一、选择题

C B B D D

二、可访问成员和访问权限

A的可访问成员和访问权限

**private:**

a

**protected:**

b,e

**public:**

c,d

B的可访问成员和访问权限

**private:**

d,A::a

**protected:**

c,e,A::b,A::e,A::d

**public:**

f,b,A::c

C的可访问成员和访问权限

**private:**

g,A::a

**protected:**

h,d,A::b,A::e

**public:**

i,A::c,A::d

D的可访问成员和访问权限

**private:**

j,C::g

**protected:**

k,B::c,B::e,C::h,C::d,B::f,B::b,C::i

**public:**

n,d

三、

A a; A

B b; B

C c; AC

D d; ACBD

E e; ACBDACBDE

F f; ACBDACBDEBACBDACBDACBDEF

四、改错题

(1) ~A( ) { } 析构函数必须是public

(2) A(int x) { b=x; }; 有未被初始化的引用变量

(3) virtual ~C( ) { }; 析构函数必须是public

(4) int \*A::\*p=&c.z; c.z不在A中

(5) int i=a.b; a.b不可访问

(6) i=a; a不能转换为int

(7) i=b.b; b中没有变量b

(8) i=c.d; c.d不可访问

(9) i=b.\*p; b中没有z变量

(10) return 1; main应该没有返回值

五、执行后的值

x=3

9

2

3

14

16

5

六、写出成员函数

MAP::MAP(int n):n(n),c(0),e(new int[n][2]) //图最多n个顶点，假设图初始时无边

{

}

MAP::MAP(const MAP& m):n(m.n),c(m.c),e(new int[m.n][2]) //深拷贝构造函数

{

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

{

e[i][0] = m.e[i][0];

e[i][1] = m.e[i][1];

}

}

MAP& MAP::operator=(const MAP& m) //深拷贝赋值函数

{

if(this == &m)

return \*this;

if(e != nullptr)

delete[] e;

\*(const\_cast<int (\*\*)[2]>(&e)) = new int[m.n][2];

c = m.c;

\*(const\_cast<int \*>(&n)) = m.n;

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

{

e[i][0] = m.e[i][0];

e[i][1] = m.e[i][1];

}

return \*this;

}

MAP& MAP::operator( )(int v0, int v1) //连接顶点v0和v1成边,设v0<v1

{

e[c][0] = v0;

e[c][1] = v1;

c++;

return \*this;

}

int(\*MAP::operator[ ](int x))[2] //取图中的边x

{

return e + x;

}

MAP::~MAP( )

{

delete[] e;

}