REPORT #6

|  |
| --- |
| **제 출 일** : 2017년 11월 28일 |
| **과 목 명** : 컴퓨터 프로그래밍 및 실습 |
| **담당교수 : 김정국 교수님** |
| **학 과** : 컴퓨터 전자 시스템 공학부 |
| **학 번** : 201702234 |
| **성 명 : 유동혁** |

**HANKUK UNIVERSITY OF FOREIGN STUDIES**

****

**1. 가상 함수의 예제 1**

|  |
| --- |
| /\*  \* gasang1.cpp  \*  \* Created on: 2017. 11. 27.  \* Author: Yoo DongHyuk  \*/  #include <iostream>  using namespace std;  class figure  {  protected://하위 함수에 상속되는 변수  int x, y;  public:  virtual void draw();//가상함수 - 하위 함수에서 동일한 함수명을 사용하기 위함  void setOrigin(int x, int y);  };  void figure::draw()//figure의 draw  {  cout << "figure draw" << endl;  }  void figure::setOrigin(int x, int y)  {  this -> x = x;  this -> y = y;  }  class rectangle : public figure//figure의 하위 클래스 - rectangle  {  private:  int width, height;  public:  void setWidth(int w);  void setHeight(int h);  void draw();//rectangle의 draw  };  void rectangle::setWidth(int w)  {  width = w;  }  void rectangle::setHeight(int h)  {  height = h;  }  void rectangle::draw()//rectangle 의 draw  {  cout << "rectangle draw" << endl;  }  class circle : public figure//figure의 하위 클래스 - circle  {  private:  int radius;  public:  void setRadius(int r);  void draw();//circle의 draw  };  void circle::setRadius(int r)  {  radius = r;  }  void circle::draw()//circle의 draw  {  cout << "circle draw" << endl;  }  int main()  {  figure \*ps = new rectangle();  ps -> draw();//rectangle draw  delete ps;  figure \*ps2 = new circle();  ps2 -> draw();//circle draw  delete ps2;  return 0;  } |

**1.4 결과 화면**

|  |
| --- |
|  |

**2. 가상 함수를 사용하지 않았을 때의 예제**

|  |
| --- |
| /\*  \* gasangX.cpp  \*  \* Created on: 2017. 11. 27.  \* Author: Yoo DongHyuk  \*/  #include <iostream>  using namespace std;  class figure  {  protected:  int x, y;  public:  void draw();//virtual void draw(); ==> 가상함수를 사용하지 않음  void setOrigin(int x, int y);  };  void figure::draw()//figure의 draw  {  cout << "figure draw" << endl;  }  void figure::setOrigin(int x, int y)  {  this -> x = x;  this -> y = y;  }  class rectangle : public figure//figure의 하위 클래스 - rectangle  {  private:  int width, height;  public:  void setWidth(int w);  void setHeight(int h);  void draw();//rectangle의 draw  };  void rectangle::setWidth(int w)  {  width = w;  }  void rectangle::setHeight(int h)  {  height = h;  }  void rectangle::draw()//rectangle의 draw  {  cout << "rectangle draw" << endl;  }  class circle : public figure//figure의 하위 클래스 - circle  {  private:  int radius;  public:  void setRadius(int r);  void draw();//circle의 draw  };  void circle::setRadius(int r)  {  radius = r;  }  void circle::draw()//circle의 draw  {  cout << "circle draw" << endl;  }  int main()  {  figure \*ps = new rectangle();  ps -> draw();//rectangle draw(x) figure draw(o)  delete ps;  figure \*ps2 = new circle();  ps2 -> draw();//circle draw(x) figure draw(o)  delete ps2;  return 0;  } |

**2.4 결과 화면**

|  |
| --- |
|  |

**3. 참조자와 가상 함수의 예제**

|  |
| --- |
| /\*  \* gasang2.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class animal//최상위 클래스 - animal  {  public:  virtual void speak();//가상함수 - speak  };  void animal::speak()//animal의 speak  {  cout << "animal speak()" << endl;  }  class dog : public animal//animal의 하위 클래스 - dog  {  public:  void speak();//dog의 speak  };  void dog::speak()//dog의 speak  {  cout << "dog speak" << endl;  }  class cat : public animal//animal의 하위 클래스 - cat  {  public:  void speak();//cat의 speak  };  void cat::speak()//cat의 speak  {  cout << "cat speak" << endl;  }  int main()  {  dog d;//dog 클래스의 d  animal &a1 = d;//animal클래스의 멤버를 참조  a1.speak();//a1.speak == d.speak  cat c;//cat클래스의 c  animal &a2 = c;//animal클래스의 멤버를 참조  a2.speak();//a2.speak == c.speak  return 0;  } |

**3.4 결과 화면**

|  |
| --- |
|  |

**4. 가상 소멸자 예제**

|  |
| --- |
| /\*  \* gasangsomyeol.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class animal  {  public:  animal();  virtual ~animal();//가상 생성자를 소멸 시키기 위한 가상 소멸자  virtual void speak();//가상 생성자  };  animal::animal()//animal생성자  {  cout << "animal sang sung" << endl;  }  animal::~animal()//animal소멸자  {  cout << "animal so myeol" << endl;  }  void animal::speak()//animal의 speak  {  cout << "animal speak()" << endl;  }  class dog : public animal//animal의 하위 클래스 - dog  {  public:  dog();//dog생성자  ~dog();//dog소멸자  void speak();//dog의 speak  };  dog::dog()//dog 생성자  {  cout << "dog sang sung" << endl;  }  dog::~dog()//dog 소멸자  {  cout << "dog so myeol" << endl;  }  void dog::speak()//dog의 speak  {  cout << "dog speak" << endl;  }  int main()  {  animal \*a1 = new dog();//dog 동적할당  a1->speak();//dog speak  delete a1;  return 0;  } |

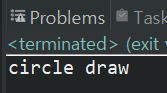
**4.4 결과 화면**

|  |
| --- |
|  |

**5. 순수 가상 함수 예제**

|  |
| --- |
| /\*  \* sunsoo.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class figure  {  protected:  int x, y;  public:  void setOrigin(int x, int y);  virtual void draw() = 0;//순수 가상 함수 - 함수의 형식/함수명만 상속시킴  };  void figure::setOrigin(int x, int y)  {  this -> x = x;  this -> y = y;  }  class circle : public figure//figure의 하위 클래스인 circle  {  private:  int radius;  public:  void setRadius(int r);  void draw();//circle의 draw  };  void circle::setRadius(int r)  {  radius = r;  }  void circle::draw()  {  cout << "circle draw" << endl;  }  int main()  {  figure \*ps = new circle();  ps -> draw();  delete ps;  return 0;  } |

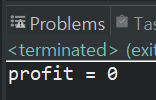
**결과화면**



**6. 프렌드 함수 예제 1**

|  |
| --- |
| /\*  \* friend.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class company  {  private:  int sales, profit;  friend void sub(company& c);//프렌드 함수 선언  public:  company();  };  company::company()  {  sales = 0;  profit = 0;  }  void sub(company& c)//company&를 넣음으로써, company의 멤버인 c를 사용할 수 있다.  {  cout << "profit = " << c.profit << endl;  }  int main()  {  company c1;//c1생성  sub(c1);//sub실행 - c1대입  return 0;  } |

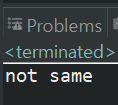
**결과 화면**



**7. 프렌드 함수 예제 2**

|  |
| --- |
| /\*  \* friend2.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class date  {  friend bool equals(date d1, date d2);//프렌드함수 선언 - date의 멤버를 사용하기 위함  private:  int year, month, day;  public:  date(int y, int m, int d);  };  date::date(int y, int m, int d)  {  year = y;  month = m;  day = d;  }  bool equals (date d1, date d2)  {  return d1.year == d2.year && d1.month == d2.month && d1.day == d2.day;//이 문장이 참이면 1을 반환  }  int main()  {  date d1(1975, 5, 22), d2(2002, 8, 12);  if(equals(d1, d2) == 1)  cout << "same" << endl;  else  cout << "not same" << endl;  return 0;  } |

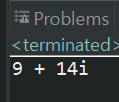
**결과 화면**



**8. 프렌드 함수 예제 3**

|  |
| --- |
| /\*  \* friend3.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class complex  {  private:  int re, im;  public:  friend complex add(complex a1, complex a2);//프렌드 함수 선언 - complex의 멤버를 사용하기 위함  complex();  complex(double r);  complex(double r, double i);  void output();  };  complex::complex()  {  re = 0;  im = 0;  }  complex::complex(double r)  {  re = r;  im = 0;  }  complex::complex(double r, double i)  {  re = r;  im = i;  }  void complex::output()  {  cout << re << " + " << im << "i" << endl;  }  complex add(complex a1, complex a2)  {  return complex(a1.re + a2.re, a1.im + a2.im);  }  int main()  { complex c1(4, 6), c2(5, 8);  complex c3 = add(c1, c2);  c3.output();  return 0;  } |

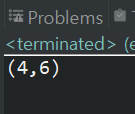
**결과 화면**



**9. 연산자 중복 예제 1**

|  |
| --- |
| /\*  \* yeon.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class vector  {  private:  double x, y;  public:  vector(double xvalue = 0.0, double yvalue = 0.0) : x(xvalue), y(yvalue)  {  }  void display()  {  cout << "(" << x << "," << y << ")" << endl;  }  vector operator+(vector& v2)//+연산자 함수 정의  {  vector v;  v.x = this -> x + v2.x;  v.y = this -> y + v2.y;  return v;  }  };  int main()  {  vector v1(1.0, 2.0), v2(3.0, 4.0);  vector v3 = v1 + v2;//operator에서 정의한 +계산방법으로 계산된다.  v3.display();  return 0;  } |

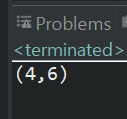
**결과 화면**



**10. 연산자 중복 예제 2**

|  |
| --- |
| /\*  \* yeon2.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class vector  {  friend vector operator+(const vector& v1, const vector& v2);  private:  double x, y;  public:  vector(double xvalue = 0.0, double yvalue = 0.0) : x(xvalue), y(yvalue)  {  }  void display()  {  cout << "(" << x << "," << y << ")" << endl;  }  };  vector operator+(const vector& v1, const vector& v2)//+연산자 함수 정의  {  vector v(0.0, 0.0);  v.x = v1.x + v2.x;  v.y = v1.y + v2.y;  return v;  }  int main()  {  vector v1(1.0, 2.0), v2(3.0, 4.0);  vector v3 = v1 + v2;//operator에서 정의한 대로 +연산이 실행된다.  v3.display();  return 0;  } |

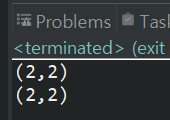
**결과 화면**



**11. \*연산자 중복 예제**

|  |
| --- |
| /\*  \* yeon3.cpp  \*  \* Created on: 2017. 11. 21.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class vector  {  friend vector operator\*(vector& v, double alpha);  friend vector operator\*(double alpha, vector& v);//프렌드/\*연산자 함수 선언  private:  double x, y;  public:  vector(double xvalue = 0.0, double yvalue = 0.0) : x(xvalue), y(yvalue){}  void display()  {  cout << "(" << x << "," << y << ")" << endl;  }  };  vector operator\*(vector& v, double alpha)  {  return vector(alpha\*v.x, alpha\*v.y);  }  vector operator\*(double alpha, vector& v)  {  return vector(alpha\*v.x, alpha\*v.y);  }  //\*연산자 함수 정의  int main()  {  vector v(1.0, 1.0);  vector w = v\*2.0;  vector z = 2.0 \* v;  w.display();  z.display();  return 0;  } |

**결과 화면**



**12. ==연산자 중복 예제**

|  |
| --- |
| /\*  \* yeon4.cpp  \*  \* Created on: 2017. 11. 27.  \* Author: Yoo DongHyuk  \*/  #include<iostream>  using namespace std;  class vector  {  private:  double x, y;  public:  vector(double xvalue = 0.0, double yvalue = 0.0) : x(xvalue), y(yvalue){}  void display()  {  cout << "(" << x << "," << y << ")" << endl;  }  friend bool operator==(const vector& v1, const vector& v2);  friend bool operator!=(const vector& v1, const vector& v2);//==, !=연산자함수 선언  };  bool operator==(const vector& v1, const vector& v2)  {  return v1.x == v2.x && v2.y == v2.y;  }  bool operator!=(const vector& v1, const vector& v2)  {  return !(v1 == v2);  }  int main()  {  vector v1(1, 2), v2(1, 2);  vector v3(5, 6), v4(7, 8);  if(v1 == v2)  cout << "true" << endl;  if(v3 != v4)  cout << "false" << endl;  return 0;  } |

**결과 화면**

