2023年计算机学院面向对象技术与方法期末试卷A卷

班级________　学号________　姓名________　成绩________

（注意：把所有答案写在答题纸上；交卷时把试卷与答题纸分开交。试卷反面可以做草稿）

Part I. (30 points) Choose A, B, C or D and write your choices on the Answer Sheet.

1. ____ is the copy-constructor interface of class Tool

   (A) Tool (const Tool & S);　　(B) Tool (const Tool * S);

   (C) Tool (const Tool S);　　(D) None of the above

2. ______ is used to implement dynamic polymorphism in running time

   (A) static functions　(B) virtual functions　(C) const functions　(D) overloaded functions

3. If there are "int g[]={0, 2, -3, 5, -5}", the value of "g - &g[3]" is ________.

   (A) -5　(B) -3　(C) 5　(D) 3

4. The output is ______.

```
#include <iostream>
using namespace std;
class Tool {
public:
     Tool (char ch ) { cout << ch; }
     void fun() { static Tool   obj('3'); }
};
Tool   obj('2');
void main() {
     Tool   obj('1');
     obj.fun();
     obj.fun();
}
```

(A) 321

(B) 3213

(C) 2133

(D) 213

5. ______ is correct.

   (A)　const int *p;　p = new int(8);　　(B)　int * const p;　p = new int(8);

   (C)　const int &p;　p = new int(8);　　(D)　int & const p;　p = new int(8);

6. ________ can be used to create an object of template Tsam.

   (A)　Tsam ( )　obj;　　(B)　Tsam(double)　obj;

   (C)　Tsam < >　obj;　　(D)　Tsam < double >　obj;

7. _______ is NOT correct.

   (A) class Sam { public: virtual void fun(int a)=0; };　Sam　Obj;

   (B) class Sam { public: virtual void fun(int a)=0; };　Sam * Obj;

   (C) class Sam { public: virtual void fun(int a) { } };　Sam　Obj;

   (D) class Sam { public: virtual void fun(int a) {} };　Sam * Obj;

稿）

8. If operator >> has been overloaded as a friend of class Sam, and obj is an object of Sam, " cin >> obj " is compiled as ______.

(A) cin.operator >> (obj) (B) operator >> (cin, obj) (C) obj.operator >> (cin) (D) operator>> (obj, cin)

9. ______ can NOT be a member variable of class Sam.

(A) Sam * p; (B) Sam a; (C) Sam & r; (D) char * s;

10. The output is ______.

```
#include <iostream>
using namespace std;
class Sam
{ public:
   Sam( ) { cout << 'A'; }
   Sam(const Sam&) { cout << 'B'; };
   Sam& operator=(const Sam&) { cout << 'C'; }
};
void main( ) { Sam S1[3],   S2,   *S3,   &S4 = S2; }
```

(A) AAAC

(B) AAABC

(C) AAABC

(D) AAAA

11. Among the following function definitions, ______ is NOT correct.

(A) double& Fun1(double& a) { double &r = a; return r; }

(B) double& Fun2(double& a) { a *= 5; return a; }

(C) double& Fun4(double a) { a *= 5; return a; }

(D) double Fun3(double a) { double &r = a; return r; }

12. After executing the following codes, ______ is right.

```
#include <iostream>
using namespace std;
class A
{ public:   void print() {cout << "A:: print, " ; }
};
class B: public A
{public:   void print() {cout <<"B:: print, " ;}
};
```

```
void tune( A& x ) { x.print(); }
void main()   {
      B flute1;
      tune(flute1);
      A flute2;
      tune(flute2);
   }
```

(A)B::print, A::print, (B)B::print, B::print, (C)A::print, A::print, (D)A::print, B:: print,

13. ______ is right in the following codes.

```
class Base {
  public:      void print() { }
  protected:   short num;
  private:     string name;
};
class Derived : public Base
{public:
     void meeting(int n)
     {   num = n;          // (A)
```

```
          name = "John";   // (B)
     }
};
void main()
{
     Derived   D;
     D.name = "Joan";        // (C)
     D.num = 3;              // (D)
}
```

14. Among the following functions, ______ is right.

```
class Student
{
        int id;
public:
        void read() { cout   << id; }
        void get() const { read(); cout << id; }  ①
        void set(int a) const { id = a; cout << id;    } ②
        friend void show(const Student& s) const { cout << s.id; }  ③
};
```

(A) ①　　(B) ②　　(C) ③　　(D) None

15. After executing the following codes, ______ is the right.

```
#include <iostream>
using namespace std;
class Base{
private:   int a;
public:   Base(int x = 0) : a(x) { }
          ~ Base () { cout << a << ", "; } };
```

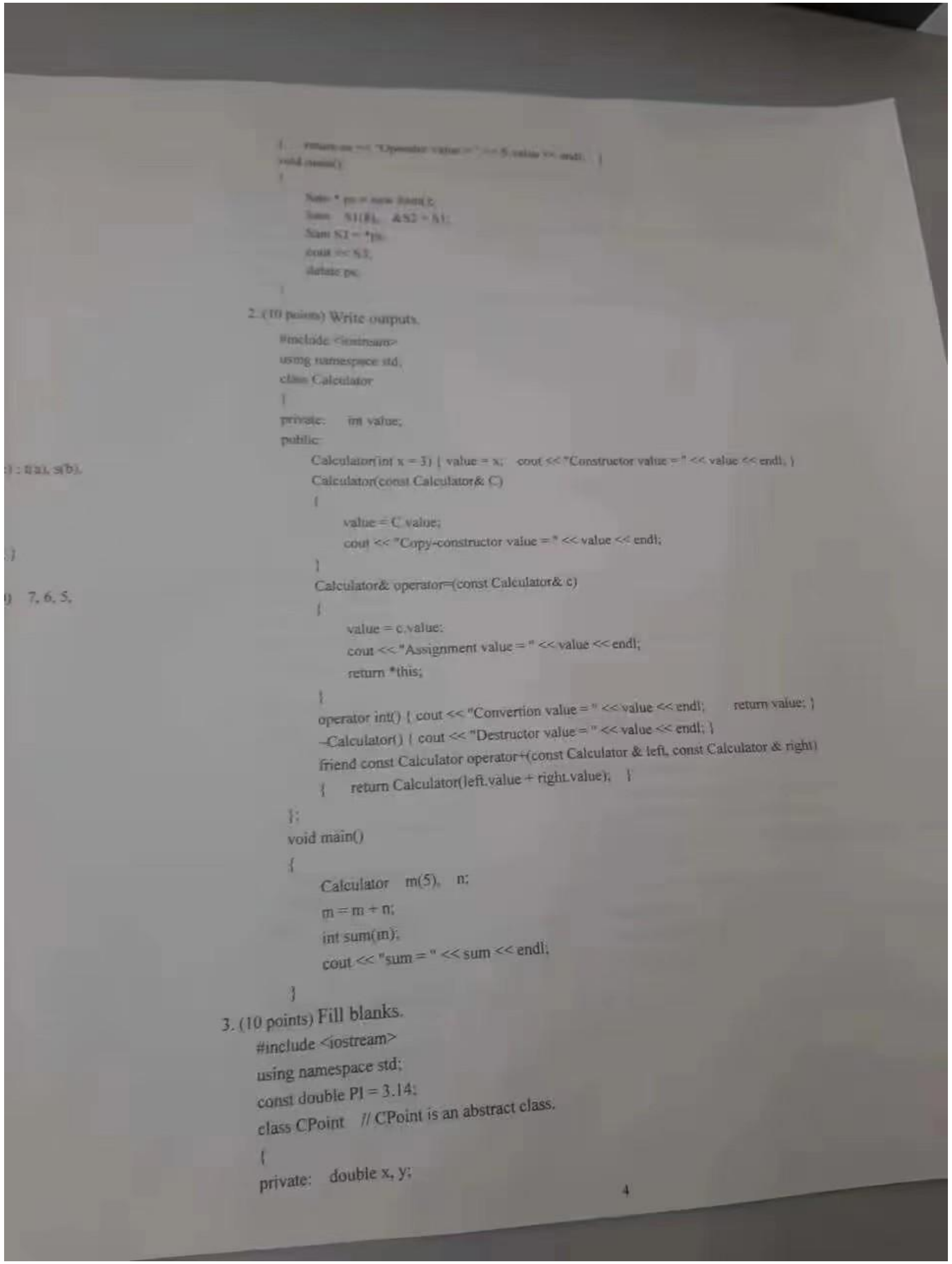
```
class Derived: public Base {
private:   Base   s, t;
public:   Derived(int a, int b, int c) : t(a), s(b),
Base(c) { }
};
void main() { Derived   d(5, 6, 7); }
```

(A) 5, 7, 6,　　(B) 7, 5, 6,　　(C) 5, 6, 7,　　(D) 7, 6, 5,

Part II. (40 points) Fill blanks or write outputs.

1. (10 points) Fill blanks and write outputs.

```
#include <iostream>
using namespace std;
class Sam
{
    int value;
public:
    Sam(int a = 0) : value(a)
    {         cout << "Constructor value = " << value << endl;      }
    Sam(const Sam& S) :_______________①_______________
    {         cout << "Copy-constructor value = " << value << endl;      }
    Sam& operator=(const Sam& S)
    {
        value = S.value;
        cout << "Assignment value = " << value << endl;
        _______________②_______________;
    }
    ~Sam() { cout << "Destructor value = " << value << endl; }
    _______________③_______________;
};
ostream& operator<<(ostream& os, const Sam& S)
```



): a(a), s(b),

}

) 7, 6, 5,

```
{    return os << "Operator value = " << S.value << endl;   }
void main()
{
        Sam * ps = new Sam[illegible];
        Sam   S1(8),   &S2 = S1;
        Sam S3 = *ps;
        cout << S3;
        delete ps;
}
```

2. (10 points) Write outputs.

```
#include <iostream>
using namespace std;
class Calculator
{
private:     int value;
public:
        Calculator(int x = 3) { value = x;    cout << "Constructor value = " << value << endl; }
        Calculator(const Calculator& C)
        {
                value = C.value;
                cout << "Copy-constructor value = " << value << endl;
        }
        Calculator& operator=(const Calculator& c)
        {
                value = c.value;
                cout << "Assignment value = " << value << endl;
                return *this;
        }
        operator int() { cout << "Convertion value = " << value << endl;        return value; }
        ~Calculator() { cout << "Destructor value = " << value << endl; }
        friend const Calculator operator+(const Calculator & left, const Calculator & right)
        {       return Calculator(left.value + right.value);    }
};
void main()
{
        Calculator    m(5),    n;
        m = m + n;
        int sum(m);
        cout << "sum = " << sum << endl;
}
```

3. (10 points) Fill blanks.

```
#include <iostream>
using namespace std;
const double PI = 3.14;
class CPoint    // CPoint is an abstract class.
{
private:    double x, y;
public:
    CPoint(double a , double b) { x = a; y = b; }
    _______①_______
};
class Circle : public CPoint
{
___②___:    double radius;
public:
    Circle(double a, double b, double r) : _____③_____ { }
    double Area() { return PI * radius * radius; }
};
class Sphere : public Circle
{
public:
    Sphere(double a, double b, double r) : _____④_____ { }
    double Area() {   return 4 * PI * radius * radius; }
};
void ShowArea (CPoint___⑤___) { cout << s.Area() << endl; }
void main()
{
    Sphere s(1, 1, 2);
    Circle c(1, 2, 3);
    ShowArea (s);       //50.24
    ShowArea (c);       //28.26
}
```

4. (10 points) Fill blanks and write outputs.

```
#include <iostream>
using namespace std;
template<typename T>   class Stack
{
public:
    Stack(): top(0) { }
    void push(const T& value);   // push an element to stack
    T pop();                     // Get an element at the top of stack
private:
    T stack[10];
    int top;
};
_______①_______{   stack[top++] = value;    } // push an element to stack
_______②_______{   return stack[--top];   }     // Get an element at the top of stack
void main( )
{
    Stack<int>   is;
    for(int i = 0; i < 8; i++)    is.push(i * 2);
    for(int k = 0; k < 8; k++)    cout << is.pop() << ",";
    cout << endl;
    Stack<double>   ds;
    for( int i = 0; i < 8; i++)    ds.push(i * 0.5);
    for( int k = 0; k < 8; k++)    cout << ds.pop() <<",";
}
```

PartIII (30 points) Programming

1. (15 points) Define appropriate member functions for class *Timer* in order to allow clients to use it in main( ) and get the results as in the block.

```
class Timer
{
    int hour, minute, second;     // 1 hour =60 minutes, 1 minute =60 seconds
public:
    // your functions:
    ……
};
void main()
{
    Timer T1(23, 59, 58);
    ++T1;
    cout << T1 << endl;
    Timer T2 = T1++;
    cout << T1 << endl;
    cout << T2 << endl;
}
```

```
23:59:59
00:00:00
23:59:59
```

2. (15 points) Please complete the definitions of class *Animal*、*Wolf* and *Tiger*.

```
#include <iostream>
#include <string>
using namespace std;
class Food
{
    string foodname;
public:
    Food(string s) : foodname(s) { };
    string Getfoodname() { return foodname; }
};
class Animal    // abstract class
{
    string animalname;
    Food& food;
public: // your functions:
        ……
};
class Wolf : public Animal
{
public: // your functions:
        ……
};
```

```
class Tiger : public Animal
{
public: // your functions:
    ……
};

void main()
{
    Food meat("meat");
    Animal* panimal = new Wolf("wolf", meat);
    panimal->Eat();     // display: Wolf::Eat
    cout << *panimal << endl; // display: Wolf likes to eat
    delete panimal;
    panimal = new Tiger("Tiger", meat);
    panimal->Eat();     // display: Tiger::Eat
    cout << *panimal << endl; // display: Tiger likes to e
    delete panimal;
}
```