


北京科技大学 2017-2018 学年 第一学期

程序设计基础（C++）期末模拟试卷

一、选择题

1. 下列说法中正确的是（ ）
 - A. C++中的输入可以使用 cin，动态数组的优点在于存取性能比静态数组高
 - B. 程序中的其他函数可以调用主函数
 - C. C++中的字符指针在大多数情况下都指的是一个字符串
 - D. 一个 cin 可以输入多项数据，每项前面加左移运算符
2. 在 `int x=-9; while(++x){}` 中，循环体一共被执行了（ ）次
 - A. 7
 - B. 8
 - C. 9
 - D. 10
3. 关于 C++中运算和运算符的表述，下列说法中错误的是（ ）
 - A. 条件运算符的优先级高于赋值运算符和逗号运算的优先级
 - B. 关系运算符是双目运算符，优先级低于算数表达式，是右结合的
 - C. 自增运算符的运算对象只能是变量，不能是表达式或常量
 - D. 可以把两个整型变量做除法运算后得到的结果赋给浮点型变量
4. 以下程序段中由 while 构成的循环执行的次数为（ ）
`int k = 0; while(k = 1) k++;`
 - A. 无限次
 - B. 一次也不执行
 - C. 执行一次
 - D. 有错误，不能执行
5. 以下关于 C++控制语句的表述中错误的是（ ）
 - A. 可在两层 for 循环的内层 for 循环体中使用 break 语句跳出两层循环
 - B. 当程序执行到 continue 语句时，continue 以下的语句将不会被执行
 - C. `for(int i = 0; i < 5; i++);` 是个完整的语句，但循环体为空语句
 - D. 在 switch 中，case 后面可以是单个语句、多个语句，也可以无语句
6. 对下列函数的叙述错误的是（ ）

```
void swap(int * p, int * q)
{
    int * temp;
    temp = p;
    p = q;
    q = temp;
}
```

 - A. 可以通过在主函数中调用该函数来实现两个整型变量的交换
 - B. 定义 temp 指针之后，程序使其指向了 p 所指向的内存地址
 - C. 在主函数中调用该函数时，应传入两个指向整型的指针变量
 - D. 当函数执行完成后，系统将释放 12 个字节的存储空间

7. 假定所有变量均已正确说明，下列程序段运行后，x 的值是()。

```
a=b=c=0;
x=35;
if (!a)    x--;
else if (b);
if (c)    x=3;
else    x=4;
```

A、3 B、35 C、34 D、4

8. C++语言中 while 循环和 do...while 循环的主要区别是()。

A. do...while 的循环体不能是复合语句
B. do...while 的循环体至少无条件执行一次
C. while 的循环控制条件比 do...while 的循环控制条件严格
D. do...while 允许从外部转到循环体内

9. 执行语句序列：

```
int x=3;
do
{
    x-=2;
    cout<<x;
}while(!(--x));
```

输出结果是()。

A. 3 0 B. 死循环 C. 1 -2 D. 1

10. 对二维数组的正确定义是()

A. int a[][]={1,2,3,4,5,6};
B. int a[2,3]={1,2,3,4,5,6};
C. int a[2][]={1,2,3,4,5,6};
D. int a[][3]={1,2,3,4,5,6};

11. 在 int a[][3]={ {1}, {3,2}, {4,5,6}, {0} } 中，a[2][2] 的值是()

A. 2 B. 6 C. 4 D. 3

12. 下列关于数组的叙述，错误的是()

A. 可以用循环的方式输出整个字符数组
B. 当数组越界时，程序并不会报错
C. 可以直接输出字符数组和整型数组
D. 不能通过“=”对字符数组整体赋值

13. 定义如下结构体变量：

```
struct st1{int a, b; float x, y;}s1, s2;
struct st2{int a, b; float x, y;}s3, s4;
```

则下列说法正确的是()

-
- A. s1、s2、s3、s4 可以相互赋值
B. 只有 s1 和 s2、s3 和 s4 之间可以相互赋值
C. s1、s2、s3、s4 之间均不可以相互赋值
D. 结构体变量不可以整体赋值
14. 设有数组定义:char array[]="China";,则数组 array 所占的空间为()
A. 4 个字节 B. 7 个字节 C. 6 个字节 D. 5 个字节
15. 假定已经定义了一个有返回值的函数,那么该函数调用不可以做的是()
A. 作为一个函数的形参 B. 出现在表达式中
C. 作为独立的语句存在 D. 作为一个函数的实参
16. 在一个程序中,如果函数 fA 调用了函数 fB,函数 fB 又调用了函数 fA,那么()
A. 称为函数的间接递归调用
B. 这样调用方式是不允许的
C. 称为函数的直接递归调用
D. 称为函数的循环调用
17. 下列关于函数的表述,错误的是()
A. 在不同的作用域中可以有同名变量存在
B. 两个同名但形参类型不同的函数可以构成重载函数
C. 定义函数体时,可在其内部直接调用此函数
D. 定义函数体时,可在其内部对另一个函数进行定义
18. 指针做形参的函数不能做到的是()
A. 将调用函数中变量的值通过参数传递到被调用函数。
B. 将被调用函数内部的变量地址传递到调用函数。
C. 将被调用函数内部变量的值通过参数传递到调用函数。
D. 在被调用函数内部修改调用函数中变量的值。
19. 若有以下定义,则赋值正确的是()
int a, b, *p;
float c, *q;
A. p=NULL; B. q=p; C. p=&c; D. q=&a;
20. 如果定义 int aa[10]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, *p=aa;则数值为 6 的是()
A. *p+6 B. p+5 C. *p+=5 D. *(p+6)
21. 下列说法中正确的是()。
A. 类定义中只能说明函数成员的函数头,不能定义函数体
B. 类中的函数成员可以在类体中定义,也可以在类体之外定义
C. 类中的函数成员在类体之外定义时必须要与类声明在同一文件中

- D. 在类体之外定义的函数成员不能操作该类的私有数据成员
22. 关于静态成员的描述中，() 是错误的。
- A. 静态成员可分为静态数据成员和静态成员函数
 - B. 静态数据成员定义后必须在类体内进行初始化
 - C. 静态数据成员初始化不使用其构造函数
 - D. 静态数据成员函数中不能直接引用非静态成员
23. 关于多继承二义性的描述，下列说法中错误的是 ()
- A. 派生类的多个基类中存在同名成员时，派生类对这个成员访问可能出现二义性
 - B. 一个派生类是从具有共同的间接基类的两个基类派生来的，派生类对该公共基类的访问可能出现二义性
 - C. 解决二义性最常用的方法是作用域运算符对成员进行限定
 - D. 派生类和它的基类中出现同名函数时，将可能出现二义性
24. 下列关于构造函数的说法，错误的是 ()
- A. 如果在定义类时没有定义类的构造函数系统会生成一个无参构造函数
 - B. 拷贝构造函数是构造函数，该函数中形参可以是其它类对象的引用
 - C. 当函数的返回值是类的对象时，系统调用拷贝构造函数实现拷贝赋值
 - D. 构造函数不能显示调用，调用的具体构造函数根据对象的定义形式确定
25. 在 C++ 中，要实现动态联编，必须使用 () 调用虚函数。
- A. 类名
 - B. 派生类指针
 - C. 对象名
 - D. 基类指针

二、填空题

1. C++ 程序中 `cout<<endl;` 的作用是_____。
2. 在 C++ 语言中，表示一条语句结束的标号是_____。
3. C++ 语言提供的基本控制结构可以分为 3 种类型：顺序结构、_____和循环结构。
4. 如果定义 `int e=8; double f=6.4, g=8.9;`，则表达式 `f+int(e/3*int(f+g)/2)%4` 的值为_____。
5. 输出数组 `char a[] = {'t', 'o', '\\0', 'y', 'o', 'u'}`；的最终结果为_____。
6. 在 C++ 中，系统为 `double * p[20]={p1, p2, p3}` 分配的存储空间为_____个字节（假设 p1、p2、p3 已提前定义为指向 double 型的指针）
7. 定义二维字符数组 `char arr[2][4];`
则 `strcpy(arr[0], "you"); strcpy(arr[1], "me"); arr[0][3]='&; cout<< arr[0];` 输出结果为_____。
8. 设有一下类的定义：
- ```
class A
{
 int a1;
 protected: int a2;
```

---

```

 public: int a3;
};
class B : protected A
{
 int b1;
 protected: int b2;
 public: int b3;
};
class C : private B
{
 int c1;
 protected: int c2;
 public: int c3;
};

```

在派生类 C 的所有数据成员中，访问属性为私有的数据成员有\_\_\_\_\_。

9. 在 C++ 中，虚函数帮助实现了类的\_\_\_\_\_。
10. 派生类的缺省继承方式是\_\_\_\_\_。

### 三、读程序写结果

1.

```

#include <iostream.h>
int a[]={2,4,6,8,10};
int &index(int i)
{
 return a[i];
}
void main()
{
 int i;
 index(3)=12;
 for (i=0;i<=4;i++)
 cout<<a[i]<<" ";
}

```

2.

```

#include <iostream.h>
void f(int *m, int n)
{
 int temp;
 temp=*m;
 *m=n;
 n=temp;
}

```

---

```
}
void main()
{
 int a=5,b=10;
 f(&a, b);
 cout<<a<<" "<<b<<endl;
}
```

3.

```
#include <iostream.h>
int i=15;
void main()
{
 int i;
 i=100;
 ::i=i+1;
 cout<<::i<<endl;
}
```

4.

```
#include<iostream>
using namespace std;
class Test
{
private:
 static int val;
 int a;
public:
 static int func();
 void sfunc(Test &r);
};
int Test::val=200;
int Test::func()
{
 return val++;
}
void Test::sfunc (Test &r)
{
 r.a=125;
 cout<<"Result3="<<r.a<<endl;
}
void main()
{
 cout<<"Result1="<<Test::func()<<endl;
```



北京科技大学学生学习与发展指导中心  
Center for Student Learning and Development USTB

```
Test a;
cout<<"Result2="<<a.func()<<endl;
a.sfunc(a);
}
```

#### 四、程序填空

1. 下列程序中，a、b 分别代表直角三角形的两个直角边之长，输出结果为此三角形的斜边之长。请将程序补充完整。

```
#include <iostream.h>
#include <math.h>
void main()
{
 float a=3,b=4;
 _____;
}
```

2. 下面程序是输出 100 内能被 3 整除且个位数是 6 的所有整数，请将程序补充完整

```
#include<iostream.h>
void main()
{
 int i,j;
 for (i=0; _____;i++)
 {
 j=i*10+6;
 if (_____)
 continue;
 _____;
 cout<<j<<endl;
 }
}
```

3. 填写下列程序中的三个空白部分，使程序的最终输出结果为

pri1=7, pri2=6

pri4=7

pri12=8

待填程序代码如下：

```
#include<iostream>
using namespace std;
class P
{
public:
 P(int p1, int p2) { pri1 = p1; pri2 = p2; }
 int incl() { return ++pri1; }
```

---

```

 int inc2() { return ++pri2; }
 void display () { cout << "pri1=" << pri1 << ",pri2=" << pri2 <<
endl; }
 private:
 int pri1, pri2;
};
class D1 : virtual private P
{
 public:
 D1(int p1, int p2, int p3) : P(p1, p2)
 { pri3 = p3; }
 int incl() { return P::incl(); }
 int inc3() { return ++pri3; }
 void display()
 {
 cout << "pri3=" << pri3 << endl;
 }
 private:
 int pri3;
};
class D2 :
{
 public:
 D2(int p1, int p2, int p4) : P(p1, p2)
 { pri4 = p4; }
 int incl()
 {
 P::incl();
 P::inc2();
 return P::incl();
 }
 int inc4() { return ++pri4; }
 void display()
 {
 P::display();
 cout << "pri4=" << pri4 << endl;
 }
 private:
 int pri4;
};
class D12 : private D1, public D2
{
 public:

```



---

```

D12(int p11, int p12, int p13, int p21, int p22, int p23, int p)
:D1(p11, p12, p13), D2(p21, p22, p23), P(p11, p21)
{ pril2 = p; }
int incl()
{
 D1::incl();
 return D2::incl();
}
int inc5() { return ++pril2; }
void display()
{
 D2::display();
 cout << "pril2=" << pril2 << endl;
}
private:
 int pril2;
};

int main(void)
{
 D12 d(1, 2, 3, 4, 5, 6, 7);
 d.display();
 d.incl();
 d.inc4();
 d.inc5();
 d.D12::incl();

 return 0;
}

```