**模拟试题（第二套）**

**一、单项选择题（每小题2分，共20分）**

1. 字符串”a+b=12\n”的长度为（ ）。

A. 6 B. 7 C. 8 D. 9

2. 假定一个二维数组的定义语句为“int a[3][4]={{3,4},{2,8,6}};”，则元素a[2][1]的值为（ ）。

A. 0 B. 4 C. 8 D. 6

3. 以下正确的描述是（ ）。

A. 函数的定义可以嵌套，函数的调用不可以嵌套

B. 函数的定义不可以嵌套，函数的调用可以嵌套

C. 函数的定义和函数的调用均可以嵌套

D. 函数的定义和函数的调用均不可以嵌套

4. 设有定义“double array[10];”，则表达式sizeof(array)/sizeof(array[0])的结果为array数组的( )。

A.首地址 B.元素个数

C.每个元素所占的字节数 D.总字节数

5. 若使p指向包含30个整型元素的动态数组空间，则使用的定义语句为（ ）。

A. int \*p=new int[30]; B. int \*p=new int(30);

C. int \*p=new [30]; D. \*p=new int[30];

6. 软件产品与其他软件产品组合成一个整体的难易程度叫做软件的（ ）。

A. 可维护性 B. 可复用性 C. 兼容性 D. 正确性

7. 在多文件结构的程序中，通常把类的声明单独存放于（ ）中。

A. 主文件 B. 实现文件 C. 库文件 D. 头文件

8. 在一个用数组实现的队列类中，假定数组长度为MS，队首元素位置为first，队列长度为length，则队首的后一个位置为( )。

A. first+1 B. (first+1)%MS

C. (first-1)%MS D. (first+length)%MS

9. 假定一个类的构造函数为“A(int aa=1, int bb=0) {a=aa; b=bb;}”，则执行“A x(4);”语句后，x.a和x.b的值分别为( )。

A. 1和0 B. 1和4 C. 4和1 D. 4和0

10. 假定AB为一个类，则执行“AB a(2), b[3], \*p[4];”语句时共调用该类构造函数的次数为( )。

A. 3 B. 4 C. 5 D. 9

**二、填空题（每小题2分，共20分）**

1. 用于输出表达式值的标准输出流对象是\_\_\_\_\_\_\_\_\_\_\_\_。

2. 假定x=5，y=6，则表达式x--\*--y的值为\_\_\_\_\_\_\_\_\_\_。

3. 假定x是一个逻辑量，则x || true的值为\_\_\_\_\_\_\_\_\_\_。

4. C++提供的预处理命令有宏定义命令，条件编译命令和\_\_\_\_\_\_\_\_\_\_\_\_命令。

5. 假定a是一个一维数组，则a[i]的指针访问方式为\_\_\_\_\_\_\_\_。

6. 变量v定义为“double v=23.4;”，要使指针pv指向v，则定义pv的语句为\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

7. 若在类的定义体中只给出了一个成员函数的原型，则在类外给出完整定义时，其函数名前必须加上\_\_\_\_\_\_\_\_和两个冒号分隔符。

8. 若采用x.abc(y)表达式调用一个成员函数，在成员函数中使用的\_\_\_\_\_\_\_\_就代表了类外的x对象。

9．假定用户没有给一个名为AB的类定义析构函数，则系统为其定义的析构函数为\_\_\_\_\_\_\_\_\_\_。

10．假定用户只为类AB定义了一个构造函数"AB(int aa, int bb=0) {a=aa; b=bb;}"，则定义该类的对象时，其实参表中至多带有\_\_\_\_\_\_\_\_\_个实参。

**三、程序填充题，根据题意在横线上填写合适的内容。（每小题6分，共24分）**

1. 下面函数是求两个整型参数a和b的最小公倍数。

int f2(int a, int b)

{

int i=2, p=1;

do {

while(a%i==0 && \_\_\_\_\_\_\_\_\_\_\_\_) {

p\*=i; a/=i; b/=i;

}

i++;

}while(a>=i && \_\_\_\_\_\_\_\_\_\_\_\_);

return p\*a\*b;

}

2. 对数组a中的元素按相反次序重新排列。

void PV(int a[], int n)

{

int i;

for(i=0; i<n/2; i++) {

int x=a[i];

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

}

3. 假定有定义为“struct NODE{int data; NODE\* next;};”，下面算法是把以L为表头指针的链表中各结点依次按相反次序链接并返回新链表的表头指针。

NODE\* f8(NODE\* L)

{

if(L==NULL) return NULL;

NODE \*p=NULL, \*q=L, \*t; //p为新链表的表头指针，初始值为空

while(q!=NULL) {

t=q;

q=q->next;

t->next=\_\_\_\_\_\_\_\_\_\_;

p=\_\_\_\_\_\_\_\_\_\_;

}

return p;

}

4. 类A的定义

class A {

int \*a; int n; int MaxLen;

public:

A(): a(0), n(0), MaxLen(0) {} //无参构造函数

A(int \*aa, int nn, int MM) { //带参构造函数

n=nn;

MaxLen=MM;

if(n>MaxLen) exit(1);

a=new int[MaxLen];

for(int i=0; i<n; i++) \_\_\_\_\_\_\_\_\_\_\_\_; //利用数组aa给数组a赋值

}

~A(){\_\_\_\_\_\_\_\_\_\_\_\_\_\_;} //释放动态存储空间

int GetValue(int i) {return a[i];} //函数体返回a[i]的值

};

**四、理解问答题，分别写出前2小题的程序运行结果和指出后2小题的程序或函数功能。（每小题6分，共24分）**

1. #include<iostream.h>

const int B=2;

void main()

{

int p=1,s=1;

while(s<50) {

p\*=B;

s+=p;

}

cout<<"s="<<s<<endl;

}

运行结果：

2. #include<iostream.h>

class CE {

private:

int a,b;

int getmin() {return (a<b? a:b);}

public:

int c;

void SetValue(int x1,int x2, int x3) {

a=x1; b=x2; c=x3;

}

int GetMin();

};

int CE::GetMin() {

int d=getmin();

return (d<c? d:c);

}

void main()

{

int x=5,y=12,z=8;

CE \*ep=new CE;

ep->SetValue(x+y,y-z,10);

cout<<ep->GetMin()<<endl;

}

运行结果：

3. int ff(int x, int y, char op) {

int z;

switch(op) {

case '+':

cout<<x<<'+'<<y<<'='; cin>>z;

if(x+y==z) return 1; else return 0;

break;

case '-':

cout<<x<<'-'<<y<<'='; cin>>z;

if(x-y==z) return 1; else return 0;

break;

case '\*':

cout<<x<<'\*'<<y<<'='; cin>>z;

if(x\*y==z) return 1; else return 0;

break;

case '/':

cout<<x<<'/'<<y<<'='; cin>>z;

if(x/y==z) return 1; else return 0;

break;

default: cout<<"运算符错，退出!"; exit(1);

}

}

函数功能：

4. char\* f(char \*s){

int n=strlen(s);

char\* r=new char[n+1];

for(int i=0; i<n; i++)

if(s[i]>='a' && s[i]<='z') r[i]=s[i]-'a'+'A';

else r[i]=s[i];

r[n]=’\0’;

return r;

}

函数功能：

**五、编程题（每小题6分，共12分）**

1. 按照下面函数原型语句编写一个递归函数求出并返回数组a中n个元素的平方和。

int f(int a[],int n);

2. 根据下面类中拷贝构造函数的原型写出它的类外定义。

class Array {

int \*a; //指向动态分配的整型数组空间

int n; //记录数组长度

public:

Array(int aa[], int nn); //构造函数，利用aa数组长度nn初始化n，

//利用aa数组初始化a所指向的数组空间

Array(Array& aa); //拷贝构造函数

};