

<div><div>✕ True-or-False 10</div><div>A. Multiple-Choice - 1 10</div><div>A Fill-in-Blank 9</div><div>🔒 Fill-in-Blank - P 3</div></div>		
1-1	catch (type p) acts very much like a parameter in a function. Once the exception is caught, you can access the thrown value from this parameter in the body of a catch block.。(2分)	Author: 张德慧 Organization: 西安邮电大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-1	Accepted (2 point(s))	
1-2	Insertter << can be used to output all kinds of primitive types, including the pointers. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-2	Accepted (2 point(s))	
1-3	The reason inline functions are introduced into the C++ is to reduce the complecity of space, i.e. to shorten the code. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input type="radio"/> T</div><div><input checked="" type="radio"/> F</div></div>	
1-3	Accepted (2 point(s))	
1-4	In C++, only existing operators can be overloaded. (2分)	Author: 张德慧 Organization: 西安邮电大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-4	Accepted (2 point(s))	
1-5	If you are not interested in the contents of an exception object, the catch block parameter may be omitted.。(2分)	Author: 张德慧 Organization: 西安邮电大学
	<div><div><input type="radio"/> T</div><div><input checked="" type="radio"/> F</div></div>	
1-5	Wrong Answer (0 point(s))	
1-6	It is possible to access any item in a vector directly via its index. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-6	Accepted (2 point(s))	
1-7	Functions with the same name can be identified via namespaces. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-7	Accepted (2 point(s))	
1-8	To make functions overloaded, the parameter list of the functions have to be different from each other. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input type="radio"/> T</div><div><input checked="" type="radio"/> F</div></div>	
1-8	Wrong Answer (0 point(s))	
1-9	Constructors are able to be declared as virtual. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input type="radio"/> T</div><div><input checked="" type="radio"/> F</div></div>	
1-9	Accepted (2 point(s))	
1-10	Manipulators are objects to be inserted or extracted into/from streams. (2分)	Author: 翁恺 Organization: 浙江大学
	<div><div><input checked="" type="radio"/> T</div><div><input type="radio"/> F</div></div>	
1-10	Accepted (2 point(s))	

2-1 Given code below:

```
vector<int> v;
for ( int i=0; i<4; i++ ) {
    v.push_back(i+1);
}
cout << v.size();
```

The output should be: (2分)

☐ A. 1

☐ B. 2

☐ C. 3

☒ D. 4

Author: 翁恺

Organization: 浙江大学

2-1 Accepted (2 point(s))

2-2 About virtual function, which statement below is correct? (2分)

☐ A. Virtual function is a static member function

☐ B. Virtual function is not a member function

☒ C. Once defined as virtual, it is still virtual in derived class without virtual keyword,.

☐ D. Virtual function can not be overloaded.

Author: 翁恺

Organization: 浙江大学

2-2 Accepted (2 point(s))

2-3 It is better to choose \_\_\_\_ when the function is not complecated and is to be called frequently. (2分)

☐ A. overloaded function

☒ B. inline function

☐ C. recuisive function

☐ D. embedded function

Author: 翁恺

Organization: 浙江大学

2-3 Accepted (2 point(s))

2-4 Suppose that statement3 throws an exception of type Exception3 in the following statement: (2分)

```
try {
statement1; statement2; statement3; }
catch (Exception1 ex1) {}
catch (Exception2 ex2) {}
catch (Exception3 ex3) { statement4; throw; }
statement5;
```

Which statements are executed after statement3 is executed?

☐ A. statement2

☐ B. statement3

☒ C. statement4

☐ D. statement5

Author: 张德慧

Organization: 西安邮电大学

2-4 Accepted (2 point(s))

2-5 What is wrong in the following code?

```
vector v; v[0] = 2.5;
```

(2分)

☒ A. The program has a compile error because there are no elements in the vector.

☐ B. The program has a compile error because you cannot assign a double value to v[0].

☐ C. The program has a runtime error because there are no elements in the vector.

☐ D. The program has a runtime error because you cannot assign a double value to v[0].

Author: 张德慧

Organization: 西安邮电大学

2-5 Wrong Answer (0 point(s))

2-6 Given:

```
template <class T>
void max(T a, T b, T &c)
{
    c = a+b;
}
```

Which code fragement below is correct? (2分)

☐ A. `int x,y;char z;max(x,y,z);`

☒ B. `double x,y;double z;max(x,y,z);`

☐ C. `int x,y;float z;max(x,y,z);`

☐ D. `float x,y;double z;max(x,y,z);`

Author: 翁恺

Organization: 浙江大学

2-6 Accepted (2 point(s))

2-7 About `const` data member, which statement below is correct? (2分)

☐ A. `const` member can be defined without any initialization, and can not be modified.

☒ B. `const` member has to be initialized, and can not be modified.

☐ C. `const` member can be defined without any initialization, and can be modified later.

☐ D. `const` member has to be initialized, and can be modified later.

Author: 翁恺

Organization: 浙江大学

2-7 Accepted (2 point(s))

2-8 Which operator below can not be overloaded? (2分)

☐ A. `&&`

☐ B. `[]`

☒ C. `::`

☐ D. `<<`

Author: 翁恺

Organization: 浙江大学

2-8 Accepted (2 point(s))

2-9 Which one below can NOT be overloaded? (2分)

☐ A. member function

☐ B. free function (global function)

☒ C. destructor

☐ D. constructor

Author: 翁恺

Organization: 浙江大学

2-9 Accepted (2 point(s))

2-10 About `delete` operator, which statement below is NOT correct? (2分)

☐ A. Only pointers as the result of a `new` opertion can be used to be `delete` d.

☐ B. Destructor will be called automatically during the `delete` operation.

☒ C. It is safe to `delete` the same pointer multiple times.

☐ D. There's only one pair of `[]` followed to `delete` a multi-dimension array.

Author: 翁恺

Organization: 浙江大学

2-10 Accepted (2 point(s))

4-1 The output of the code below is:

```
#include<iostream>
using namespace std;
class MyClass {
public:
    MyClass(int x): val(x) {}
    void Print() const {cout << 1 << val;}
    void Print() {cout << 2 << val;}
private:
    int val;
};
int main() {
    const MyClass obj1(10);
    MyClass obj2(20);
    obj1.Print();
    obj2.Print();
    return 0;
}
```

Author: 翁恺

Organization: 浙江大学

110220 (3分)

4-1 Accepted (3 point(s))

4-2 The output of the code below is:

```
#include<iostream>
using namespace std;
class AA {
public:
    AA() { cout << 1; }
    ~AA() { cout << 2; }
};
class BB: public AA {
    AA aa;
public:
    BB() { cout << 3; }
    ~BB() { cout << 4; }
};
int main() {
    BB bb;
    return 0;
}
```

Author: 翁恺

Organization: 浙江大学

113422 (3分)

4-2 Accepted (3 point(s))

4-3 The output of the code below is:

```
#include <iostream>
using namespace std;

class A {
public:
    A() { cout << 1; }
} a;

int main()
{
    cout << 2;
    A a;

    return 0;
}
```

Author: 翁恺

Organization: 浙江大学

121 (3分)

4-3 Accepted (3 point(s))

4-4 write the output of the code below.

```
#include<iostream>
using namespace std;
```

🏆 Author: hulanqing

Organization: 浙江大学

```

class INCREMENT
{
public:
    INCREMENT( int v = 0, int i = 1 );
    void addIncrement()
    {
        v += increment;
    }
    void print() const;
    int get() const
    {
        return v;
    }
private:
    int v;
    const int increment;
};

INCREMENT::INCREMENT( int v, int i ) : v( v ), increment( i )
{
}

void INCREMENT::print() const
{
    cout << v << endl;
}

int main()
{
    INCREMENT value( 1, 2);
    value.print();

    for ( int j = 1; j <= 2; j++ )
    {
        value.addIncrement();
        value.print();
    }
    return 0;
}

```

One for each line:

line 1:  (1分) line 2:  (1分)  
line 3:  (1分)


4-4 Accepted (3 point(s))

4-5 write the output of the code below.

```

#include<iostream>
using namespace std;
class TEST
{
    int num;
public:
    TEST( int num=0);
    void increment( ) ;
    ~TEST( );
};
TEST::TEST(int num) : num(num)
{
    cout << num << endl;
}
void TEST::increment()
{
    num++;
}
TEST::~~TEST( )
{
    cout << num << endl;
}
int main( )
{
    TEST array[2];
    array[0].increment();
    array[1].increment();
    return 0;
}

```

 Author: hulanqing  
Organization: 浙江大学



One for each line:

line 1:	0	(1分)
line 2:	0	(1分)
line 3:	1	(1分)
line 4:	1	(1分)

4-5 Accepted (4 point(s))

4-6 The output of the code below is:

Author: 翁恺  
Organization: 浙江大学


```
#include <iostream>
using namespace std;

class MyClass {
public:
    MyClass() {
        ++count;
    }
    ~MyClass() {
        --count;
    }
    static int getCount() {
        return count;
    }
private:
    static int count;
};
int MyClass::count = 0;
int main() {
    MyClass obj;
    cout << obj.getCount();
    MyClass obj2;
    cout << MyClass::getCount();
    cout << obj2.getCount();
    return 0;
}
```

122 (3分)

4-6 Accepted (3 point(s))

4-7 write the output of the code below.

 Author: hulanqing  
Organization: 浙江大学

```
#include<iostream>
using namespace std;

enum NOTE { middleC, Csharp, Cflat };
class Instrument {
public:
    virtual void play(NOTE) const = 0;
    virtual char* what() const = 0;
    virtual void adjust(int) = 0;
};

class Wind : public Instrument {
public:
    void play(NOTE) const {
        cout << 1 << endl;
    }
    char* what() const { return "Wind"; }
    void adjust(int) {}
};

class Percussion : public Instrument {
public:
    void play(NOTE) const {
        cout << 2 << endl;
    }
    char* what() const { return "Percussion"; }
    void adjust(int) {}
};

class Stringed : public Instrument {
public:
    void play(NOTE) const {
        cout << 3 << endl;
    }
    char* what() const { return "Stringed"; }
    void adjust(int) {}
};
```

```

void adjust(int) {}

};

class Brass : public Wind {
public:
    void play(NOTE) const {
        cout << 11 << endl;
    }
    char* what() const { return "Brass"; }
};

class Woodwind : public Wind {
public:
    void play(NOTE) const {
        cout << 12 << endl;
    }
    char* what() const { return "Woodwind"; }
};

void tune(Instrument& i) {
    i.play(middleC);
}

void f(Instrument& i) { i.adjust(1); }

int main() {
    Wind flute;
    Percussion drum;
    Stringed violin;
    Brass flugelhorn;
    Woodwind recorder;
    tune(flute);
    tune(drum);
    tune(violin);
    tune(flugelhorn);
    tune(recorder);
    f(flugelhorn);
    return 0;
}

```

One for each line:

line 1:	1	(1分)
line 2:	2	(1分)
line 3:	3	(1分)
line 4:	11	(1分)
line 5:	12	(1分)

4-7 Accepted (5 point(s))

4-8 write the output of the code below.

```


#include<iostream>
#include<string>
using namespace std;

class Pet {
public:
    virtual string speak() const { return "pet!"; }
};

class Dog : public Pet {
public:
    string speak() const { return "dog!"; }
};

int main() {
    Dog ralph;
    Pet* p1 = &ralph;
    Pet& p2 = ralph;
    Pet p3;
    cout << p1->speak() <<endl;
    cout << p2.speak() << endl;
    cout << p3.speak() << endl;
    return 0;
}

```

 Author: hulanqing  
Organization: 浙江大学

dog! (1分)

dog! (1分)

pet! (1分)

4-8 Accepted (3 point(s))

4-9 The output of the code below is:

Author: 翁恺  
Organization: 浙江大学

```
#include <iostream>
using namespace std;

class A {
    int i;
public:
    A() : i(0) {}
    ~A() { cout << get(); }
    void set(int i) { this->i = i; }
    int get() { return i; }
};

int main()
{
    A* p = new A[2];
    delete p;
    return 0;
}
```

0 (3分)

4-9 Accepted (3 point(s))

5-1 The function template printArrayInfo() computes the minimal, maximal and average value of a two dimension array and prints them out, where nrows is number of rows and ncols is the number of columns.

Author: hulanqing  
Organization: 浙江大学  
Time Limit: 400 ms  
Memory Limit: 64 MB

```
#include <iostream>

template<class T> (1分)
void printArrayInfo(T* (1分) array, int nrows, int ncols)
{
    T (1分) max = array[0], min = array[0];
    double avg = 0 (1分);
    for(int i = 0; i < nrows; ++i)
    {
        for(int j = 0; j < ncols; ++j)
        {
            T val (1分) = array[i*ncols+j] (1分);
            if(val<min (1分)) min = val;
            if(val>max (1分)) max = val;
            avg =avg+static_cast<double>(v (1分);
        }
    }
    avg /= (nrows*ncols (1分));
    std::cout << "min=" << min << std::endl;
    std::cout << "max=" << max << std::endl;
    std::cout << "avg=" << avg << std::endl;
}

int main()
{
    int ai[2][3]={8,10,2},{14,4,6}};
    printArrayInfo(ai[0], 2, 3);
    double af[1][5]={3.4f,4.2f,6.6f,2.4f,-0.9f}};
    printArrayInfo(af[0], 1, 5);
    return 0;
}
```

5-1 Accepted (10 point(s))

5-2 The class String is a simple C++ encapsulation of the C character arrays.

Author: hulanqing  
Organization: 浙江大学  
Time Limit: 400 ms  
Memory Limit: 64 MB

```
#include <cstring>
#include <iostream>
#include <stdexcept>

class StringIndexError : public std::out_of_range {
private:
    int index;
public:
    StringIndexError(int idx) : std::out_of_range(""), index(idx) {}
    int getIndex() const
    {
        return index;
    }
};

class String {
private:
    char *m_ptr;
public:
    String(const char *ptr)
    {
        m_ptr = new char[strlen(ptr)+1] (1分);
        strcpy(m_ptr, ptr);
    }
    ~String()
    {
        delete[] m_ptr (1分);
    }
    String &operator+=(const String &str)
    {
        char *s = new char[strlen(m_ptr)+strlen(str.m_ptr)+1] (1分);
```



```

        if (m_ptr)
        {
            strcpy(s, m_ptr);
            delete[] (1分) m_ptr;
        }
        strcat(s, str.m_ptr); // appends str.m_ptr to s
        m_ptr (1分) = s;
        return *this (1分);
    }
    bool operator==(const String &str) const
    {
        return (strcmp(m_ptr, str.m_ptr) == 0);
    }
    char& operator[](int i)
    {
        if (i >= 0 && i < strlen(m_ptr)) return m_ptr[i];
        throw StringIndexError(i);
    }
    friend (1分) std::ostream& operator<<(std::ostream &, const String &);
};

std::ostream& (1分) operator<<(std::ostream &out, const String &str)
{
    return out << str.m_ptr;
}

int main()
{
    String s1("Hello "), s2("world!");

    if (s1 == s2)
        std::cout << "S1==S2" << std::endl;
    else
        std::cout << "S1!=S2" << std::endl;

    s1 += s2;
    std::cout << s1 << std::endl;

    try (1分) {
        int k = 0;
        while (true)
            std::cout << s1[k++];
    }
    catch (1分) (const StringIndexError& ex) {
        std::cout << "\nString index is out of range: " << ex.getIndex() << std::endl;
    }

    return 0;
}

```

5-2 Accepted (10 point(s))

5-3 The class Queue implements a circular queue data structure.

```

#include <iostream>

template<class T>
class Queue {
private:
    int capacity;          // capacity of the queue
    T* (1分) data;          // dynamically allocated array of doubles
    int front;             // head of the queue
    int rear;              // tail of the queue
public:
    Queue(int maxsize);
    ~Queue();
    bool empty();
    bool full();
    void push(T a);        // append a double value to the tail of queue
    T pop();               // delete the head element of the queue
};

template<class T> Queue<T>::Queue(int maxsize)
{
    capacity = maxsize;
    data = new T[maxsize] (1分);
}

```

Author: hulanqing  
 Organization: 浙江大学  
 Time Limit: 400 ms  
 Memory Limit: 64 MB

```

front = rear = 0;
std::cout << "queue initialized! ";
}
template<class T> Queue<T>::~~Queue()
{
    delete[] data (1分);
    std::cout << "queue destroyed! ";
}
template<class T> bool Queue<T>::empty()
{
    return (front == rear) (1分);
}
template<class T> bool Queue<T>::full()
{
    return (front == ((rear+1)%capacity)) (1分);
}

//The dynamic array data will be a circular Queue
template<class T> void Queue<T>::push(T a)
{
    if (full())
    {
        exit(0);
    }
    else
    {
        data[rear] (1分) = a;
        rear = (rear+1)%capacity (1分);
    }
}

template<class T> T Queue<T>::pop()
{
    if (empty())
    {
        exit(0);
    }
    T top=data[front] (1分);
    front = (front+1)%capacity (1分);
    return top;
}

int main()
{
    Queue<double> (1分) q(5);
    std::cout << q.empty();

    q.push(1.3);
    q.push(2.3);
    q.push(3.3);
    q.push(4.3);
    std::cout << q.full();

    q.pop();
    q.pop();
    q.pop();
    q.push(5.3);
    q.push(6.3);
    q.push(7.3);
    std::cout << q.full();

    q.pop();
    q.pop();
    q.pop();
    q.pop();
    std::cout << q.empty();
    return 0;
}

```