座位号＿＿＿＿ 专业＿＿＿＿ 学院＿＿＿＿ 学号□□□□□□□□□□□□ 姓名＿＿＿＿

（密封线内不答题）

密 封 线

诚信应考,考试作弊将带来严重后果!

华南理工大学期末考试

《 C++程序设计(II) 》试卷 A

注意事项：1. 考前请将密封线内各项信息填写清楚；
2. 所有答案请直接答在试卷上；
3．考试形式：闭卷；
4. 本试卷共 四 大题，满分 100 分， 考试时间 120 分钟。

题 号	一	二	三	四	五	总分
得 分						
评卷人						

1. Sate whether each of the following is **true** or **false**. (20 scores, each 1 scores)

1) A constant object must be initialized, it can be modified after it is created. ( )

2) A **static** class member represents class-wide information. ( )

3) C++ provides for multiple inheritance, which allows a derived class to inherit from many based classes, even if these base classes are unrelated. ( )

4) Treating a bass-class object as a derived-class object can cause errors. ( )

5) Operator **dynamic-cast** can be used to downcast base-class pointers safely. ( )说实在话没用过...

6) A function template can be overloaded by another function template with the same function name. ( )

7) Input/output in C++ occurs as streams of bytes. ( )

8) When using parameterized manipulators, the header file <iostream> must be included. ( )

9) Member function **read** cannot be used to read data from the input object **cin**. ( )

10) The programmer must create the **cin, cout, cerr** and **clog** objects explicitly. ( )

11) A nonmember function must be declared as a **friend** of a class to have accessed to that class's **protected** data members. ( )

12) A member function can not be declared **static** if it must access **non-static** class member. ( )
静态成员函数可以访问非静态类成员

13) The precedence, associativity and "arity" of an operator can not be changed by overloading the operator. ( )

14) In C++, all existing operators can be overloaded. ( )

15) Base-class constructors are not inherited by derived classes. ( )

16) A “has -a” relationship is implemented via inheritance. ( )

17) Polymorphic programming can eliminate the need for switch logic. ( )

18) A **friend** function of a function template must be a function-template specialization. ( )

19) The **cin** stream normally is connected to the keyboard. ( )

20) An exception thrown outside a **try** block causes a call to **terminate.** [异常处理] ( )

2. Answer the following questions.（29 scores）

1) Fill in the blanks in each of the following program. The program should read the record from the file “C：\boot.ini”, and display it on screen. (8 scores)

```
    #include______________________
    void main()
   {    char buffer[100];
        ifstream input (________________________)；
        while (_________________________)
        { input.getline(________________________，  80)；
            cout<< buffer<<endl;
        }
        input.close();
  }
```

2) Find the errors in the following program and explain how to correct them.（5scores）

```
# include <iostream.h>
# include <stdlib.h>
 class CTest{
    public:
        CTest()
          { x=20;}
        void use_this();
    private:
        int x;
 }
 void CTest::use_this()
 {
      CTest y,*pointer;
      this=&y;
      *this.x=10;
      pointer=this;
```

```
        pointer=&y;
  }
  void main()
  {
        CTest y ;
        this->x=235;
}
```

3) Fill in the blanks in each of the following program.（8 scores）

```
#include <iostream>
using std::cout;
using std::cin;
using std::endl;
template  ________________________
class Stack {
public:
      Stack( int = 10 );
      ~Stack()
        {delete [] stackPtr; }
      bool push( const T& );    // push an element onto the stack
      bool pop( T& );                  // pop an element off the stack
      bool isEmpty() const // determine whether Stack is empty
      bool isFull() const        // determine whether Stack is full
private:
      int size;
      int top;
      T *stackPtr;
}; // end class Stack


______________________________
Stack< T >::Stack( int s ){
      size = s > 0 ? s : 10;
      top = -1;    // Stack initially empty
      stackPtr = new T[ size ]; // allocate memory for elements
}
………..//Omit other member function definition
void main()
  {______________________  doubleStack( 5 );
      double doubleValue = 1.1;
      doubleStack.push( doubleValue )
```

```
    ________________________    intStack;
   int intValue = 1;
   intStack.push(intValue);
} // end main
```

4) Finish the definition and implement of class CTest。（8 scores）

```
#include <iostream.h>
Class CTest{
  private:
        int x,y;
  public:
      CTest(int n1, int n2)
       { x=n1;y=n2}
      ______ CTest& operator++( int = 0); (这个 int = 0  忘了是不是正确用法,懒得查了)
      _________________________
      void print()
           {cout<<"x="<<x<<"y="<<y<<endl;}
}
  CTest CTest:: operator++(int = 0)___________________________
       { CTest temp( *this); x++; y++; return &temp;}
(不知道红字那部分行不行…稳妥点的话就改成  CTest temp; temp.x = x; temp.y = y;万无一失)______
}

void main()
{
      CTest d1(2,3);
      d1.print();
      d1++;
      d1.print();
}
```

(↓不做,你能理解的)

2. For each of the following, show the output（25 scores, each 5 scores）

```
1). #include<iostream.h>
struct list
{     int data ;
      list * next ;
} ;
list * head ;
list * insert ( int num )
```

```
{ list * s, *p, *q ;
    s = new list ;
    s->data = num ;    s->next = NULL ;
    if ( head == NULL )
        { head = s ;      return( head ) ; }
    if ( head->data > s->data )
      { s->next = head ;      head = s ;
          return ( head ) ;
      }
    for ( q = head, p = head->next ;    p ; q = p, p = p->next )
      if ( p->data > s->data )
        { s->next = p ;           q->next = s ;
          return ( head ) ;
        }
    q->next = s ;
    return ( head ) ;
}
void showlist( const list * head )
{ cout << "now the items of list are: \n" ;
    while( head )
    { cout << head->data << '\t';        head = head->next ;    }
    cout << endl ;
}
 void main()
{ int k[5]={2,9,1,6,4} ;
    head = NULL ;
    cin >> k ;
    for (int i=0;i<5;i++0 )
        head = insert(k[i]) ;
    showlist( head ) ;
}
```

2).   #include <iostream.h>

```
class BASE
{    public:
            void get( int i,int j,int k,int l )
          { a = i; b = j; x = k;    y = l;      }
            void print()
```

```
        {cout << "a = "<< a << '\t' << "b = " << b << '\t'<< "x = " << x << '\t' << "y = " << y << endl;      }
         int a,b;
      protected:
         int x, y;
};
class A: public BASE
{ public:
      void get( int i, int j, int k, int l )
         { BASE obj3;
                obj3.get( 50, 60, 70, 80 );
                obj3.print();
                a = i; b = j; x = k; y = l;
                u = a + b + obj3.a ; v = y - x + obj3.b;
         }
          void print()
            { cout << "a = " << a << '\t' << "b = " << b << '\t'<< "x = " << x << '\t' << "y = " << y << endl;
              cout << "u = " << u << '\t' << "v = " << v << endl;
         }
   private:
            int u, v ;
};
void main()
{ BASE obj1;
       A obj2;
   obj1.get( 6, 9, 8, 7 );
       obj2.get( 8, 3, 5, 6 );
       obj1.print();
       obj2.print();
 }
```

```
3). #include <iostream.h>
   class BASE
   { public:
        virtual void getxy( int i,int j = 0 )
            { x = i; y = j; }
        virtual void fun() = 0 ;
   protected:
         int x , y;
} ;
class A: public BASE
{ public:
    void fun()
      { cout << "x = " << x << '\t' << "y = x * x = " << x * x << endl; }
};
class B:public BASE
{ public:
       void fun()
            { cout << "x = " << x << '\t' << "y = " << y << endl;
           cout << "y = x / y = " << x / y << endl;
            }
} ;
void main()
{  BASE * pb;
   A obj1;
   B obj2;
   pb = &obj1;
   pb -> getxy( 10 );
   pb -> fun();
   pb = &obj2;
   pb -> getxy( 80, 5 );
   pb -> fun();
}
```

```
4).#include < iostream.h >
  void main()
  { double x = 123.456;
      cout.width( 10 );
      cout.setf( ios :: dec, ios :: basefield );
      cout << x << endl;
      cout.setf( ios :: left );
      cout << x << endl;
```

```
    cout.width( 15 );
    cout.setf( ios::right , ios::left );
    cout << x << endl;
    cout.setf( ios::showpos );
    cout << x << endl;
    cout << -x << endl;
    cout.setf( ios :: scientific );
    cout << x << endl;
  }
```

5)．#include <iostream.h>

```
class Bclass
{  public:
       Bclass( int i, int j )
    { x = i; y = j; }
      virtual int fun() { return 0 ; }
     protected:
        int x, y ;
};
class Iclass:public Bclass
{ public :
     Iclass( int i, int j, int k ) : Bclass( i, j )
          { z = k; }
       int fun() { return ( x + y + z ) / 3; }
   private :
      int z ;
};
void main()
{ Iclass obj( 2, 4, 10 );
   Bclass p1 = obj;
   cout << p1.fun() << endl;
   Bclass & p2 = obj ;
   cout << p2.fun() << endl;
   cout << p2.Bclass :: fun() << endl;
   Bclass *p3 = &obj;
   cout << p3 -> fun() << endl;
}
```

4、Create a class **RationaNumber**(fractions) with the following capabilities: （12 scores）

a) Enable input and output of fractions through the overloaded >> and << operators.

b) Create a constructor that prevents a 0 denominator in a fraction.

c) Overloaded the addition operator (+), subtraction operator (-) for this class.

```
Class RationalNumber{
int nomi, denom;
public:
RationalNumber( int, int);
RationalNumber& operator+( RationalNumber& );
RationalNUmber& operator-( RationalNumber& );
或者
Friend RationalNumber& operator+( RationalNumber& , RationNumber&);
…

Friend ostream& operator<<( ostream&, RationalNUmber&);
Friend istream& operator>>( istream&, RationalNumber&);
};

Operator+的原理:
RationalNUmber result; result.denom = denom1*denom2;
Result.nom = nomi1*denom2 + nomi2*denom1;
If( result.denom% result.nomi == 0 || result.nomi%result.denom == 0)
{
Int quot = result.denom/result.nomi;
If( quot == 0) quot = result.nomi/result.denom;
}
Result.denom /= quot, result.nomi/= quot;
```

有更好的算法的话就指正吧

5、Implement the **Shape** hierarchy shown in fig.1. Each **TwoDimentsionShape** should contain function **getArea** to calculate the area of the two-dimensional shape. Each **ThreeDimentsionShape** should contain functions **getArea** and **getVolume** to calculate the surface area and volume of the three-dimensional shape, respectively. Create a program that uses a **vector**, determine whether each shape is a **TwoDimentsionShape** or a **TwoDimentsionShape**. If shape is a **TwoDimentsionShape**, display its area. If shape is a **ThreeDimentsionShape**, display its area and volume. （16 scores）

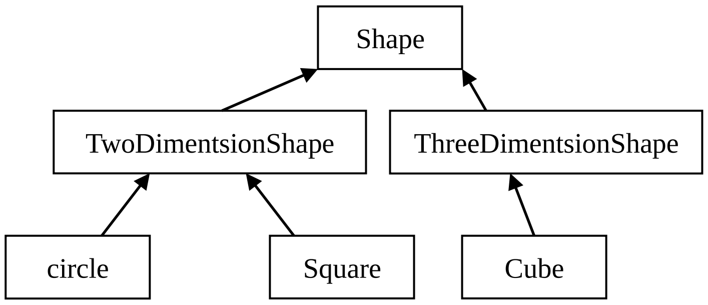

Fig.1 Shape Hierarchy

```
#include <iostream>
#include <typeinfo>
Using namespace std;

Class Shape{
Public:
Virtual double getarea() = 0;
};
Class twoDshape : public Shape{};
Class threeDshape : public shape{public: double getvolume();};
Class circle : public twoDShape{};
Class square : public twoDshape{};
Class cube : public threeDshape{};
(… you know all about those boring implementations don’ t cha?...)

Int main()
{
Char name2d[50], name3d[50];
Strcpy( name2d, typeid( twoDshape).name());
Strcpy( name3d, typeid( threeDshape).name());

Shape* shapearray[5];
Shapearray[0] = new circle;
Shapearray[1] = new cube;

If( strcmp( typeid(*shapearray[0]).name(), name2d) == 0)
   Cout<<Shapearray[0]->getarea();

If( strcmp( typeid(*shapearray[0]).name(), name3d) == 0)
   Cout<<shapearray[0]->getcube()<<”  “<<shapearray[0]->getarea();

If( strcmp( typeid(*shapearray[1]).name(), name2d) == 0)
   Cout<<Shapearray[1]->getarea();

If( strcmp( typeid(*shapearray[1]).name(), name3d) == 0)
   Cout<<shapearray[1]->getcube()<<”  “<<shapearray[0]->getarea();

Return 0;
}
```