

一、选择题

1. 由 C++ 目标文件连接而成的可执行文件的缺省扩展名为\_\_\_\_\_。

- A. cpp

B. ~~exe~~

C. obj

D. ~~lik~~

2. 在下面的一维数组定义中，哪一个有语法错误。\_\_\_\_\_

- A. int a[ ]={1,2,3}

B. int a[10]={0}

C. ~~int a[ ]~~

D. int a[5]

3. 在下面的函数声明中，存在着语法错误的是\_\_\_\_\_。

- A. void BC(int a , int)

B. void BD(int , int)

C. void BE(int , int=5)

D. ~~int BF(int x ; int y)~~

4. 假定 AB 为一个类，则该类的拷贝构造函数的声明语句为\_\_\_\_\_。

- A. AB &(AB x)

B. AB(AB x)

C. ~~AB(AB &)~~

D. AB(AB \* x)

5. 对于结构中定义的成员，其隐含访问权限为\_\_\_\_\_。

- A. ~~public~~

B. protected

C. private

D. static

6. 当使用 fstream 流类定义一个流对象并打开一个磁盘文件时，文件的隐含打开方式为\_\_\_\_\_。

- A. ios::in

B. ios::out

C. ios::int | ios::out

D. ~~没有~~

( 1 ) C++ 中，封装是借助于 ( ) 达到的。  
A. 指针 B. ~~类~~ C. 数组 D. 函数

□□ ( 2 ) 下面是关于一个类的友元的说法，其中不正确的是 ( )  
A. 友元函数可以访问该类的私有数据成员  
B. 友元的声明必须放在类的内部  
C. ~~友元函数可以是类的成员函数~~  
D. 类与类之间的友元关系不能被继承

□□□□ ( 5 ) 下面是对类 ttt 的声明。正确的是 ( )  
A. void class ttt;  
B. class int ttt;

- C. class ttt( );

D. ~~class ttt;~~

( 6 ) 下列关于拷贝构造函数的描述中，错误的是 ( B )

- A) 拷贝构造函数可以设置参数

B) ~~拷贝构造函数在定义类对象时自动执行~~

C) 拷贝构造函数可以是内联函数;

D) 拷贝构造函数不可以重载

□□  
□□ ( 8 ) 下列程序引起的后果是 ( C )

3. 对于类中定义的成员，其隐含访问权限为\_ C \_。

- A. public

B. protected

C. ~~private~~

D. static

4. 定义在 \*.h 头文件中的系统库函数，其所属命名空间是\_ B \_。

- A. STD

B. ~~全局~~

C. 自定义

D. 不隶属于任何命名空间

5. 类型兼容规则是指：在需要基类对象的任何地方，可以

使用\_ B \_派生类的对象来代替。

- A. 私有

B. ~~公有~~

C. 保护

D. 任意

6. 含有纯虚函数的类称为\_ B \_。

- A. 虚基类

B. ~~抽象类~~

C. 友元类

D. 虚类

□□ ( 1 ) 面向对象程序设计的四大特点是 ( B )  
□□ A) 安全性、抽象性、继承性、复用性; B) ~~抽象性、封装性、继承性、多态性;~~  
C) 复用性、抽象性、封装性、继承性; D) 安全性、抽象性、继承性、多态性;

□□ ( 2 ) C++ 语言中有四种类型的多态，它们是 ( B )

- A) 重载多态、参数多态、强制多态、运算符多态;

□□ B) ~~重载多态、参数多态、强制多态; 包含多态;~~

□□ C) 静态多态、强制多态;

□□ D) 动态多态、虚函数多态

□□ ( 3 ) 使用 C++ 编写模板类时，对模板类的实现与声明必须是 ( C )

- A) ~~在不同的物理文件中;~~

B) ~~可在、也可不在同一文件中;~~

C) 必须在同一头文件中; D) 必须在同一实现文件中;

二、填空题

1. 执行 “cout <<43<<'\<<18<<'\<<18<<'\<<43-18<<endl;” 语句后得到的输出结果为\_\_\_\_\_。
2. 已知'A'~'Z'的 ASCII 码为 65~90, 当执行 “char ch=14\*5+2; cout <<ch<<endl;” 语句序列后, 得到的输出结果为\_\_\_\_\_。
3. 使用 const 语句定义一个标识符常量时, 则必须对它同时进行\_\_\_\_\_。
4. 表达式 x=x+1 表示成增量表达式为\_\_\_\_\_。
5. 若 x=5, y=10, 则 x>y 和 x<=y 的逻辑值分别为\_\_\_\_\_和\_\_\_\_\_。
6. 执行 “typedef int ABC[10];” 语句把 ABC 定义为具有 10 个整型元素的\_\_\_\_\_。
7. 假定 p 所指对象的值为 25, p+1 所指对象的值为 46, 则执行 “(\*p)++;” 语句后, p 所指对象的值为\_\_\_\_\_。
8. 假定一个二维数组为 a[M][N], 则 a[i]的地址值(以字节为单位)为\_\_\_\_\_。
9. 假定要访问一个结构指针 p 所指对象中的 b 指针成员所指的对象, 则表示方法为\_\_\_\_\_  
\_\_\_\_\_。
10. 设 px 是指向一个类动态对象的指针变量, 则执行 “delete px;” 语句时, 将自动调用该类的\_\_\_\_\_  
\_\_\_\_\_。
11. 若需要把一个函数 “void F( );” 定义为一个类 AB 的友元函数, 则应在类 AB 的定义中加入一条语句: \_\_\_\_\_。
12. 若要在程序文件中进行标准输入输出操作, 则必须在开始的 # include 命令中使用\_\_\_\_\_  
\_\_\_\_\_头文件。
12. 若要在 C++程序文件中使用 COUT 与 CIN 进行标准输入/输出操作, 则必须在程序文件开始的地方包含如下\_\_\_\_\_语句。

- 1、用变量 a 给出下面的定义（写出相应的 C++语句）
- a) 一个整型数 ( )
- b) 一个指向整型数的指针 ( )
- c) 一个指向指针的的指针，它指向的指针是指向一个整型数 ( )
- d) 一个有 10 个整型数的数组 ( )

- e) 一个有 10 个指针的数组，该指针是指向一个整型数的 ( )
- f) 一个指向有 10 个整型数数组的指针 ( )
- g) 一个指向函数的指针，该函数有一个整型参数并返回一个整型数 ( )
- h) 一个有 10 个指针的数组，该指针指向一个函数，该函数有一个整型参数并返回一个整型数 ( )

2、解释下面有关变量 a 的定义

- Const int a;  
解释为: ( )
- int const a;  
解释为: ( )
- const int \*a;  
解释为: ( )
- int \* const a;  
解释为: ( )
- int const \* a const;  
解释为: ( )

(1) 下面程序的打印结果是 [ ]

```
#include <iostream.h>
int f(int,int j);
int main()
{
    try{
        Cout<<"5/3="<<f(5,3)<<endl;
        Cout<<"5/0="<<f(5,0)<<endl;
        Cout<<"5/3="<<f(4,2)<<endl;
    }catch(int i){
        Cout<<i<<" can't devided by zero";
    }
    cout<<"end"<<endl;
    return 0;
}
```

三、下列程序运行后的输出结果

```
1. 写出下列程序的运行结果.
#include<iostream>
Using namespace std;
int i;
namespace NS{
    int j;
}
void main()
{
    i=5;
    NS::j=6
    {
        using namespace NS;
        int i;
        i=7;
        cout<<"i="<<i<<endl;
        cout<<"j="<<j<<endl;
    }
    Cout<<"i="<<i<<endl;
}
```

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

void main()

{

    int s=0;

    for (int i=1; ; i++) {

        if (s>50) break;

        if (i%2==0) s+=i;

    }

    cout <<"i,s="<<i<<","<<s<<endl;

}
```

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

void main()
```

```
{

    char a[]="abcdabcbafgacd";

    int i1=0,i2=0,i=0;

    while (a[i]) {

        if (a[i]=='a') i1++;

        if (a[i]=='b') i2++;

        i++;

    }

    cout <<i1<<' '<<i2<<endl;

}
```

```
3. # include <iomanip.h>

void main()

{

    int a[9]={2,4,6,8,10,12,14,16,18};

    for (int i=0; i<9; i++) {

        cout <<setw(5)<<*(a+i);

        if ((i+1)%3==0) cout <<endl;

    }

}
```

```
4. # include <iomanip.h>

void LE(int * a,int * b) {

    int x=*a;

    *a=*b; *b=x;

    cout <<*a<<' '<<*b<<endl;

}
```

```
void main()
{
    int x=10,y=25;

    LE(&x,&y); cout <<x<<' '<<y<<endl;
}

5. # include <iostream.h>

class A {

    int a,b;

public :

    A() { a=b=0; }

    A(int aa,int bb) {

        a=aa; b=bb;

        cout <<a<<' '<<b<<endl;

    }

};

void main()

{

    A x,y(2,3),z(4,5);

}
```

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

template <class TT>

class FF {

    TT a1,a2,a3;

public :
```

```
FF(TT b1,TT b2,TT b3) {

    a1=b1; a2=b2; a3=b3;

}

TT Sum() { return a1+a2+a3; }

};

void main()

{

    FF <int> x(2,3,4),y(5,7,9);

    cout <<x.Sum()<<' '<<y.Sum()<<endl;

}
```

四、写出下列每个函数的功能

```
1. double SF(double x,int n) {

    // n 为大于等于 0 的整数

    double p=1,s=1;

    for (int i=1; i<=n; i++) {

        p*=x;

        s+=p/(i+1);

    }

    return s;

}

2. float FH() {

    float x,y=0,n=0;

    cin >>x;

    while (x!=-1) {

        n++; y+=x;

    }

}
```

```
        cin >>x;

    }

    if (n==0) return y; else return y/n;

}

3. # include <iostream.h>

void WA(int a[],int n) {

    for (int i=0; i<n-1; i++) {

        int k=i;

        for (int j=i+1; j<n; j++)

            if (a[j]<a[k]) k=j;

        int x=a[i]; a[i]=a[k]; a[k]=x;

    }

}

4. # include <iomanip.h>

# include <fstream.h>

void JB(char * fname)

    // 可把以 fname 所指字符串作为文件标识符的文件称为 fname 文件

    // 假定该文件中保存着一批字符串，每个字符串的长度均小于 20

{

    ifstream fin(fname);

    char a[20];

    int i=0;

    while (fin>>a) {

        cout <<a<<endl;

        i++;

    }
```

```
    }

    fin.close();

    cout <<"i="<<i<<endl;

}
```

四、编写一个函数，统计出具有 n 个元素的一维数组中大于等于所有元素平均值的元素个数并返回。  
int Count(double a[],int n); // 此为该函数的声明  
五、2. 已知下图是一个求解定积分时所使用到的各个类及其相互关系的 UML 设计图，请写出能实现该图中各个类定义的 C++ 源代码（注意：只写出头文件即可 Trapzint.h）

