中 山 大 学 软 件 学 院 2010 级 软 件 工 程 专 业 （ 2010 学年度第 2 学期）

《程序设计 （II）》期末试题 （A卷，附参考答案）

（考 试 形 式 ： 闭 卷　　考 试 时 间 ： 2 小 时）

警 示

《中山大学授予学士学位工作细则》第六条

考 试 作 弊 不 授 予 学 士 学 位

方向：__________________　　姓名：_______________　　学号：________________

Section I:　Multiple Choice (20 points)

For each of the following questions, choose ONE and only ONE of the provided multiple choices: A, B, C and D, corresponding to the best answer for them.

1. (2 points)　What are the outputs of the execution of the following code?

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

void foo(int& x) {
    cout << x++;
};

int main() {
    int c = 0;

    foo(c);
    foo(c);
    return 0;
}
```

(A) 11　　　(B) 10　　　(C) 00　　　(D) 01

Solution:　(D)

2. (2 points)　Given the code “`int a = 3;  int* p = &a;`”, what is the value of *p?

(A) The address of a　　　(B) The value of p　　　(C) 3　　　(D) None of the above

Solution:　(C)

3. (2 points)　Which of the following statement is NOT true?

(A) A reference is the entire object, while a pointer is only the address of it.

(B) A pointer can point to NULL, while a reference can never point to NULL.

(C) Pointers can be assigned NULL directly, while a reference cannot be assigned NULL and must be assigned at initialization.

(D) A pointer is a variable that holds a memory address, while a reference has the same memory address as the item it references.

Solution: (A)

4. (2 points) If two functions within the same class have the same name, what happens?

(A) Either one will be called depending on the parameters.

(B) Neither function will be called.

(C) This is dead code and a very difficult problem to debug.

(D) Both functions will be called in order. This is named multiple inheritance.

Solution: (A)

5. (2 points) Which of the following statement about static members is NOT true?

(A) A static member is declared in class.

(B) A static member function cannot access any non-static member variable.

(C) A static member does not have `this` pointer.

(D) A static member must be initialized in a constructor.

Solution: (D)

6. (2 points) Assume that in the run-time environment, an instance of `int` type occupies 4 bytes. Given the class `Min2Max` declared as following, the value of `sizeof(Min2Max)` is ______ .

(A) 4 (B) 8 (C) 12 (D) 16

```
class Min2Max {
public:
    int getMax();
    void setMax(int m);
    int getMin();
    void setMin(int m);
private:
    static int std;
    int max, min;
};
```

Solution: (B)

7. (2 points) What is wrong with the following abstract class definition?

(A) There are no pure virtual function. (B) There is a non-virtual function.

(C) Private members are being accessed by a public function. (D) Nothing.

```
class Shape {
public:
    virtual double print() const;
    double area() const {
       return base * height;
    }
private:
   float base, height;
};
```

Solution: (A)

8. (2 points) Which is true?

(A) A derived class may have more than one base class.

(B) A base class may have more than one derived class.

(C) Both (a) and (b)

(D) Neither (a) nor (b)

Solution: (C)

9. (2 points) What is the primary value in using virtual functions within C++?

(A) They prevent the base class function from ever being called.

(B) They allow you to provide unique behavior for derived class.

(C) Since they are "virtual" they sometimes don't really exist.

(D) None of the above.

Solution: (B)

10. (2 points) If an object-oriented programming language supports multiple inheritance of classes, it must take repeated inheritance into consideration. The C++ programming language makes use of ________ to resolve the ambiguous references to data members in a derived class, and ensure the derived class keeping a solely copy of members from repeatedly inherited ancestors.

(A) polymorphism	(B) virtual members
(C) virtual functions	(D) virtual base classes

Solution: (D)

Section II: Short Answer (40 points)

Briefly answer the questions according the requirements.

1. (5 points) Point out the syntax errors in the following C++ source code:

```
int getValue(int index);
double getValue(int index);
int getValue(int& index);

int main() {
   int i = 0;

   getValue(i);
   return 0;
}
```

Solution:

(1) Function declarations at line 2 and line 3: overloaded function differs only by return type. (3 points)

(2) Function call at line 8: ambiguous call to overloaded function. (2 points; note that different types of parameter passing cannot be used to distinguish two overloaded functions)

2. (5 points) Fill in the blank in the following C++ source code:

```
class Complex {
    double real, imag;
public:
    Complex(double real = 0., double imag = 0.);    // constructor
    ____ operator+ ______  ;    // overloading operator+()
};

// define overloaded '+'(plus) operator for Complex
______ Complex::operator+ _____________ {
    ___________________________
}
```

Solution:

(1) Function interface (prototype): 3 points.

(2) Function implementation (body): 2 points.

```
Complex Complex::operator+(const Complex& c) const {
    Complex result;

    result.real = this->real + c.real;
    result.imag = this->imag + c.imag;
    return result;
}
```

3. (5 points) Implement the copy constructor and assignment operator for the following class.

```
class Array {
public:
    Array(int arraySize): data(0), size(arraySize) {
        if (size > 0)  data = new T[size];
    }
    ~Array() {
        if (data)  delete[] data;
    }
    // ...
private:
    T* data;
    int size;
};
```

Solution:

(1) Function interface (prototype): 1 points.

(2) Implementation of copy constructor: 2 points.

(3) Implementation of assignment operator: 2 points.

```
Array(const Array& another): data(0), size(another.size) {
    if (size > 0) {
        data = new T[size];
        for (int i = 0; i < size; ++i)  data[i] = another.data[i];
    }
}

const Array& operator=(const Array& another) {
    if (data == another.data)  return *this;
    if (data) {
        delete[] data;
        data = 0;
    }
    size = another.size;
    if (size) {
        data = new T[size];
        for (int i = 0; i < size; ++i)  data[i] = another.data[i];
    }
    return *this;
}
```

4. (5 points) What are the major problems in the following code?

```
1   class Counter {
2       int count;
3   public:
4       Counter() { count = 0; }
5       void increment() { count++; }
6       int& getVal() { return count; }
7   };
8
9   ostream& operator<<(ostream& os, Counter& counter) {
10      os << counter.count;
11  }
```

Solution:

(1) Line 6: it is not safe to return the reference of a private member. (2 points)

(2) Line 9: `operator<<()` should be declared as friend function. (3 points)

5. (5 points) Explain the difference between "vector" and "array"?

Solution:

(1) Vector is a template class in STL and array is native in C++ language. (2 points)

(2) The length of a vector can be dynamically increased, while an array has a fixed length. (3 points)

6. (5 points) Declare a class `Account` to represent a bank account. The following members are included in the class:

- Data members: name of the depositor, account number, type of account, and balance amount in the account.
- Member functions: to assign initial values, to deposit an amount, to withdraw an amount after checking the balance, and to display the name and balance.

The implementation of member functions (function bodies) are not required.

Solution:

(1) Data members: 2 points.

(2) Function members: 3 points.

```
class Account {
    string depositor;
    string accNumber;
    int accType;
    double balance;
public:
     Account(double init);
    bool deposit(double amount);
    bool withdraw(double amount);
    void display();
}
```

7. (5 points) Define a friend function to compute the distance between two `Point` objects.

```
class Point {
    float x, y;
public:
    Point(float x, float y) { this.x = x; this.y = y; }
};
```

Solution:

(1) Add the following in Point class declaration (2 points):

```
friend float distance(const Point& p, const Point& q);
```

(2) The following is function implementation (3 points):

```
float distance(const Point& p, const Point& q) {
    return sqrt((p.x – q.x) * (p.x - q.x) + (p.y - q.y) * (p.y - q.y));
}
```

8. (5 points) What are the functions which compiler implements for us if we do not define one ?

Solution:

(1) Default constructor (2 points)

(2) Copy constructor (2 points)

(3) Assignment operator (1 points)

(4) Default destructor (0 points)

(5) Address operator (0 points)

Section III: Program Output Analysis (20 points)

Write the result after executing the following programs or program fragments.

1. (5 points) What are the outputs of the execution of the following C++ program:

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

class Window {
public:
    Window() {
        count = count + 1;
    }
    ~Window() {
        count = count - 1;
    }
    int countWindows() {
        return count;
    }
private:
    static int count;
```

```cpp
};

int Window::count = 0;

class WorkWindow: public Window {
public:
    WorkWindow() {
        cout << "Open a work window" << endl;
    }
    ~WorkWindow() {
        cout << "Close a work window" << endl;
    }
};

class MessageWindow: public Window {
public:
    MessageWindow() {
        cout << "Open a message window" << endl;
    }
    ~MessageWindow() {
        cout << "Close a message window" << endl;
    }
};

class Screen {
public:
    Screen(): msgWin(), workWin() {
        cout << "Initialize the screen" << endl;
    }
    ~Screen() {
        cout << "Clear the screen" << endl;
    }
    int countWindows() {
        return workWin.countWindows();
    }
private:
    WorkWindow workWin;
    MessageWindow msgWin;
};

int main() {
    Screen screen;
    MessageWindow mwin;

    cout << screen.countWindows() << " window(s). " << endl;
    cout << mwin.countWindows() << " message window(s)." << endl;
    return 0;
}
```

Solution: 3 points for correct count number (line 5 and 6).

```
1    Open a work window
2    Open a message window
3    Initialize the screen
4    Open a message window
5    3 window(s).
6    3 message window(s).
7    Close a message window
8    Clear the screen
9    Close a message window
```

10	Close a work window

2. (5 points) What are the outputs of the execution of the following C++ program:

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

const int MAX_LEN = 80;

class Employee {
public:
    Employee(char* n, int s): salary(s) {
        strncpy(name, n, MAX_LEN);
        name[MAX_LEN] = '\0';
        cout << "Constructing Employee " << name << endl;
    }
    ~Employee() {
        cout << "Destructing Employee " << name << endl;
    }
    virtual void show() {
        cout << "Salary for Employee " << name;
        cout << " is " << salary << " yuan." << endl;
    }
protected:
    char name[MAX_LEN + 1];
    int salary;
};

class Faculty: public Employee {
public:
    Faculty(char* n, int s): Employee(n, s) {
        cout << "Constructing Faculty " << name << endl;
    }
    ~Faculty() {
        cout << "Destructing Faculty " << name << endl;
    }
    virtual void show() {
        cout << "Salary for Faculty " << name;
        cout << " is " << salary << " yuan." << endl;
    }
};

class Professor: public Faculty {
public:
    Professor(char* n, int s): Faculty(n, s) {
        cout << "Constructing Professor " << name << endl;
    }
    ~Professor() {
        cout << "Destructing Professor " << name << endl;
    }
    virtual void show() {
        cout << "Salary for Professor " << name;
        cout << " is " << salary << " yuan." << endl;
    }
};

int main() {
    Employee empl("Zhang3", 3500);
    Faculty facl("Li4", 5000);
    Professor prof("Wang5", 8500);
    Employee* group[3];

    group[0] = &empl;
    group[1] = &facl;
    group[2] = &prof;
    for (int i = 0; i < 3; i++)  group[i]->show();
    return 0;
}
```

Solution:

```
Constructing Employee Zhang3
Constructing Employee Li4
Constructing Faculty Li4
Constructing Employee Wang5
Constructing Faculty Wang5
Constructing Professor Wang5
Salary for Employee Zhang3 is 3500 yuan.
Salary for Faculty Li4 is 5000 yuan.
Salary for Professor Wang5 is 8500 yuan.
Destructing Professor Wang5
Destructing Faculty Wang5
Destructing Employee Wang5
Destructing Faculty Li4
Destructing Employee Li4
Destructing Employee Zhang3
```

3. (5 points) What are the outputs of the execution of the following C++ program:

```cpp
class A {
    int val;
    static int sVal;
public:
    A(int val) {
        this->val = val;
        sVal++;
    }
    static int incr() { return sVal++; }
    void operator++() { val++; }
    int getVal() { return val; }
};

int A::sVal = 1;

int main() {
    A a1(0);
    A a2(5);

    a1++;
    cout << A::incr() << " " << a1.getVal() << endl;
    cout << A::incr() << " " << a2.getVal() << endl;
}
```

Solution:

```
3 1
4 5
```

4. (5 points) What are the outputs of the execution of the following C++ program:

```
class Base {
public:
    int bVal;
    Base() { bVal = 0; }
};

class Deri: public Base {
public:
    int dVal;
    Deri() { dVal = 1; }
};

void foo(Base *arr, int size) {
    for (int i = 0; i < size; i++, arr++)  cout << arr->bVal;
    cout << endl;
}

int main() {
    Base baseArr[5];
    Deri deriArr[5];

    foo(baseArr, 5);
    foo(deriArr, 5);
    return 0;
}
```

Solution:

This is rather tricky. When `foo()` is called, it treats `deriArr` as a Base object array, and thus when `arr++` is met, the pointer is incremented by `sizeof(int)` bytes for `bVal`.

(1) Output line 1: 2 points.

(2) Output line 2: 3 points.

```
00000
01010
```

Section IV: Programming (20 points)

Write C++ source code to satisfy requirements.

1. (15 points) The class **`IntegerSet`** defines a set of integers. The elements of the set can be organized by a linked list or an array. **`IntegerSet`** provides some public services to other classes, including:

- one function member **`hasElement()`** to judge whether an element is in the set.
- three function members overloading operator "+", "*" and "=" to perform union, intersection and assignment operations on sets.
- two friend functions overloading operator "<<" and ">>" to insert to and abstract from input and output streams of the iostream library.

The following source code demonstrate the usage of the class **`IntegerSet`**. You are asked to write the class interface of **`IntegerSet`**. You do not need to write the implementation (function bodies) of the class.

```
#include <iostream>
```

```
using namespace std;

int main() {
    IntegerSet s1, s2, s3;
    IntegerSet* ptr;

    cin >> s1 >> s2 >> s3;
    s1 = s1 + s2 * s3;
    ptr = new IntegerSet(s1);
    cout << *ptr * s2;
    delete ptr;
    return 0;
}
```

Solution: 5 points for constructors and destructors; 1 point for hasElement(); 5 points for overloading operator members; 4 points for friend operators. Data members of the class can be ignored.

```
class IntegerSet {
public:
    IntegerSet();
    IntegerSet(const IntegerSet& another);
    ~IntegerSet();

    bool hasElement(int elem);
    IntegerSet operator+(const IntegerSet& another);
    IntegerSet operator*(const IntegerSet& another);
    IntegerSet operator=(const IntegerSet& another);

    friend ostream& operator<<(ostream& stream, IntegerSet set);
    friend istream& operator>>(istream& stream, IntegerSet& set);
protected:
    struct Node {
        int element;
        Node* link;
    };
    Node* first;
};
```

2. (5 points) Change the aforementioned class **IntegerSet** to a generic class **MySet**, that is a template class with a type parameter indicating the type of elements in the set. **MySet** can be instantiated as a set of characters or a set of floats. You are asked to write the class interface of **MySet**. You do not need to write the implementation (function bodies) of the class.

Solution:

```
template <class ELEMENT>
class MySet {
public:
    MySet();
    MySet(const MySet& another);
    ~MySet();

    bool has_element(ELEMENT elem);
    MySet operator+(const MySet& another);
    MySet operator*(const MySet& another);
    MySet operator=(const MySet& another);

    friend ostream& operator<<(ostream& stream, MySet set);
    friend istream& operator>>(istream& stream, MySet& set);
protected:
    struct Node {
        ELEMENT element;
        Node* link;
    };
    Node* first;
};
```