  Point p; // 기본 클래스의 객체 생성  p.set(2,3); // ②  p.x = 5; // ③  p.y = 5; // ④  p.showPoint();  ColorPoint cp; // 파생 클래스의 객체 생성  cp.x = 10; // ⑤  cp.y = 10; // ⑥  cp.set(3,4);  cp.setColor("Red");  ColorPoint cp2;  cp2.set(3,4);  cp2.setColor("Red");  cout << ((cp.equals(cp2))?"true":"false"); // ⑦  } |
| **[실행 결과]** |

* 상속 2 - private

|  |
| --- |
| #include <iostream>  using namespace std;  class Base {  int a;  protected:  void setA(int a) { this->a = a; }  public:  void showA() { cout << a; }  };  class Derived : private Base {  int b;  protected:  void setB(int b) { this->b = b; }  public:  void showB() { cout << b; }  };  int main() {  Derived x;  x.a = 5; // ①  x.setA(10); // ②  x.showA(); // ③  x.b = 10; // ④  x.setB(10); // ⑤  x.showB(); // ⑥  } |
| **[실행 결과]** |

* 상속 3 - protected

|  |
| --- |
| #include <iostream>  using namespace std;  class Base {  int a;  protected:  void setA(int a) { this->a = a; }  public:  void showA() { cout << a; }  };  class Derived : protected Base {  int b;  protected:  void setB(int b) { this->b = b; }  public:  void showB() { cout << b; }  };  int main() {  Derived x;  x.a = 5; // ①  x.setA(10); // ②  x.showA(); // ③  x.b = 10; // ④  x.setB(10); // ⑤  x.showB(); // ⑥  } |
| **[실행 결과]** |

* 내용 점검

1. 교재 8장 연습문제

|  |  |
| --- | --- |
| 번호 | 문제풀이 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |
| 16 |  |
| 17 |  |

* 프로그램 과제

1. 교재 418p : 문제 1번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Circle {  int radius;  public:  Circle(int radius=0) { this->radius = radius; }  int getRadius() { return radius; }  void setRadius(int radius) { this->radius = radius; }  double getArea() { return 3.14\*radius\*radius; };  };  class NamedCircle : public Circle {  string name;  public:  NamedCircle(int radius, string name);  void show();  };  NamedCircle::NamedCircle(int radius, string name)  : Circle(radius) {  this->name = name;  }  void NamedCircle::show() {  cout << "반지름이 " << getRadius() << "인 " << name << endl;  } |
| **[실행 결과]** |

1. 교재 418p : 문제 2번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Circle {  int radius;  public:  Circle(int radius=0) { this->radius = radius; }  int getRadius() { return radius; }  void setRadius(int radius) { this->radius = radius; }  double getArea() { return 3.14\*radius\*radius; };  };  class NamedCircle : public Circle {  string name;  public:  NamedCircle(int radius, string name);  void set(int radius, string name){  setRadius(radius);  this->name = name;  }  string getName() { return name; }  void show();  };  NamedCircle::NamedCircle(int radius=0, string name="NONAME") : Circle(radius) {  this->name = name;  }  void NamedCircle::show() {  cout << "반지름이 " << getRadius() << "인 " << name << endl;  }  string biggest(NamedCircle p[], int n) {  if(n<=0)  return ""; // 오류  int big = 0;  for(int i=1; i<n; i++) {  if(p[i].getRadius() > p[big].getRadius())  big = i;  }  return p[big].getName();  }  int main() {  NamedCircle c[5];  cout << "5 개의 정수 반지름과 원의 이름을 입력하세요" << endl;  for(int i=0; i<5; i++) {  int r;  string name;  cout << i+1 << ">> ";  cin >> r;  getline(cin, name);  if(r <= 0) {  cout << "다시 입력하세요" << endl;  i--;  continue;  }  c[i].set(r, name);  }  cout << "가장 면적이 큰 피자는 " << biggest(c, 5) << "입니다" << endl;  } |
| **[실행 결과]** |

1. 교재 419p : 문제 3번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Point {  int x, y;  public:  Point(int x, int y) {  this->x = x; this->y = y;  }  int getX() { return x; }  int getY() { return y; }  protected:  void move(int x, int y) { this->x = x; this->y = y; }  };  class ColorPoint : public Point {  string color;  public:  ColorPoint(int x, int y, string color)  : Point(x, y) {  this->color = color;  }  void setPoint(int x, int y) {  move(x, y);  }  void setColor(string color) {  this->color = color;  }  void show() {  cout << color << "색으로 " << '(' << getX() << ',' << getY() << ')' << "에 위치한 점입니다." << endl;  }  };  int main() {  ColorPoint cp(5, 5, "RED");  cp.setPoint(10, 20);  cp.setColor("BLUE");  cp.show();  } |
| **[실행 결과]** |

1. 교재 419p : 문제 4번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Point {  int x, y;  public:  Point(int x, int y) {  this->x = x; this->y = y;  }  int getX() { return x; }  int getY() { return y; }  protected:  void move(int x, int y) { this->x = x; this->y = y; }  };  class ColorPoint : public Point {  string color;  public:  ColorPoint() : Point(0,0) {  this->color = "BLACK";  }  ColorPoint(int x, int y) : Point(x, y) {  this->color = "BLACK";  }  ColorPoint(int x, int y, string color) : Point(x, y) {  this->color = color;  }  void setPoint(int x, int y) {  move(x, y);  }  void setColor(string color) {  this->color = color;  }  void show() {  cout << color << "색으로 " << '(' << getX() << ',' << getY() << ')' << "에 위치한 점입니다." << endl;  }  };  int main() {  ColorPoint zeroPoint; // BLACK에 (0, 0) 위치의 점  zeroPoint.show(); // zeroPoint를 출력한다.  ColorPoint cp(5, 5);  cp.setPoint(10, 20);  cp.setColor("BLUE");  cp.show(); // cp를 출력한다.  } |
| **[실행 결과]** |

1. 교재 420p : 문제 6번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  class BaseArray {  private:  int capacity;  int \*mem;  protected:  BaseArray(int capacity=100) {  this->capacity = capacity; mem = new int [capacity];  }  ~BaseArray() { delete [] mem; }  void put(int index, int val) { mem[index] = val; }  int get(int index) { return mem[index]; }  int getCapacity() { return capacity; }  };  class MyStack : public BaseArray {  int tos; // 다음에 push 할 위치. 스택에 들어있는 개수와 동일  public:  MyStack(int capacity);  void push(int n);  int pop();  int length() { return tos; }  int capacity() { return getCapacity(); }  };  MyStack::MyStack(int capacity) : BaseArray(capacity) {  tos = 0; // tos는 현재 스택에 들어 있는 개수로 다루어져도 된다.  }  void MyStack::push(int n) {  if(tos == capacity())  return; // stack full error  put(tos, n);  tos++;  }  int MyStack::pop() {  if(tos == 0)  return -1; // stack empty error  tos--;  return get(tos);  }  int main() {  MyStack mStack(100);  int n;  cout << "스택에 삽입할 5개의 정수를 입력하라>> ";  for(int i=0; i<5; i++) {  cin >> n;  mStack.push(n); // 스택에 푸시  }  cout << "스택 용량:" << mStack.capacity() << ", 스택 크기:" << mStack.length() << endl;  cout << "스택의 모든 원소를 팝하여 출력한다>> ";  while(mStack.length() != 0) {  cout << mStack.pop() << ' '; // 스택에서 팝  }  cout << endl << "스택의 현재 크기 : " << mStack.length() << endl;  } |
| **[실행 결과]** |

1. 교재 422p : 문제 8번

|  |
| --- |
| **[프로그램 소스]**  **==== BasePrinter.h ======**  #ifndef BASEPRINTER\_H  #define BASEPRINTER\_H  #include <string>  using namespace std;  class BasePrinter {  string model;  string manufacturer;  int printedCount;  int availableCount;  public:  BasePrinter(string model, string manufacturer, int totalPapers);  protected:  bool print(int pages=1);  string getModel() { return model; }  string getManufacturer() { return manufacturer; }  int getPrintedCount() { return printedCount; }  int getAvailableCount() { return availableCount; }  void show();  };  #endif  **==== InkJetPrinter.h ======**  #ifndef INKJETPRINTER\_H  #define INKJETPRINTER\_H  #include <string>  #include "BasePrinter.h"  using namespace std;  class InkJetPrinter : public BasePrinter {  int availableInk;  public:  InkJetPrinter(string model, string manufacturer, int totalPapers, int totalInk);  bool printInkJet(int pages=1);  void showState();  };  #endif  **==== LaserPrinter.h ======**  #ifndef LASERPRINTER\_H  #define LASERPRINTER\_H  #include <string>  #include "BasePrinter.h"  using namespace std;  class LaserPrinter : public BasePrinter {  int availableToner;  public:  LaserPrinter(string model, string manufacturer, int totalPapers, int totalToner);  bool printLaser(int pages=1);  void showState();  };  #endif  **==== PrinterManager.h ======**  #ifndef PRINTERMANAGER\_H  #define PRINTERMANAGER\_H  #include "InkJetPrinter.h"  #include "LaserPrinter.h"  class PrinterManager {  InkJetPrinter \*ip;  LaserPrinter \*lp;  public:  PrinterManager();  ~PrinterManager();  void operate();  };  #endif  **==== BasePrinter.cpp ======**  #include <iostream>  #include "BasePrinter.h"  using namespace std;  BasePrinter::BasePrinter(string model, string manufacturer, int totalPapers) {  this->model = model;  this->manufacturer = manufacturer;  this->printedCount = 0;  this->availableCount = totalPapers;  }  bool BasePrinter::print(int pages) {  if(availableCount < pages) {  cout << "용지가 부족하여 프린트할 수 없습니다." << endl;  return false;  }  printedCount+=pages;  availableCount-=pages;  return true;  }  void BasePrinter::show() {  cout << model+" ," << manufacturer << " ,남은 종이 " << availableCount << "장" << " ,";  }  **==== InkJetPrinter.cpp ======**  #include <iostream>  #include "BasePrinter.h"  #include "InkJetPrinter.h"  using namespace std;  InkJetPrinter::InkJetPrinter(string model, string manufacturer, int totalPapers, int totalInk)  : BasePrinter(model, manufacturer, totalPapers)  {  this->availableInk = totalInk;  }  bool InkJetPrinter::printInkJet(int pages) {  if(!print(pages)) // BasePrinter::print() 호출  return false; // 용지 부족  if(availableInk < pages) {  cout << "잉크가 부족하여 프린트할 수 없습니다." << endl;  return false;  }  availableInk-=pages;  cout << "프린트하였습니다. " << endl;  return true;  }  void InkJetPrinter::showState() {  show(); // BasePrinter::show() 호출  cout << "남은 잉크 " << availableInk;  }  **==== LaserPrinter.cpp ======**  #include <iostream>  #include "BasePrinter.h"  #include "LaserPrinter.h"  using namespace std;  LaserPrinter::LaserPrinter(string model, string manufacturer, int totalPapers, int totalToner)  : BasePrinter(model, manufacturer, totalPapers)  {  this->availableToner = totalToner;  }  bool LaserPrinter::printLaser(int pages) {  if(!print(pages)) // BasePrinter::print() 호출  return false; // 용지 부족  if(availableToner < pages) {  cout << "토너가 부족하여 프린트할 수 없습니다." << endl;  return false;  }  availableToner--;  cout << "프린트하였습니다. " << endl;  return true;  }  void LaserPrinter::showState() {  show(); // BasePrinter::show() 호출  cout << "남은토너 " << availableToner;  }  **==== PrinterManager.cpp ======**  #include <iostream>  #include "PrinterManager.h"  using namespace std;  PrinterManager::PrinterManager() {  ip = new InkJetPrinter("Officejet V40", "HP", 5, 10);  lp = new LaserPrinter("SCX-6x45", "삼성전자", 3, 20);  }  PrinterManager::~PrinterManager() {  delete ip;  delete lp;  }  void PrinterManager::operate() {  cout << "현재 작동중인 2 대의 프린터는 아래와 같다" << endl;  cout << "잉크젯 : ";  ip->showState();  cout << endl;  cout << "레이저 : ";  lp->showState();  cout << endl;  int printer, pages;  char conti;  while(true) {  cout << endl << "프린터(1:잉크젯, 2:레이저)와 매수 입력>>";  cin >> printer >> pages;  switch(printer) {  case 1 : // 잉크젯  ip->printInkJet(pages);  break;  case 2 : // 레이저  lp->printLaser(pages);  break;  default:  cout << "잘못된 입력입니다." << endl;  continue;  }  ip->showState();  cout << endl;  lp->showState();  cout << endl;  cout << "계속 프린트 하시겠습니까(y/n)>>";  cin >> conti;  if(conti == 'y')  continue;  else  break;  }  }  **==== main.cpp ======**  #include "PrinterManager.h"  int main() {  PrinterManager man;  man.operate();  } |
| **[실행 결과]** |