陳銘憲詳程

Computer Programming Final Examination

1/10/2001

1. Explain briefly

(a.) operator overloading; (b.) pure virtual function (c.) early binding (d.) friend class (e.) static variable

2~10. What is the output of code 2 ~ code 10?

CODE 4.

```
#include <iostream.h>
#include <string.h>
class Item{
  char item_no[5];
  double *price;
public:
  Item(char *no, double &pr){
    strcpy(item_no,no); price = &pr;
  }
  void operator=(Item &it){
    strcpy(item_no,it.item_no);
    price = it.price;
  }
  void display(){
    cout << "Item_No: " << item_no << endl;
    cout << "Price : " << *price << endl;
  }
  void update(char* no, double p){
    strcpy(item_no, no);*price = p+1;
  }
};

class Order{
  Item * item;
  int quantity;
public:
  Order(char* no, double &pr, int q){
    item = new Item(no,pr); quantity = q;
  }
  void update_item(char * no, double p)
  {item->update(no,p);}
  void display(){
    item->display();
    cout << "Qyt: " << quantity << endl;
  }
  Order &operator=(Order &o){
    *item = *(o.item); quantity=o.quantity-1;
    return *this;
  }
  ~Order(){delete item;}
};
void main(){
  double a = 26.50;
  Order od1("2005",a,88), od2("1005",a,50),
  od3("3008",a,94);
  od3=od2=od1;
  od1.update_item("3880",95.25);
  od1.display(); od2.display(); od3.display();
}
```

CODE 2

```
#include <iostream.h>
int age=5;
class Tree{
  char   variety[10];
  int age;
  double height;
public:
  void input_age(){age=::age+2;}
  void print_age(){
    cout<<"External age "<<age<<endl;
    cout<<"Internal age "<<::age<<endl;
  }
};
void main(){
  Tree k;
  k.input_age();
  k.print_age();
}
```

CODE 3

```
#include <iostream.h>
class Saving{
  int access;
  double money;
public:
  Saving(double m){money = m;access = 0;}
  double operator!(){
    access++;
    return money-access;
  }
  Saving& operator-(){
    money = -money;
    return *this;
  }
};
void main(){
  Saving s(100), t(50);
  -s; -t;
  cout << !s << endl;
  cout << !(-s) << endl;
  cout << !(-t) << endl;
  cout << !(-(-s)) << endl;
}
```

CODE 5

```
#include <iostream.h>
double salary=2500.5;
double change(double &money){
  money-=100.0;
  salary+=30;
  return money+salary;
}
void main(){
  double x = change(salary);
  x+=100;
  salary+=50;
  cout <<"x = "<<x<<endl;
  cout <<"salary = "<<salary<<endl;
}
```

CODE 7

```
#include <iostream.h>
class CLOCK;
class WATCH{
  int hour, minute, second;
public:
  WATCH(){hour=0;minute=0;second=0;}
  WATCH( int m, int h, int sec )
  {hour=h; minute=m; second=sec; }
  friend void timing( WATCH *pw, CLOCK *pc );
};
class CLOCK{
  int hour, minute;
public:
  CLOCK(){}
  CLOCK( int h, int m ) { hour=h; minute=m; }
  friend void timing( WATCH *pw, CLOCK *pc );
  friend void print( CLOCK *pc );
};

void timing( WATCH *pw, CLOCK *pc ){
  pc->hour = pw->hour;
  pc->minute = pw->minute;
}
void print( CLOCK *pc ){
  cout << "The CLOCK time is "<< pc->hour << ":"
     << pc->minute << '\n';
}
void main(){
  CLOCK c( 10, 12 );
  WATCH w( 11, 30, 20 );
  timing( &w, &c );
  print( &c );
}
```

CODE 6

```
#include <iostream.h>
class Integer{
  static int numb;
  int *pv, *pm;
public:
  Integer(){
    pv = new int;
    pm = new int[4];
    cout << "Constructing" << endl;
    *pv = numb++;
    pm[0]=pm[2]=numb--;
    pm[1]=pm[3]=numb++;
  }
  ~Integer(){
    cout << "value = "<<(*pv)<<endl;
    delete pv;
    cout <<"num = "
     <<(pm[0]+pm[1]-pm[2]+pm[3])<<endl;
  }
};
int Integer::numb = 6;
void main(){
  Integer *py = new Integer[2];
  delete[] py;
  cout << "--------" << endl;
  py = new Integer[2];
  delete [] py;
}
```

CODE 8

```
#include <iostream.h>
#include <string.h>
class CA2{
  friend int & func(int &);
private:
    char month[20];
public:
  CA2(const char* str = " "){
    strncpy(month,str,20);
    cout << "Hello.. "<< month << "\n";
  }
  CA2(const CA2 & ca){
    strncpy(month,ca.month,20);
    cout << 'Hi.. " << month << "\n";
  }
  ~CA2(){cout << "Bye.. " << month << "\n";}
};
```

Continued

CODE 8 (continued)

```
int & func(int &);
int date =1;
CA2 ca1("ca1");
void main(){
  static CA2 ca2("ca2" );
  CA2 *ca_ptr = new CA2("ca_ptr");
  CA2 ca3(ca2);
  cout << "date = " << date << endl;
  date += func(date);
  cout << "date = " << date << endl;
  date -= func(date);
  cout << "date = " << date << endl;
}
int & func(int &val){
  CA2 ca4("ca4");
  static CA2 ca5("ca5" );
  static int count=val;
  int tmp = val;
  count += tmp;
  val += count;
  return ++val;
}
```

CODE 9

```
#include <iostream.h>
class SalesPerson{
protected:
  double total_amount;
public:
  SalesPerson( double a ){total_amount=a;}
  virtual double bonus()=0;
};
class SalesEngineer : public SalesPerson{
public:
  SalesEngineer(double a):SalesPerson(a){}
  double bonus(){
    return( total_amount*0.008+500.0 );
  }
};
class SalesManager : public SalesPerson{
public:
  SalesManager(double a):SalesPerson(a){}
  double bonus(){
    return( total_amount*0.008+2000.0 );
  }
};
void compute_bonus( SalesPerson *ps ){
  cout <<'bonus:"<< ps->bonus() <<endl;
}
void main(){
  SalesManager peter( 20000.0);
  SalesEngineer alvin(80000.0),
                lily(50000.0);
  compute_bonus(&peter);
  compute_bonus( &alvin );
  compute_bonus( &lily );
}
```

CODE 10

```
#include <iostream.h>
class Fairy_tale{
public:
  virtual void act1(){
    cout << "Greeting 21 century\n";
    act2();
  }
  void act2(){
    cout << "Enjoy having been with you\n";
    act3();
  }
  virtual void act3(){
    cout << "You are good students\n";
    act4();
  }
  virtual void act4()=0;
  void act5(){
    cout << "See you in other classes";
    cout << "(not this one again\n";
  }
};
class Unhappy_tale : public Fairy_tale{
public:
  void act3(){
    cout << "I love discrete math\n";
    act4();
  }
  void act4(){
    cout << "Happy Lunar New Year\n";
    act5();
  }
  void act5(){
    cout << "Happy Winter Break\n";
  }
};
class Happy_tale : public Fairy_tale{
public:
  void act4(){
    cout<<"Please live happily ever after\n";
    act5();
  }
};
void main(){
  char ch;
  Fairy_tale *tale;
  tale = new Happy_tale;
  tale -> act1();
  tale = new Unhappy_tale;
  delete tale;
}
```