座位号_____ 专业_____ 学院_____ 学号□□□□□□□□□ 姓名_____

（密封线内不答题）

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

《高级语言程序设计（2）》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. 所有答案写在答题纸上，答在其它地方无效；
3. 试卷可做草稿纸，试卷必须与答题纸同时提交；
4. 考试形式：闭卷；
5. 本试卷共五大题，满分 100 分，考试时间 120 分钟。

一. 单项选择题（每题 2 分,共 20 分）

1. 在 C++中，有关类和对象正确说法是（ A ）。
A. 对象是类的一个实例　　B. 对象是类的存储空间
C. 一个类只能有一个对象　　D. 类是程序包，对象是存储空间

2. 在类定义中，称为接口的成员是（ C ）。
A. 所有类成员　　B. private或protected的类成员
C. public的类成员　　D. public或private的类成员

3. 一个类的友员函数能够通过（ D ）访问该类的所有成员。
A. 静态数据　　B. 析构造函数　　C. this 指针　　D. 类对象参数

4. 下面描述错误的是（ B ）。
A. 自定义构造函数应该是公有成员函数　　B. 构造函数可以是虚函数
C. 构造函数在建立对象时自动调用执行　　D. 构造函数可以重载

5. 在类的继承关系中，基类的（ B ）成员在派生类中可见。
A. 所有　　B. public和protected
C. 只有public　　D. 只有protected

6. 设 B 类是 A 类的派生类，有说明语句
A a, *ap; B b, *bp;
则以下正确语句是（ C ）。
A. a=b;　　B. b=a;　　C. ap=&b;　　D. bp=&a;

7. C++中，以下（ D ）语法形式不属于运行时的多态。
A. 根据if语句的求值决定程序流程　　B. 根据基类指针指向对象调用成员函数
C. 根据switch语句的求值决定程序流程　　D. 根据参数个数、类型调用重载函数

8. 假设对 A 类定义一个重载“+”号运算符的成员函数,以便实现两个 A 类对象的加法,并返回相加结果,则该成员函数的函数原型为（ B ）。
A. A operator +(const A &A1, const A &A2);　　B. A A:: operator +(const A &A2);
C. A::operator +(A &A2);　　D. A A::operator +( );

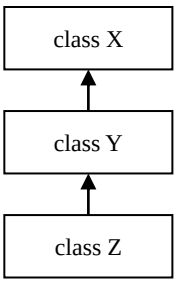
9. 一个类模板定义了静态数据成员，则（ A ）。
A. 每一个实例化的模板类都有一个自己的静态数据成员。
B. 每一个实例化的对象都有一个自己的静态数据成员。
C. 它的类型必须是类模板定义的抽象类型。
D. 所有模板类的对象公享一个静态数据成员。

10. 读一个 C++数据文件，要创建一个（ A ）流对象。
A. ifstream　　B. ofstream　　C. cin　　D. cout

二．简答题（每小题 4 分，共 20 分）

1．有右图所示类格。类 X 中有数据成员 int a。根据以下函数注释的编译信息，分析 int X::a 的访问特性，class Y 对 class X 和 class Z 对 class Y 的继承性质。

```
void Y::funY() { cout<<a<<endl; }      //正确
void Z::funX() { cout<<a<<endl; }      //错误
void main()
{ X x;  Y y;  Z z;
  cout<<x.a<<endl;       //正确
  cout<<y.a<<endl;       //错误
  cout<<z.a<<endl;       //错误
}
```



int X::a 是 class X 的 public 数据成员，class Y 为 protected 继承 class X，class Z 为 private 继承 class Y。

2．有人定义一个教师类派生一个学生类。他认为“姓名”和“性别”是教师、学生共有的属性，声明为 public，“职称”和“工资”是教师特有的，声明为 private。在学生类中定义特有的属性“班级”和“成绩”。所以有

```
class teacher
{ public:
    char name[20];  char sex;
  private:
    char title[20]; double salary;
};
class student : public teacher
{ private:
    char grade[20] ; int score;   };
```

你认为这样定义合适吗？请做出你认为合理的类结构定义。

```
class person
{ public:
    char name[20];  char sex;
};
class teacher : public person
{ private:
    char title[20]; double salary;
};
class student : public person
{ private:
    char grade[20] ; int score;
};
```

3．有类定义

```
class Test
{  int a,b;
 public:
    Test ( int m, int n ) { a=m; b=n; }
```

```
    void Set( int m, int n ) { a=m; b=n; }
    //……
};
```

有人认为“Test 和 Set 函数的功能一样，只要定义其中一个就够了”。这种说法正确吗？为什么？

带参数的构造函数用于建立对象数据初始化，成员函数用于程序运行时修改数据成员的值。

4. 若有声明

```
template <typename T>   class Tclass { /*……*/ } ;
```

建立一个 Tclass 对象用以下语句

```
Tclass Tobj;
```

有错误吗？若出错，请分析原因，并写出一个正确的说明语句。

没有实例化类属参数。

```
Tclass Tobj<int>;
```

5. C++的文本文件可以用 binary 方式打开吗？若有以下语句

```
fstream of("d:testfile", ios::out|ios::binary);
double PI=3.1415;
```

请写出把 PI 的值写入文件"d:testfile"末尾的语句。

可以。

```
of.seekp(0,ios::end);
of.write((char*)&PI, sizeof(double));
```

三. 阅读下列程序，写出执行结果（每题 6 分，共 24 分）

1.

```
#include <iostream.h>                    //运算符重载
enum BoolConst { False=0 , True=1 };   //定义枚举类型
class Boolean
  { public:
    Boolean(BoolConst x = False) { logic = x; }
    void print() const { logic? cout<<"  TRUE  " : cout<<"  FALSE
"; }
    friend Boolean operator +(const Boolean & obj1, const Boolean &
obj2);
    friend Boolean operator *(const Boolean & obj1, const Boolean &
obj2);
    protected:   BoolConst logic;
  };
Boolean operator+ ( const Boolean & obj1, const Boolean & obj2 )
  {return (obj1.logic||obj2.logic)?Boolean(True):Boolean(False);}
Boolean operator* ( const Boolean & obj1, const Boolean & obj2 )
  {  return  (obj1.logic  &&  obj2.logic)  ?  Boolean(True)  :
Boolean(False); }
void main()
{ Boolean a(False), b(True), c, d ;
```

```
    c = a * b;  d = a + b;
    a.print(); b.print(); c.print(); d.print();
    cout<<endl;
}
```

FLASE TRUE FALSE TRUE

2.

```
#include <iostream.h>        //模板，静态数据成员
template <typename T>
class List
{ public:
    List(T x=0) { data = x; }
    void append(List *node ) { node->next=this; next=NULL; total++;}
    List *getnext() { return next; }
    T getdata() {return data; }
   static int total;
  private:
    T data;
    List *next;
};
template <typename T> int List<T>::total=0;
void main()
{ int i, n=5;
  List <int> headnode;  List <int> *p, *last;
  last = &headnode;
  for( i=1; i<=n; i++ )
    { p = new List<int>(i*2);  p->append( last );  last = p; }
  p = headnode.getnext();
  while( p )
    { cout << p->getdata() <<"  ";  p = p->getnext();  }
  cout<<endl;
  cout<<"total="<<List<int>::total<<endl;
}
```

2 4 6 8 10

Total=5

3.

```
#include<iostream.h>        //类成员
#include<math.h>
class  Point
{ public:
   Point(int x1=0, int y1=0)
     { x = x1; y = y1; cout<<"Point 构造函数\n"; }
    int GetX()  {  return x;  }
   int GetY()  {  return y;  }
  private:
```

```
    int x;  int y;
};

class Distance
{ public:
    Distance(Point xp1, Point xp2);
    double GetDis()  {  return dist;  }
  private:
    Point p1;
    Point p2;
    double dist;
};
Distance::Distance(Point xp1, Point xp2): p1(xp1), p2(xp2)
{ cout<<"Distance 构造函数\n";
  double x = double(p1.GetX() - p2.GetX());
  double y = double(p1.GetY() - p2.GetY());
  dist = sqrt(x * x + y * y);
}
void main()
{ Point myp1(0,0), myp2(0,20);
  Distance mydist(myp1,myp2);
  cout<<"The distance is "<<mydist.GetDis()<<endl;
}
```

Point构造函数
Point构造函数
Distance构造函数
The distance is 20

4. 写出 data.txt 中的结果和屏幕显示的结果。

```
#include <fstream.h>
void main()
{ int a=10; double x=50.5;
  char str[10],  fname[20] = "d:\\data.txt";
  fstream  iofile(fname,ios::out);
  if(!iofile)  return;
  iofile<<"Data:\t"<<a+10<<" "<<x<<endl;
  iofile.close();
  iofile.open(fname, ios::in);
  if(!iofile)  return;
  iofile>>str>>a>>x;
  cout<<"string="<<str<<"\n"<<"a="<<a<<", x= "<<x<<endl;
}
```

Data.txt

Data: 20 50.5

输出

string= Data:
a=20, x=50.5

四. 根据程序输出填空。（每空 2 分，共 24 分）

1. //成员和友员

```
#include<iostream.h>
class Time
{ public:
   Time(int h, int m)    {hours=h; minutes=m;}
   ________(1)________ Time12();                    void
   ________(2)________ Time24(Time time);      friend void
  private: int hours, minutes;
};
_________(3)__________ Time12()                  void Time::
{if(hours>12)  { cout<<hours-12<<":"<<minutes<<"PM\n" ; }
 else   cout<<hours<<":"<<minutes<<"AM\n" ;
}
void Time24(Time time)
{                       cout          _______(4)_______          ;}
<<time.hours<<":"<<time.minutes<<"\n"
void main()
{Time T1(20,30), T2(10,45);
 T1.Time12();  Time24(T1);  T2.Time12();  Time24(T2);
}
```

程序输出：

```
    8:30PM
    20:30
    10:45AM
    10:45
```

2. //虚继承

```
#include<iostream.h>
class A
{ public:
    A(const char *s){cout<<s<<'\t';}
    ~A(){}
};
class B: _____(5)_____  A            virtual public
{ public:
    B(const char *s1, const char *s2):A(s1){cout<<s2<<'\t';}
};
class C: _____(6)_____               virtual public A
{ public:
    C(const char *s1, const char *s2):A(s1){ cout<<s2<<'\t';}
```

```
};
class D:public B, public C
{ public:
    D(const char *s1,const char *s2,const char *s3,const char *s4)
      : _____(7)_____          A(s1),B(s1,s2),C(s1,s3)
   {cout<<s4<<'\t';}
};
void main()
{ D *ptr=new D("class A","class B","class C","class D");
 delete ptr;
}
```

程序输出：

```
    class A       class B       class C       class D
```

3.

```
#include<iostream.h>     //继承
#include<string.h>
class studentID
{ public:
    studentID ________(8)________   //(int d=0)       构造函数的默认参数
     {  value=d;  cout<<value<<'\t'; } ;
  protected:
    int  value;
} ;
class student : public studentID
{ public:
    student    ______(9)______     (char    *pname="no    name",int ssID=0):studentID(ssID)
    { strncpy(name, pname, sizeof(name));
      name[sizeof(name)-1]='\0';    cout<<name<<'\n';
    } ;
  protected:
    char name[20];
};
void main()
{  student  s1("Ranry",9818) ,  s2("Jenny"),  s3; }
```

程序输出：

```
    9818   Ranry
       0   Jenny
       0    no name
```

4. //多态

```
#include <iostream.h>
class p_class
{    int  num ;
   public :
     void  set_num( int  val ){ num=val;}
     void  show_num( ) ;
};
void  p_class :: show_num( ){ cout<<num<< '\t';}
void main()
{ p_class ob[3], *p ;
  for( int i=0; i<3;i++ ) ob[i].set_num((i+1)*15);
        (10)      ;  p->show_num( );     p=ob
        (11)      ;  p->show_num( );     p=ob+2
        (12)      ;  p->show_num( );     p=ob+1
}
```

程序输出：

15　　45　　30

五、完成程序。（第 1 小题 4 分，第 2 小题 8 分，共 12 分）

1. 根据程序输出，以最小形式补充 A 类和 B 类的成员函数。

```
#include < iostream.h >
class  A
{ public :
    //A 类的成员函数
    virtual ~A(){cout<<"A_object destroyed.\n";}
};
class  B : public  A
{ public :
    //B 类的成员函数
    ~B(){cout<<"B_object destroyed.\n";}
};
void main ( )
{ A * p=new B ;  delete p; }
```

输出：

```
B_object destroyed.
A_object destroyed.
```

2. 给出基类 Figure 定义和 main 函数如下:

```
class Figure
{ protected :
    double x,y;
  public:
    void set(double i, double j=0) { x=i; y=j; }
    virtual void showarea()const = 0 ;
```

```
  };
#include<iostream.h>
void main()
{ Triangle t;   Square s;
  t.set(10.48,50);
  t.showarea();
  s.set(888,100);
  s.showarea();
}
```

编写派生类 Triangle 和 Square 的最小定义,以便在 main 函数中调用派生类函数 showarea()的不同实现版本求直角三角形和矩形的面积。

```
class Triangle : public Figure
{ public :
     void showarea()const
    { cout<< "Triangle with high "<<x<<" and base "<<y;
      cout<< " has an area of "<<x*0.5*y<<"\n"; }
     };
class Square : public Figure
{ public:
     void showarea()const
   { cout<<"Square with dimension "<<x<<"*"<<y;
     cout<<" has an area of "<<x*y<<"\n";
   }
  };
```