中山大学计算机科学系 2006 级

计算机科学与技术专业、网络工程专业、信息安全专业（ABCDE 班）

程序设计　A 卷

学号 _____　姓名 _____　成绩_____

(试卷共 6 页，答案全写在答题纸上，交卷时连试卷一同交回)

考试形式：闭卷　　任课老师：林瑛、肖菁、杨永红　　2007-6

警示 **《中山大学授予学士学位工作细则》第六条："考试作弊不授予学士学位。"**

一、单项选择 (每小题 1 分，共 15 分)

1. C++语言新引入了在一种 C 语言中没有的参数传递方式是：
   A) 按指针调用　B) 按名调用　C) 按值调用　D) 按引用调用

2. 在 C++语言中，以下哪个表达式采用了八进制表示整型常量：
   A) k=0123;　B) k=123;　C) k='\x23';　D) k=0x123;

3. 下面对结构或类中成员的访问不正确的是：
   A). *p.salary (p 为指向类对象的指针)
   B) p->salary (p 为指向类对象的指针)
   C) (*p).salary (p 为指向类对象的指针)
   D) Worker.salary (Worker 是类类型的对象)

4. 类 A 中有一成员函数说明如下 void A::Set(A & a); 其中 A & a 的含义是：
   A) 指向类 A 的指针为 a
   B) 变量 A 与 a 按位与作为函数 Set()的参数
   C) 将 a 的地址值赋给变量 Set
   D) a 是类 A 的对象引用，用作函数 Set()的形参

5. 假定一个类有两个数据成员 a 和 b，其构造函数为：

```
A(int aa=1,int bb=0){ a = aa;  b = bb;  }
```

   则执行语句 A x(4); 后，x.a 和 x.b 值分别是：
   A) 1 和 0　B) 1 和 4　C) 4 和 0　D) 4 和 1

6. 可以用友元方式重载的运算符是：
   A) + :: <<　B) = >> /　C) + & []　D) + || !

7. 设有如下声明的类：

```
class FOO {
    private:
        static float std;
        float max, min;
};
```

   则表达式 sizeof(FOO)的值为：
   A) 4　B) 8　C) 12　D) 16

8. 若在一个类中用成员函数重载了某种二元运算符@，而 obj1 和 obj2 都是该类的对象，则表达式 obj1@obj2 被 C++编译器解释为：
   A) obj1.operator@(obj2)
   B) obj2.operator@(obj1)
   C) operator@(obj1,obj2)
   D) operator@(obj2,obj1)

9. 下列函数中，不能重载的是：
   A) 类的成员函数　B) 非成员函数　C) 析构函数　D) 构造函数

10. 关于构造函数**不正确**的说法是：
    A) 构造函数可以有返回值
    B) 一个类可以有多个构造函数
    C) 构造函数名与类名相同
    D) 构造函数初始化时为对象开辟一个内存

11. 假定 AB 为一个类，则执行语句

```
AB  a(2), *p[3], b[4];
```

时，自动调用该类构造函数的次数为：

A) 3　　B) 5　　C) 6　　D) 9

12.
```
template<class T>
class APPLE{……};
```

定义类模板 APPLE 的成员函数的正确格式是：

A) `T APPLE<T>::Push(T obj) {……}`　　B) `T APPLE::Push(T obj) {……}`

C)
```
template<class T>
T APPLE::Push(T obj) {……}
```

D)
```
template<class T>
T APPLE<T>::Push(T obj) {……}
```

13. 假设程序中已有#include <string.h>，从而可用字符串的库函数。以下声明了一个字符串 name，并设置它的值为"Computer"，然后输出该字符串。在以下用法中，哪个有可能引起语法错误或逻辑错误？

A) `char* name = "Computer";`

B) `char* name;  strcpy(name, "Computer");`

C) `char name[9] = {'C', 'o', 'm', 'p', 'u', 't', 'e', 'r', '\0'};`

D) `char name[] = "Computer";`

14. 如果类 A 被说明成类 B 的友元，则：

A) 类 A 的成员函数不得访问类 B 的成员　　B) 类 A 的成员即类 B 的成员

C) 类 B 不一定是类 A 的友元　　D) 类 B 的成员即类 A 的成员

15. 关于虚基类的描述，正确的是：

A) 虚基类的唯一副本只被初始化一次

B) 无论是虚基类还是普通基类，其构造函数的调用的次序取决于基类在声明时的次序

C) 类中对象成员的初始化先于虚基类副本的初始化

D) 虚基类的析构函数最先调用

二、程序改错：指出以下题目所示程序段的语法错误(请通过行号来指出错误位置)，说明其错误原因并改正之！(8 个错误，每个错 2.5 分：指出错误位置 0.5 分；错误原因 1 分；改正 1 分，共 20 分)

1. 下列程序段包含 1 个错误：

```
(1) template <class ITEM, class LINK>
(2) int compare(ITEM source, ITEM target)
(3) {     if (source > target) return 1;
(4)       else return 0;
(5) }
```

2. 下列程序包含了 2 个错误，请在**不修改主函数**（假设主函数完全正确）的前提下改正之：

```
(1)  #include <iostream.h>
(2)  template <class TYPE>
(3)  class BASE {
(4)  public:     void show(TYPE obj)
(5)        {     cout << obj << "\n";}
(6)        void test()
(7)        {     cout << "Testing\n";}
(8)  };
(9)  template <class TYPE, class TYPE1>
(10) class DERIVED: BASE<TYPE1> {
(11) public:     void show(TYPE obj1, TYPE1 obj2)
(12)       {     cout << obj1 << "\n";
(13)             BASE::show(obj2);
(14)       }
(15) };
(16) void main()
(17) {  DERIVED<char*, double> obj;
```

```
(18)     obj.test();
(19) }
```

3. 假设 POINT 类的定义完全正确（即不允许改变 POINT 类的定义），下列程序有 1 个错误：

```
(1)  class POINT {
(2)  public:     POINT(int x1, int y1): x(x1), y(y1) {}
(3)  private:    int x, y;
(4)  };
(5)  class LINE {
(6)  public:     LINE(int x0, int y0, int x1, int y1)
(7)        {     start.x = x0;      start.y = y0;
(8)              end.x = x1;     end.y = y1;
(9)        }
(10) private:    POINT start, end;
(11) };
(12) void main()
(13) {     LINE line(0, 0, 50, 50);
(14) }
```

4. 下列程序包含了 4 个错误，请在**不修改主函数**（假设主函数完全正确）以及**不添加任何函数**的前提下改正之：

```
(1)  class MyClass{
(2)  public:
(3)      MyClass(int ini) { member = ini; }
(4)      int GetMember() const { return member; }
(5)      void SetMember(int m) { member = m;}
(6)      void ~MyClass(){ }
(7)  private:
(8)      int member = 0;
(9)  }
(10) void main()
(11) {  MyClass obj1;
(12)     MyClass obj2(3);
(13)     obj1.SetMember(10);
(14) }
```

三、程序输出(共 30 分)：写出以下程序的输出结果

1. (4.5 分)

```
#include <iostream.h>

template<class TYPE>
TYPE max(TYPE k, TYPE t)
{  cout<< ”Calling generic version of max(” << k << ”,” << t
       << "),the max is :" ;
   return (k>t?k:t);
}

int max(int k, int t)
{  cout<< ”Calling special version of max(” << k << ”,” << t
       << "),the max is :" ;
   return (k>t?k:t);
}
void main()
{  cout<<max(10, 20)<<"\n";
   cout<<max(10.5, 20.7)<<"\n";
   cout<<max('B', 'A')<<"\n";
}
```

2. (6 分)

```
#include<iostream.h>
class BASEA {
public:  BASEA( ){ cout<<"This is BASEA class!\n";   }
};
class BASEB {
public:  BASEB( ){ cout<<"This is BASEB class!\n";   }
};
class DERIVEA : public BASEB, virtual public BASEA{
public:  DERIVEA( ){   cout<<"This is DERIVEA class!\n";}
};
class DERIVEB : public BASEB, virtual public BASEA{
public:  DERIVEB( ){   cout<<"This is DERIVEB class!\n";}
};
class TOPDERIV : public DERIVEA, virtual public DERIVEB{
public:  TOPDERIV( ){  cout<<"This is TOPDERIV class!\n";   }
};
void main( )
{  TOPDERIV topobj; }
```

3. (6分)

```
#include<iostream.h>
class BASEX{
protected:  int x, y;
public:     BASEX(int i, int j){ x=i; y=j; }
            void print( )
            {   cout<<"X="<<x<<"\t Y="<<y<<endl;  }
};
class BASEY : public BASEX{
            int k;
public:     BASEY(int i, int j) : BASEX(i, j){   k=i*j; }
            void print( )
            {   cout<<"X="<<x<<"\t Y="<<y<<"\t K="<<k<<endl;   }
};
class BASEZ : BASEY{
public:     BASEZ(int i, int j) : BASEY(i, j){ }
            void printA( )
            {   cout<<"X="<<x<<"\t Y="<<y<<endl;  }
            void printB( )
            {   BASEY::print( ); }
};
void main( )
{  BASEZ obj1(10,20);  BASEY obj2(23,45);
   obj2.print( );      obj1.printB( );     obj1.printA( );
}
```

4. (13.5分)

```
#include <iostream.h>
#include <string.h>
const int CODELEN = 20;

class DEPART {
public:  DEPART(char *depCode = "Math")
         {   strcpy(code, depCode);
             cout << "Constructing depart: [" << code << "].\n";
         }
         DEPART(const DEPART& other)
         {   strcpy(code, other.code);
             cout << "Copy constructing depart: [" << code << "].\n";
         }
         ~DEPART()
```

```
            {   cout << "Destructing depart: [" << code << "].\n";
            }
            void operator=(const DEPART& other)
            {   cout << "Calling operator =, set [" << code;
                cout << "] equal to [" << other.code << "].\n";
                strcpy(code, other.code);
            }
    private: char code[CODELEN+1];
    };

    class EMPLOYEE {
    public:  EMPLOYEE( char *empCode = "Teacher",
                       char *depCode = "Computer"):depart(depCode)
            {   strcpy(code, empCode);
                cout << "Constructing employee: [" << code << "].\n";
            }
            ~EMPLOYEE()
            {   cout << "Destructing employee: [" << code << "].\n";
            }
            DEPART get_depart()
            {   return depart;
            }
    private: char code[CODELEN+1];
            DEPART depart;
    };

    void main()
    {  DEPART dep;
       EMPLOYEE emp;
       dep = emp.get_depart();
    }
```

四、程序填空（每空 2 分，共 16 分）：根据以下各小题的描述和要求在指定位置填入适当语句

1. 完成如下的程序，使得输出结果为：

```
base::10
base::12
derived::24
```

```
#include <iostream.h>
class base {
            int x;
public:     base(int a) { x=a; }
            ___①___{ cout<<"base::"<< x << endl; }
};
class derived: public base {
            int y;
public:     derived(int a,int b):base(a) { y=b; }
            void print() { ___②___; cout<<"derived::"<<y<<endl; }
};
void main()
{     base b(10), *p;
      derived d(12,24);
      b.print();
      ___③___;
      p->print();
}
```

2. 函数 int commstr(char *str1, char *str2, int *sublen)从两已知字符串 str1 和 str2 中，找出它们的所有最长的公共子串。如果最长公共子串不止 1 个，函数将把它们全部找出并输出。

函数将最长公共子串的长度送入由参数 sublen 所指的变量中，并返回字符串 str1 和 str2 的最长公共子串的个数。约定空串不做为公共子串。例如，如使用下列主函数调用该函数，程序的执行结果为：

```
456
abc
n=2 len=3
```

请根据题目要求填入适当的语句。

```
#include <string.h>
#include <iostream.h>

int commstr(char *str1, char *str2, int *sublen)
{     char *s1, *s2;
      int count = 0, len1, len2, k, j, i, p;
      len1   = strlen(str1);
      len2   = strlen(str2);
      if (len1>len2) {      s1      = str1;     s2  = str2;    }
      else { len2    = len1;s1      = str2;     s2  = str1;    }
      for (j=len2; j>0; j--) {
             for (k=0;_____④_____<= len2; k++)
                    for (i=0; s1[_____⑤_____]!='\0'; i++) {
                           for (p=0; p<j &&_____⑥_____; p++);
                           if (_____⑦_____) {
                                  for (p=0; p<j; p++) cout<<s2[k+p];
                                  cout<<endl;
                                  count++;
                           }
                    }
             if (count>0) break;
      }
      *sublen = (count>0)?_____⑧_____:0;
      return count;
}

void main()
{     int sublen,n;
      n = commstr("abc1234567","45612abc6",&sublen);
      cout<<"n="<<n<<" len="<<sublen;
}
```

五、程序设计（19 分）

1. (7 分)类属类 LIST 描述一个集合。集合中的元素记录在一个长度为 n 的数组 array 中，要求 array 根据使用时的实际长度 n 动态分配。LIST 提供如下操作：赋值运算（将一个 LIST 对象赋给另 LIST 对象），以友元形式重载了运算符+实现两个 LIST 对象的数组对应位置上的元素相加，求该 LIST 数组的最大值。LIST 的用法如以下程序所示。要求给出 LIST 的类界面。

   提示：定义该类属类的操作，使其可以实现主程序中使用到该类的功能，不必提供 LIST 类的实现。

```
void main()
{  LIST<int> s1, s2;
   LIST<int> s3(10); // 10 是数组长度
   LIST<int> s4=s1;  //调用 copy constructor 而不是赋值运算符；等价于 s4(s1)
   int max1;
   ……
   s3    = s1 + s2; //调用赋值运算符重载和重载运算符+
   max1  = s3.max();
}
```

2. (12 分)设计一个词典类 Dic，每个单词包括英文单词及对应的中文含义，提供构造这个词典的操作，并有一个英汉翻译成员函数，通过查词典的方式将一段英语翻译成对应的汉语。

   提示：例如；要把英语“I am a student”翻译为中文“我是一个学生”，则应先在词典类中添加("a","一个")、("I","我")、("am","是")、("student","学生")中英文词对。