|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cpp\_12주 | 학번 : |  | 이름 : |  |

* 개념 확인 - 상속

1. 다음 질문에 O, X로 답하시오
2. 기본 클래스로 생성된 객체는 파생 클래스 멤버에 접근할 수 있다( )
3. 접근 지정자 protected로 선언된 멤버는 전역 함수와 외부클래스에서 접근할 수 있다( )
4. 상속 접근 지정자가 생략되면 public이다 ( )
5. C++는 다중 상속을 허용한다( )
6. 파생클래스 생성자에서 명시적으로 기본 클래스 생성자를 선택하지 않으면 묵시적으로 기본 생성자를 호출한다. ( )
7. 상속은 클래스를 선언할 때 extends 키워드를 사용한다 ( )
8. 파생 클래스의 포인터가 기본클래스의 포인터에 치환되는 것을 다운캐스팅 이라고 한다 ( )
9. 업 캐스팅은 반드시 강제 타입 변환이 필요하다 ( )
10. 상속을 통하여 파생클래스에서는 기본클래스이 모든 멤버를 사용할 수 있다 ( )
11. private 상속은 기본클래스의 protected, public 멤버를 private으로 계승한다 ( )
12. 가상 상속을 선언할 때 사용하는 키워드는 무엇인가?

==== 풀이 ====

1. 다음 클래스 Rocket과 클래스 Computer를 동시에 상속받아 클래스 Statellite를 선언하시오

class Rocket{ …. }

class Computer{ … }

==== 풀이 ====

1. 다음 코드에 대하여 객체 b가 생성될 때 화면에 출력되는 내용은 무엇인가?

|  |
| --- |
| class A {  public:  A() { cout << "생성자 A" << endl; }  A(int x) { cout << "생성자 A" << x<< endl; }  };  class B: public A {  public:  B() { cout << "생성자 B" << endl;}  B(int x) { cout << "생성자 B" << x << endl; }  B(int x, int y) : A(x+y+2) {cout << "생성자 B" << x + y + 2 << endl; }  }; |

1. B b;
2. B b(10);
3. B b(10, 20);
4. C++ 상속의 종류에 대하여 예를 제시하고 설명하시오.

==== 풀이 ====

1. 다중 상속의 문제점과 해결방법을 예를 제시하고 설명하시오.

==== 풀이 ====

* **개념 적용 응용 프로그래밍 – 연산자 중복**

1. Book 객체에 대하여 다음과 같은 연산을 수행할 수 있도록 연산자 함수를 멤버 함수로 구현하고 프로그램을 완성 하시오

class Book {

string title;

int price, pages;

public:

Book(string title = "", int price = 0, int pages = 0);

void show();

string getTitle() //title 반환

};

int main() {

Book a("청춘", 20000, 300), b("미래", 30000, 500);

a += 500; // 책 a의 가격 500원 증가

b -= 500; // 책 b의 가격 500원 감소

a.show();

b.show();

Book b1("명품 C++", 30000, 500), b2("고품 C++", 30000, 500);

if (b1 == 30000) cout << "정가 30000원" << endl; // price 비교

if (b1 == "명품 C++") cout << "명품 C++ 입니다." << endl; // 책 title 비교

if (b1 == b2) cout << "두 책이 같은 책입니다." << endl; // title, price, pages 모두 비교

}

|  |
| --- |
| [프로그램 소스] |
| [실행 결과] |

1. Matrix 클래스에 대하여 다음과 같은 연산이 가능하도록 연산자 함수를 멤버 함수로 구현하고 프로그램을 완성하시오.

class Matrix{

  int ar[4];

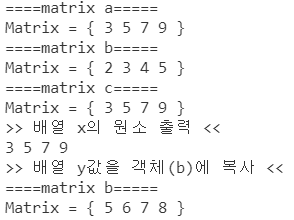
public:

  Matrix(int a1 = 0, int a2 = 0, int b1 = 0, int b2 = 0);

  void show(string matrix);

};

int main(){

  Matrix a(1, 2, 3, 4), b(2, 3, 4, 5), c;

  c = a + b;

  a += b;

  a.show("matrix a");

  b.show("matrix b");

  c.show("matrix c");

  int x[4], y[4] = {5, 6, 7, 8};

  a >> x; // a의 각 원소를 배열 x에 복사. x[]는 {4,3,2,1}

  b << y; // 배열 y의 원소 값을 b의 각 원소에 설정

  cout << ">> 배열 x의 원소 출력 << " << endl;

  for (int i = 0; i < 4; i++)

    cout << x[i] << ' '; // x[] 출력

  cout << endl;

  cout << ">> 배열 y값을 객체(b)에 복사 << " << endl;

  b.show("matrix b");

}

|  |
| --- |
| [프로그램 소스] |
| [실행 결과] |

1. 원을 추상화 한 Circle 클래스에 대하여 다음 연산이 가능하도록 멤버 함수로 작성하고 프로그램을 완성 하시오.

class Circle {

int radius;

public:

Circle(int radius = 0);

void show();

};

int main() {

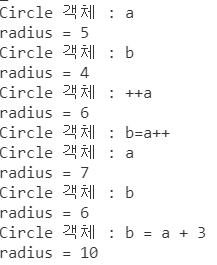
  Circle a(5), b(4);

  cout << "Circle 객체 : a" << endl;

  a.show();

  cout << "Circle 객체 : b" << endl;

  b.show();

  ++a; // 반지름을 1 증가 시킨다.

  cout << "Circle 객체 : ++a" << endl;

  a.show();

  cout << "Circle 객체 : b=a++" << endl;

  b = a++; // 반지름을 1 증가 시킨다.

  cout << "Circle 객체 : a" << endl;

  a.show();

    cout << "Circle 객체 : b" << endl;

  b.show();

  b = a + 3; // b의 반지름을 a의 반지름에 1을 더한 것으로 변경

  cout << "Circle 객체 : b = a + 3" << endl;

  b.show();

}

|  |
| --- |
| [프로그램 소스] |
| [실행 결과] |

1. []연산자를 멤버 함수로 정의하여 제시된 결과처럼 실행하는 프로그램을 작성하시오.

class Array{

private:

double\* ptr;

int size;

public:

Array(int size); //size크기를 갖는 배열 동적 생성

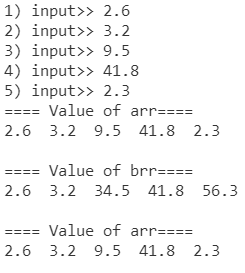
~Array();

void show();

//[]연산자와 = 연산자 중복

};

int main(){

  Array arr(5), brr(5);

  for (int i = 0; i < 5; i++) {

      cout << i+1 <<") input>> ";

    cin >> arr[i];

  }

   cout<<"==== Value of arr===="<<endl;

  arr.show();

  brr = arr;

  brr[2] = 34.5;

  brr[4] = 56.3;

  cout<<"==== Value of brr===="<<endl;

  brr.show();

  cout<<"==== Value of arr===="<<endl;

  arr.show();

  return 0;

}

|  |
| --- |
| [프로그램 소스] |
| [실행 결과] |

1. 정수 배열을 항상 증가 순으로 유지하는 StoredArray 클래스를 작성하려고 한다. 아래의 main()함수가 작동할 수 있도록 SortedArray 클래스를 작성하고 +와 =연산자도 작성하시오.

class SortedArray {

int size; // 현재 배열의 크기

int \*p; // 정수 배열에 대한 포인터

void sort(); // 정수 배열을 오름차순으로 정렬

public:

SortedArray(); // p는 NULL로 size는 0으로 초기화

SortedArray(SortedArray& src); // 복사 생성자

SortedArray(int p[], int size); // 생성자. 정수 배열과 크기를 전달받음

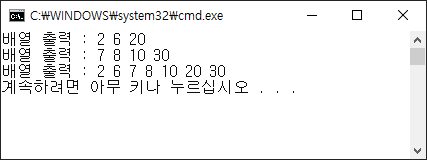
~SortedArray(); // 소멸자

SortedArray operator + (SortedArray& op2);

SortedArray& operator = (const SortedArray& op2); // 현재 배열에 op2 배열을 복사

void show(); // 배열의 원소 출력

};

int main() {

int n[] = { 2, 20, 6 };

int m[] = { 10, 7, 8, 30 };

SortedArray a(n, 3), b(m, 4), c;

c = a + b;

a.show();

b.show();

c.show();

}

|  |
| --- |
| [프로그램 소스] |
| [실행 결과] |