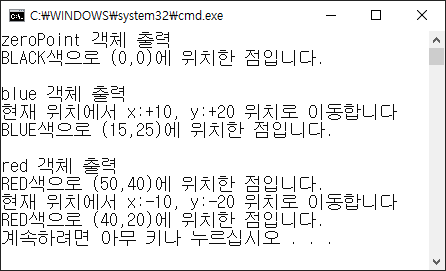
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2020\_2\_C++ \_12 | 학번 : | 20175105 | 이름 : | 곽영주 |

* **프로그램 과제**

1. Point 클래스를 상속받는 ColorPoint 클래스를 작성하고 제시된 결과처럼 실행될 수 있도록 main()을 완성하시오.

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\_up(int x, int y) { this->x += x; this->y += y; }

void move\_down(int x, int y) { this->x -= x; this->y -= y; }

};

int main() {

ColorPoint zeroPoint; // BLACK에 (0, 0) 위치의 점

ColorPoint blue(5, 5);

cout << endl<<"blue 객체 출력" << endl;

cout << "현재 위치에서 x:+10, y:+20 위치로 이동합니다" << endl;

blue.setPoint(10, 20, '+');

ColorPoint red(50, 40, "RED");

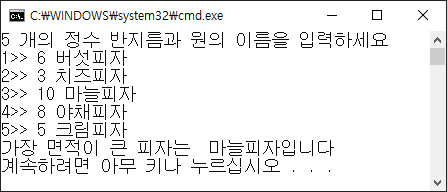
cout << "현재 위치에서 x:-10, y:-20 위치로 이동합니다" << endl;

red.setPoint(10, 20, '-');

}

|  |
| --- |
| **소스코드:**  #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\_up(int x, int y) { this->x += x; this->y += y; }  void move\_down(int x, int y) { this->x -= x; this->y -= y; }  };  class ColorPoint : public Point {  string color;  public:  ColorPoint() : Point(0, 0) { color = "BLACK"; }  ColorPoint(int x, int y, string color = "BLUE") : Point(x, y) { this->color = color; }  void setPoint(int x, int y, char oper);  void show() { cout << color + "색으로 (" << getX() << ", " << getY() << ")에 위치한 점입니다." << endl; }  };  void ColorPoint::setPoint(int x, int y, char oper) {  if (oper == '+')  move\_up(x, y);  else  move\_down(x, y);  show();  }  int main() {  ColorPoint zeroPoint;  cout << "zeorPoint 객체 출력" << endl;  zeroPoint.show();  ColorPoint blue(5, 5);  cout << endl << "blue 객체 출력" << endl;  cout << "현재 위체에서 x:+10, y:+20 위치로 이동합니다." << endl;  blue.setPoint(10, 20, '+');  ColorPoint red(50, 40, "RED");  cout << endl << "red 객체 출력" << endl;  red.show();  cout << "현재 위치에서 x:-10, y:-20 위치로 이동합니다." << endl;  red.setPoint(10, 20, '-');  return 0;  } |
| **실행결과:** |

1. 원을 추상화 한 Circle 클래스에 대하여 다음과 같이 배열을 선언하여 제시된 결과처럼 실행될 수 있도록 Circle을 상속받은 NamedCircle 클래스와 main()함수 등 필요한 함수를 작성하시오

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; };

};

NamedCircle c[5];

|  |
| --- |
| **소스코드:**  #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 = 0, string name = "") : Circle(radius) { this->name = name; }  friend void MaxArea(NamedCircle \*c);  };  void MaxArea(NamedCircle \*c) {  NamedCircle temp = c[0];  for (int i = 1; i < 5; i++) {  if (temp.getArea() < c[i].getArea())  temp = c[i];  }  cout << "가장 면적이 큰 피자는 " << temp.name << "입니다." << endl;  }  int main() {  NamedCircle c[5];  string name;  int radius;  cout << "5개의 정수 반지름과 원의 이름을 입력하세요." << endl;  for (int i = 0; i < 5; i++) {  cout << i + 1 << ">> ";  cin >> radius >> name;  c[i] = NamedCircle(radius, name);  }  MaxArea(c);  return 0;  } |
| **실행결과:** |

1. 다음과 같은 클래스에 대하여 **상속을 적용하여** 제시된 결과처럼 실행될 수 있도록 프로그램을 작성하시오. 단, 잉크젯 프린터와 레이저 프린터 객체를 하나만 동적 생성하여 사용한다.

class InkJetPrinter {

int availableInk; //잉크 잔량

string model; //모델

string manufacturer; //제조사

int printedCount; //총 인쇄 매수

int availableCount; //인쇄 종이 잔량

public:

void printInkJet(int pages = 1); //pages 수만큼 용지 사용, 잉크잔량은 pages 수만큼 감소

void showStateInkJet(); //현재 상태 출력

};

class LaserPrinter : public BasePrinter {

int availableToner; //토너 잔량

string model; //모델

string manufacturer; //제조사

int printedCount; //인쇄매수

int availableCount; //인쇄 종이 잔량

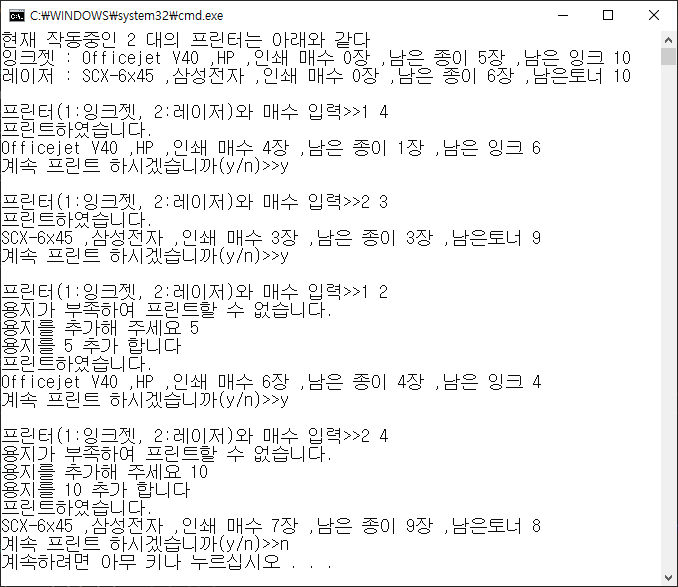
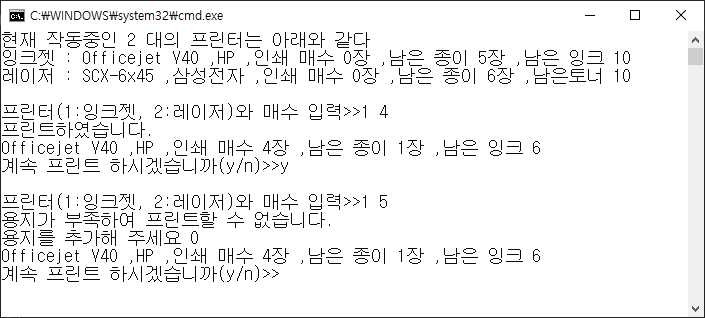
public:

void printLaser(int pages = 1); //pages 수만큼 용지 사용, 토너 잔량은 -1 감소

void showStateLaser(); //현재 상태 출력

};

**- 용지가 부족하면 입력한 값만큼 추가한 후 프린트 하고, 0을 입력하면 현재 프린터 상태만 출력한다**

|  |
| --- |
| **소스코드:**  #include <iostream>  #include <string>  using namespace std;  class BasePrinter {  string model;  string manufacturer;  int printedCount;  int availableCount;  protected:  BasePrinter(string model = "", string manufacturer = "", int printedCount = 0 , int availableCount = 0);  void showState();  int print(int pages = 1);  };  BasePrinter::BasePrinter(string model, string manufacturer, int printedCount, int availableCount) {  this->model = model;  this->manufacturer = manufacturer;  this->printedCount = printedCount;  this->availableCount = availableCount;  }  void BasePrinter::showState() {  cout << model << ", ";  cout << manufacturer << ", ";  cout << "인쇄 매수 " << printedCount << "장, ";  cout << "남은 종이 " << availableCount << "장, ";  }  int BasePrinter::print(int pages) {  while(availableCount < pages){ // 잔량보다 뽑는 수가 많으면  int addPage;  cout << "용지가 부족하여 프린트할 수 없습니다." << endl;  cout << "용지를 추가해 주세요>> ";  cin >> addPage;  if (addPage == 0) // 추가 페이지 = 0일 때  return 0;  cout << "용지를" << addPage << "장 추가 합니다." << endl;  availableCount += addPage;  }  availableCount -= pages;  printedCount += pages;  cout << "프린트하였습니다." << endl;  return 1;  }  class InkJetPrinter : virtual protected BasePrinter {  int availableInk;  public:  InkJetPrinter(string model, string manufacturer, int printedCount, int availableCount, int availableInk) : BasePrinter(model, manufacturer, printedCount, availableCount) { this->availableInk = availableInk; }  void printInkJet(int pages = 1);  void showStateInkJet();  };  void InkJetPrinter::printInkJet(int pages) {  if (availableInk < pages) { // 잉크 부족할 때  cout << "잉크 부족" << endl;  return;  }  if (print(pages) == 0) // 추가 페이지 없을 때  return;  else  availableInk -= pages;  }  void InkJetPrinter::showStateInkJet() {  showState();  cout << "남은 잉크 " << availableInk << endl;  }  class LaserPrinter : virtual protected BasePrinter {  int availableToner;  public:  LaserPrinter(string model, string manufacturer, int printedCount, int availableCount, int availableToner) : BasePrinter(model, manufacturer, printedCount, availableCount) { this->availableToner = availableToner; }  void printLaser(int pages = 1);  void showStateLaser();  };  void LaserPrinter::printLaser(int pages) {  if (availableToner == 0) { // 토너 없을 때  cout<< "토너 부족" << endl;  return;  }  if (print(pages) == 0) // 추가 페이지 없을 때  return;  else  --availableToner;  }  void LaserPrinter::showStateLaser() {  showState();  cout << "남은 토너 " << availableToner << endl;  }  int main() {  InkJetPrinter \*ink = new InkJetPrinter("C-2017", "SAMSUNG", 0, 7, 10);  LaserPrinter \*laser = new LaserPrinter("C-5105", "LG", 0, 8, 10);  int printer, page;  char check;  cout << "현재 작동중인 2대의 프린터는 아래와 같다." << endl;  cout << "잉크젯 : "; ink->showStateInkJet();  cout << "레이저 : "; laser->showStateLaser();  while (true) {  cout << endl << "프린터(1:잉크젯, 2:레이저)와 매수 입력>> ";  cin >> printer >> page;  switch (printer) {  case 1:  ink->printInkJet(page);  ink->showStateInkJet();  break;  case 2:  laser->printLaser(page);  laser->showStateLaser();  break;  default:  cout << "찾는 프린터가 없습니다." << endl;  break;  }  cout << "계속 프린트 하시겠습니까? (y/n)>> ";  cin >> check;  if (check == 'n')  break;  }  delete ink;  delete laser;  return 0;  } |
| **실행결과:** |