西安电子科技大学

考试时间 120 分钟

试题

题号	一	二	三	四	总分
分数					

1. 考试形式：闭卷； 2. 本试卷共四大题，满分 100 分。

班级学号姓名任课教师

Part I There is one error in each code paragraph. Find out the error and write down the error statement on your answer sheet. (20 points)

(1)	float* ptr = new float[20];
	for (int i = 0; i < 20; i++)
	    ptr[i] = i+2;
	delete ptr;
		(2)	namespace myspace{
			    void do() { /* … */}
			    int temp;
			}
			using namespace myspace;
			using namespace myspace:temp;

(3)	class C{
	    int x;
	    void setx(int a) { /* … */}
	};
	void main() {
	    C c1;
	    c1.setx(3);
	}
		(4)	class Student {
			// …
			public:
			void Student();
			      ~ Student();
			};

(5)	class BC {
	    int x;
	public:
	    BC(int xx = 0) { x = xx; }
	};
	class DC : public BC {
	    char c;
	    DC(int x1, char c1) {
	        x = x1;
	        c = c1;
	    }
	};
		(6)	class C {
			    int sz;
			public:
			    friend C operator+ (const C&, const C&);
			    // …
			};
			C C::operator+ (const C& c1, const C& c2){
			    cout<<c1.sz;
			    // ….
			}

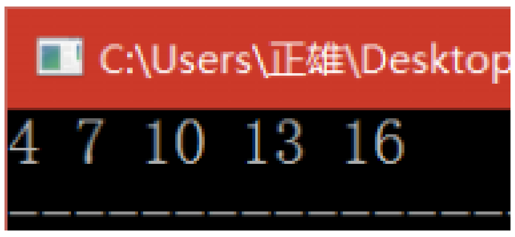
(7)	class B {
	public:
	B(int a=10, float y)
	{ i=a; z=y; }
	private:
	int i;
	float z;
	};
		(8)	class Animal {
			public:
			virtual void f()=0;
			};
			void f(){
			Animal b;
			}

(9)	class C {
	public:
	void m() {/* ··· */}
	static void s() {/* ··· */}
	};
	void main() {
	C c1;
	c1.m();
	C::m();
	c1.s();
	C::s();
	}
		(10)	template <class T, int i> class Array {
			int sz;
			public:
			Array():sz(i) { }
			// ······
			};
			void f(int x)
			{
			Array<int, x> ay;
			}

Part II Write the following programs ’ output. (30 points)

1. (6 points)

```
#include <iostream>
using namespace std;
void main() {
int i = 1;
while(i <= 15) {
i++;
if(i % 3 != 1) continue;
        else cout << i << " ";
      }
```



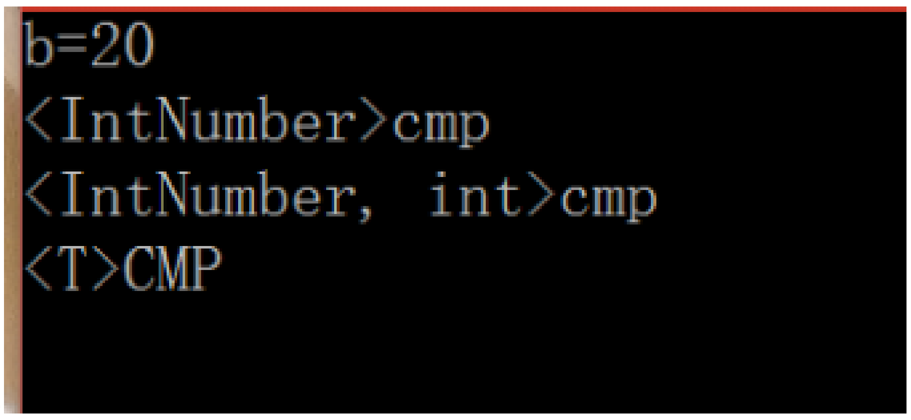
C:\Users\正雄\Desktop

```
4 7 10 13 16
---------------
```

```
}
```

2. (6 points)

```
#include <iostream>
using namespace std;
class IntNumber{
    int    value;
public:
    IntNumber(int value){this->value = value;}
    IntNumber& operator*=(int v){ value *= v; return *this;}
    friend void operator << (ostream& outf, IntNumber& n){ outf << n.value << endl;}
};
template<class T> const char* cmp(T a,    T b)    { return "<T>CMP"; }
const char* cmp(IntNumber a, IntNumber b)       { return "<IntNumber>cmp";      }
const char* cmp(IntNumber a,     int b)              { return "<IntNumber, int>cmp";      }
int main()    {
    IntNumber a(1), b(2);
    b *=10;
    cout << "b=" << b;
    cout << cmp(a, b) <<endl;
    cout << cmp(a, 0) << endl;
    cout << cmp(1, 0) << endl;
    return 0;
```



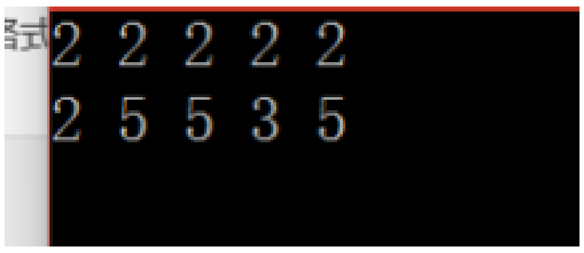
```
b=20
<IntNumber>cmp
<IntNumber,  int>cmp
<T>CMP
```

```
}
```

3. (6 points)

```
#include <iostream>
using namespace std;
void add1(int a1) {
     a1++;
}
void add2(int& a2) {
     a2++;
}
void add3(int* a3) {
     (*a3)++;
}
int main() {
     int t1=2, t2=2, *t3=&t2, t4=t2, &t5=t2;
```

```
        add1(t1) ;add2(t2); add3(t3); add3(&t4); add2(t5);
         cout<<t1<<" "<<t2<<" "<<*t3<<" "<<t4<<" "<<t5<<endl;
         return 0;
```

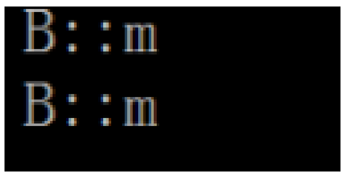


```
2 2 2 2 2
2 5 5 3 5
```

```
}
```

4. (6 points)

```
#include <iostream>
using namespace std;
class B {
public:
        void m()        {    cout << " B::m" << endl;      }
};
class D: public B {
public:
        void m()        {    cout << " D::m" << endl;      }
};
int main() {
        B *p[2];
        p[0] = new D;
        p[1] = new B;
        for(int i = 0; i <= 1; i++)
                p[i]->m();
        return 0;
```



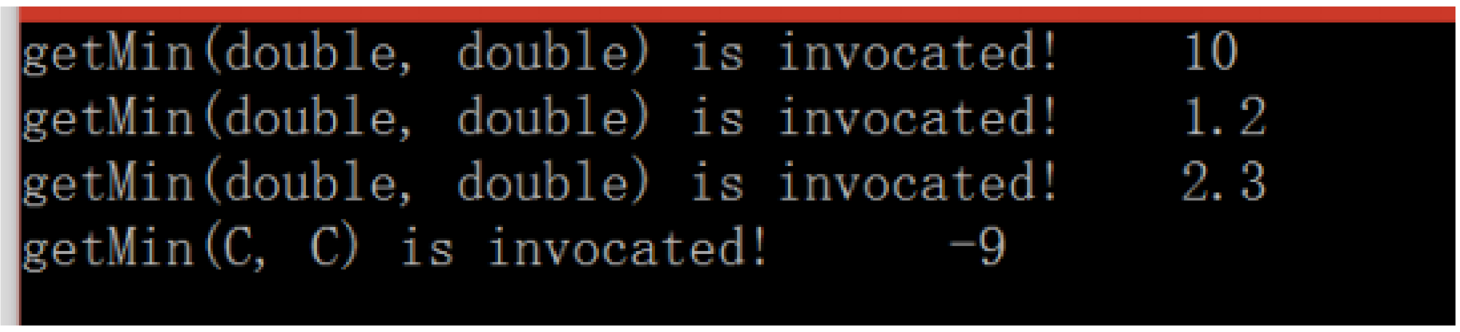
```
B::m
B::m
```

```
}
```

5. (6 points)

```
#include <iostream>
using namespace std;
class C{
        int value;
public:
        C(int v) : value(v) { }
        bool operator< ( C& b ) {
                if( value < b.value )     return true;
                return false;
        }
```

```
    int getValue( ) { return value; }
};
double getMin(double a, double b) {
    cout << "getMin(double, double) is invocated!\t";
    if( a < b ) return a;
    return b;
}
C getMin(C& a,    C& b) {
    cout << "getMin(C&, C&) is invocated!\t";
    if( a < b ) return a;
    return b;
}
C getMin(C a,    C b)    {
    cout << "getMin(C, C) is invocated!\t";
    if( a < b ) return a;
    return b;
}
int main( )    {
    C c1(-9), c2(90);
    cout << getMin(10, 99) << endl;
    cout << getMin(1.2, 2.3) << endl;
    cout << getMin(10, 2.3) << endl;
    cout << getMin(c1, 2).getValue() << endl;
    return 0;
}
```



```
getMin(double, double) is invocated!    10
getMin(double, double) is invocated!    1.2
getMin(double, double) is invocated!    2.3
getMin(C, C) is invocated!      -9
```

Part III   Object-Oriented Analyzing and Designing(30 points)

1. From following pictures, please analyze and design the class and class Hierarchies. (15 points)

游 泳
Swimming

篮 球
Basketball

足 球
Football

2. (15 points)

Define a class named **Point** which can express the position of any point in a plane coordinate(坐标) system. A **Point** object contains two private data member: **x** which holds the horizontal coordinate, **y** which holds the vertical coordinate. This class should have such public operations:

a) a default constructor that set the coordinate to (0,0);

b) a constructor that takes two integer, which initializes the x and y;

c) an overloaded operator “ += ”which moves the point to another position;

d) some member functions to re-set or get each of two data members.

Part IV Programming (20 points)

1. (10 points)

Define and implement a class **MyStrin“g** ” according to the main() and the output in comments.

```
int main()
{
   MyString   s1("0123456789"),   s2(5),   s3;
      s1.display();               // Output: [0123456789]
      s2.display();               // Output(5 spaces between []) : [     ]
      s3.display();               // Output(no space between []): []
   s3 = s1;
      s1.display();               // Output: [0123456789]
      s3.display();               // Output: [0123456789]
   s2 = s1 + 3;
      s1.display();               // Output: [0123456789]
      s2.display();               // Output: [3456789]
   s3 = ++++s2;
      s2.display();               // Output: [56789]
      s3.display();               // Output: [56789]
   return 0;
}
```

2. (10 points)

According to the main function and the output below, implement a class hierarchy with **fighter** as the base class and **Warrior** is a derived class from **fighter**.

```
int main()
{    fighter * objs[2];
     objs[0] = new fighter("Harry");
     Warrior Stallone("Stallone", objs[0]);
     objs[1] = &Stallone;
     cout << "== Test the class famliy ==" << endl;
     for(int i=0; i<2; i++) {
          objs[i] -> outTitle();
          cout << "----      " << i+1 << "       ----" << endl;
          objs[i] -> hello();
     }
     cout << "==This is the end." << endl;
     delete objs[0];
     return 0;
}
```

The output of this program is:

```
== Test the class famliy ==
We are fighters.
----    1    ----
Harry is a fighter.
We are fighters.
----    2    ----
Stallone is a stronger warrior than Harry.
==This is the end.
```