|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2018\_2\_C++ \_11 | 학번 : | 20157135 | 이름 : | 이상우 |

1. 가상 함수와 오버라이딩

|  |
| --- |
| **======= Circle.h =====**  #ifndef CIRCLE  #define CIRCLE  class Circle : public Shape {  protected:  virtual void draw();  };  #endif  **======= Line.h ====**  #ifndef LINE  #define LINE  class Line : public Shape {  protected:  virtual void draw();  };  #endif  **======== Rect.h =====**  #ifndef RECT  #define RECT  class Rect : public Shape {  protected:  virtual void draw();  };  #endif  **=======Shape.h ======**  #ifndef SHAPE  #define SHAPE  class Shape {  Shape\* next;  protected:  virtual void draw();  public:  Shape() { next = NULL;}  virtual ~Shape() { }  void paint();  Shape\* add(Shape\* p);  Shape\* getNext() { return next;}  };  #endif  **======== Circle.cpp =====**  #include <iostream>  #include "Shape.h"  #include "Circle.h"  using namespace std;  void Circle::draw() {  cout << "Circle" << endl;  }  **===== Line.cpp ====**  #include <iostream>  #include "Shape.h"  #include "Line.h"  using namespace std;  void Line::draw() {  cout << "Line" << endl;  }  =====Rect.cpp ====  #include <iostream>  #include "Shape.h"  #include "Rect.h"  using namespace std;  void Rect::draw() {  cout << "Rectangle" << endl;  }  **=== Shape.cpp ===**  #include <iostream>  #include "Shape.h"  using namespace std;  void Shape::paint() {  draw();  }  void Shape::draw() {  cout << "Shape" << endl;  }  Shape\* Shape::add(Shape \*p) {  this->next = p;  return p;  }  **==== main.cpp ====**  #include <iostream>  #include "Shape.h"  #include "Circle.h"  #include "Rect.h"  #include "Line.h"  using namespace std;  int main() {  Shape \*pStart=NULL;  Shape \*pLast;  pStart = new Circle(); // 처음에 원 도형을 생성한다.  pLast = pStart;  pLast = pLast->add(new Rect()); // 사각형 객체 생성  pLast = pLast->add(new Circle()); // 원 객체 생성  pLast = pLast->add(new Line()); // 선 객체 생성  pLast = pLast->add(new Rect()); // 사각형 객체 생성  // 현재 연결된 모든 도형을 화면에 그린다.  Shape\* p = pStart;  while(p != NULL) {  p->paint();  p = p->getNext();  }  // 현재 연결된 모든 도형을 삭제한다.  p = pStart;  while(p != NULL) {  Shape\* q = p->getNext(); // 다음 도형 주소 기억  delete p; // 기본 클래스의 가상 소멸자 호출  p = q; // 다음 도형 주소를 p에 저장  }  } |
| **[실행 결과]** |

* 추상 클래스와 순수 가상 함수

|  |
| --- |
| #include <iostream>  using namespace std;  class Calculator { //추상 클래스  public:  virtual int add(int a, int b) = 0; // 두 정수의 합 리턴, 순수 가상 함수  virtual int subtract(int a, int b) = 0; // 두 정수의 차 리턴  virtual double average(int a [], int size) = 0; // 배열 a의 평균 리턴. size는 배열의 크기  };  class GoodCalc : public Calculator { //추상 클래스 구현  public:  int add(int a, int b) { return a + b; }  int subtract(int a, int b) { return a - b; }  double average(int a [], int size) {  double sum = 0;  for(int i=0; i<size; i++)  sum += a[i];  return sum/size;  }  };  int main() {  int a[] = {1,2,3,4,5};  Calculator \*p = new GoodCalc();  cout << p->add(2, 3) << endl;  cout << p->subtract(2, 3) << endl;  cout << p->average(a, 5) << endl;  delete p;  } |
| **[실행 결과]** |

* 내용 점검

1. 교재 9장 연습문제

|  |  |
| --- | --- |
| 번호 | 문제풀이 |
| 1 | 1. Virtual |
| 2 | 1. Overriding |
| 3 | 1. Comlile-time-binding |
| 4 | 다형성 |
| 5 | 1. 기본클래스 Base 파생클래스 Derived 2. Derived::f() called , 3. Base::f() called 4. Base::f() called 5. Base::f() called |
| 6 | 1)1=A의 f(), 2=B의 f(), 3=C의 f()  2) C::f( ) called  3)C::f( ) called  4)C::f( ) called  5)C::f( ) called |
| 7 | 가까운, 범위규칙, 범위 지정 연산자 |
| 8 | 1. ::f() 2. A::f() 3. f() |
| 9 | 2. 소멸자를 가상함수로 선언하는 것이 바람직하다. |
| 10 | id=10  Vituar ~Person(){  cout << “id” << id << endl;  } |
| 11 | 3.virtual void draw()=0; |
| 12 | 3.외부함수도 순수 가상함수로 선언 가능하다. |
| 13 | 2.class Shape{ virtual void draw()=0;}; |
| 14 | 1. 1,3   void draw{  cout<< “반지름” << radius << endl;   1. } |
| 15 | 1. 순수 가상함수 호출 |

* 프로그램 과제

1. 교재 466p : 문제 4번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class LoopAdder { // 추상 클래스  string name; // 루프의 이름  int x, y, sum; // x에서 y까지의 합은 sum  void read(); // x, y 값을 읽어 들이는 함수  void write(); // sum을 출력하는 함수  protected:  LoopAdder(string name = "") { // 루프의 이름을 받는다. 초기값은 ""  this->name = name;  }  int getX() { return x; }  int getY() { return y; }  virtual int calculate() = 0; // 순수 가상 함수, 루프를 돌며 합을 구하는 함수  public:  void run();// 연산을 진행하는 함수  };  void LoopAdder::read() {// x, y입력  cout << name << ":" << endl;  cout << "처음 수에서 두번째 수까지 더합니다. 두 수를 입력하세요 >> ";  cin >> x >> y;  }  void LoopAdder::write() { // 결과 sum 출력  cout << x << "에서 " << y << "까지의 합 = " << sum << " 입니다" << endl;  }  void LoopAdder::run() {  read();// x, y를 읽는다.  sum = calculate();// 루프를 돌면서 계산한다.  write();// 결과 sum을 출력한다,  }  class WhileLoopAdder : public LoopAdder {  public:  WhileLoopAdder(string name) : LoopAdder(name) { ; }  int calculate() {  int sum = 0;  int x = getX();  while (x <= getY()) {  sum += x;  x++;  }  return sum;  }  };  class DoWhileLoopAdder : public LoopAdder {  public:  DoWhileLoopAdder(string name) : LoopAdder(name) { ; }  int calculate() {  int sum = 0;  int x = getX();  do {  sum += x;  x++;  } while (x <= getY());  return sum;  }  };  int main() {  WhileLoopAdder whileLoop("While Loop");  DoWhileLoopAdder doWhileLoop("Do while Loop");  whileLoop.run();  doWhileLoop.run();  char ch;  ch = getchar();  ch = getchar();  } |
| **[실행 결과]** |

1. 교재 467p : 문제 5번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class AbstractGate {  protected:  bool x, y;  public:  void set(bool x, bool y) {  this->x = x;  this->y = y;  }  virtual bool operation() = 0;  };  class ANDGate : public AbstractGate {  public:  bool operation() {  if (x == true && y == true)  return true;  else  return false;  }  };  class ORGate : public AbstractGate {  public:  bool operation() {  if (x == true || y == true)  return true;  else  return false;  }  };  class XORGate : public AbstractGate {  public:  bool operation() {  if (x == true != y == true)  return true;  else  return false;  }  };  int main() {  ANDGate andGate;  ORGate orGate;  XORGate xorGate;  andGate.set(true, false);  orGate.set(true, false);  xorGate.set(true, false);  cout.setf(ios::boolalpha);  cout << andGate.operation() << endl;  cout << orGate.operation() << endl;  cout << xorGate.operation() << endl;  char ch;  ch = getchar();  ch = getchar();  } |
| **[실행 결과]** |

1. 교재 469p : 문제 7번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Shape {  protected:  string name;  int width, height;  public:  Shape(string n = "", int w = 0, int h = 0) { name = n; width = w; height = h; }  virtual double getArea() { return 0; }  string getName() { return name; }  };  class Oval : public Shape {  public:  Oval(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height\*3.14; }  };  class Rect : public Shape {  public:  Rect(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height; }  };  class Triangular : public Shape {  public:  Triangular(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height / 2; }  };  int main() {  Shape \*p[3];  p[0] = new Oval("빈대떡", 10, 20);  p[1] = new Rect("찰떡", 30, 40);  p[2] = new Triangular("토스트", 30, 40);  for (int i = 0; i < 3; i++)  cout << p[i]->getName() << " 넓이는 " << p[i]->getArea() << endl;  for (int i = 0; i < 3; i++) delete p[i];  char ch;  ch = getchar();  ch = getchar();  } |
| **[실행 결과]** |

1. 교재 469p : 문제 8번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class Shape {  protected:  string name;  int width, height;  public:  Shape(string n = "", int w = 0, int h = 0) {  name = n; width = w; height = h;  }  virtual double getArea() = 0; // 순수 가상 함수  string getName() {  return name;  }  };  class Oval : public Shape {  public:  Oval(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height\*3.14; }  };  class Rect : public Shape {  public:  Rect(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height; }  };  class Triangular : public Shape {  public:  Triangular(string n, int w, int h) : Shape(n, w, h) { ; }  double getArea() { return width \* height / 2; }  };  int main() {  Shape \*p[3];  p[0] = new Oval("빈대떡", 10, 20);  p[1] = new Rect("찰떡", 30, 40);  p[2] = new Triangular("토스트", 30, 40);  for (int i = 0; i < 3; i++)  cout << p[i]->getName() << " 넓이는 " << p[i]->getArea() << endl;  for (int i = 0; i < 3; i++) delete p[i];  char ch;  ch = getchar();  ch = getchar();  } |
| **[실행 결과]** |

1. 교재 471p : 문제 9번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <string>  using namespace std;  class printer { // printer 클래스  protected:  string model; // 모델명  string manufacturer; // 제조사  int printedCount; // 인쇄 매수  int availableCount; // 인쇄 종이 잔량  public:  // print 함수와 show 함수를 가상 함수로 구현  printer(string m, string ma, int avail) { model = m; manufacturer = ma; availableCount = avail; }  virtual void print(int pages) = 0; // print 순수 가상 함수, page만큼 프린트  virtual void show() = 0;// show 순수 가상 함수, 현재 모델, 제조사, 인쇄 매수, 인쇄 종이 잔량 등을 출력  int getAvail() { return availableCount; }  };  class InkJetPrinter : public printer {  int availableInk; // 추가적 잉크 잔량 정보  public:  InkJetPrinter(string m, string ma, int avail, int availInk) : printer(m, ma, avail) {  availableInk = availInk;  }  void print(int pages) {  printedCount = pages;  availableCount -= printedCount;  availableInk -= printedCount;  }  void show() {  cout << model << " ," << manufacturer << " ,남은 종이 " << availableCount << "장 ,남은 잉크 " << availableInk << endl;  }  };  class LaserPrinter : public printer {  int availableToner; // 추가적 토너 잔량 정보  public:  LaserPrinter(string m, string ma, int avail, int availToner) : printer(m, ma, avail) {  availableToner = availToner;  }  void print(int pages) {  printedCount = pages;  availableCount -= printedCount;  availableToner -= printedCount;  }  void show() {  cout << model << " ," << manufacturer << " ,남은 종이 " << availableCount << "장 ,남은토너 " << availableToner << endl;  }  };  int main() {  InkJetPrinter\* Ink = new InkJetPrinter("Officejet V40", "HP", 5, 10);  LaserPrinter\* Laser = new LaserPrinter("SCX-6x45", "삼성전자", 3, 20);  int a, b;  string exit;  cout << "현재 작동중인 2 대의 프린터는 아래와 같다" << endl;  cout << "잉크젯 : "; Ink->show();  cout << "레이저 : "; Laser->show();  while (1) {  cout << endl << "프린터(1:잉크젯, 2:레이저)와 매수 입력>>";  cin >> a >> b;  if (a == 1) {  if (b > Ink->getAvail()) {  cout << "용지가 부족하여 프린트할 수 없습니다." << endl;  Ink->show();  Laser->show();  }  else {  cout << "프린트하였습니다." << endl;  Ink->print(b);  Ink->show();  Laser->show();  }  }  else if (a == 2) {  if (b > Laser->getAvail()) {  cout << "용지가 부족하여 프린트할 수 없습니다." << endl;  Ink->show();  Laser->show();  }  else {  cout << "프린트하였습니다." << endl;  Laser->print(b);  Ink->show();  Laser->show();  }  }  cout << "계속 프린트 하시겠습니까(y/n)>>";  cin >> exit;  if (exit == "n") break;  char ch;  ch = getchar();  ch = getchar();  }  } |
| **[실행 결과]** |