Софийски университет „Св. Кл. Охридски“
Факултет по математика и информатика

ТЕСТОВЕ

ПО

ОБЕКТНО-ОРИЕНТИРАНО ПРОГРАМИРАНЕ

Магдалина Тодорова

февруари-юни 2014

Тест 1

Разгледайте следващия клас. Посочете неговите конструктори, мутатори и член-функции за достъп.

```
class A
{ public:
    A();
    A(int a);
      A(const A&);
    void f();
    void g();
    int h() const;
    void k() const;
  private:
    int n;
};

A::A()
{ cout << "A()\n";
  n = 5;
}
A::A(int a)
{ cout << "A(int)\n";
  n = a+1;
}
A::A(const A& a)
{ cout << "A(const A&)\n";
    n = a.n + 3;
}
void A::f()
{n++;
}
void A::g()
{ f();
  n = 2*n;
  f();
}
int A::h() const
{ return n;
}
void A::k() const
{ cout << n+1 << endl;
}
```

Какво ще изведе следващата главна функция?

```
int main()
{ A a;
  A b(3);
  A c = b;
  A d = A(1);
  a.f(); b.g(); c.f(); d.g();
  d.k();
  A e(a.h() + b.h() + c.h());
  e.k();
  return 0;
}
```

Добавете към класа *A* член-функция *c()*, която дели *n* на *5*, ако то е кратно на 5, и го заменя с *5n+1*, ако не е кратно на 5. Дефинирайте и член-функция *e(A b)*, която връща истина, ако соченият от неявния параметър обект е равен на явния параметър *b*.

Тест 2

Отбележете и обяснете грешките в програмата. Поправете ги, така че да се получи работеща програма. Намерете резултата от изпълнението й.

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

  class fi
  { public:
      fi(int, int = 5);
      void print();
      int f() const;
      int g() const;
    private:
      int x, y;
  };

   fi::fi(int a, int b)
   { x = a;
     y = b;
   }
   void fi::print()
   { cout << x << " " << y << endl;
   }
   int fi::f() const
   { return x;
   }
   int fi::g() const
   { return y;
   }

   class se
   { public:
       se(double, fi);
       void print() const;
       double h() const;
       fi p() const;
     private:
       fi a;
       double b;
   };
   se::se(double d, fi e)
```

```
{ b = d;
  a = e;
}
void se::print() const
{ cout << b << endl;
  a.print();
}
double se::h() const
{ cout << a.f() << " " << a.g() << endl;
  return b;
}
fi se::p() const
{ return a;
}

int main()
{ fi m(1), n;
  m.print();
  cout << n.x << " " << n.y << endl;
  se c(4.6, m);
  c.print();
  return 0;
}
```

Тест 3

Намерете резултата от изпълнението на програмата.

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

class A
{  public:
      A(int = 1, int = 2);
      A(const A&);
      void print() const;
      int get_x() const;
      int get_y() const;
   private:
      int x, y;
};
A::A(int a, int b)
{ cout << "A(" << a << ", " << b << ")\n";
  x = a;
  y = b;
}
A::A(const A& p)
{ cout << "A(const&)\n";
  x = p.x+1;
  y = p.y+2;
}
void A::print() const
{ cout << x << " " << y << endl;
}
int A::get_x() const
{ return x;
}
int A::get_y() const
{ return y;
}

class B
{ public:
     B(double, const A&, A);
     B(const B&);
     void print() const;
     double get_x() const;
     A get_a1() const;
     A get_a2() const;
  private:
    A a1, a2;
    double x;
};

B::B(double d, const A& e1, A e2) : a2(e2)
```

```
{ cout << "B(" << d << ", " << e1.get_x() << ", "
          << e1.get_y() << ", " << e2.get_x() << ", "
          << e2.get_y()<< ")\n";
   x = d;
   a1 = A(e1);
}
B::B(const B& b) : a1(b.a1), a2(b.a2)
{ cout << "B(const B&)\n";
  x = b.x;
}
void B::print() const
{ cout << x << endl;
  a1.print();
  a2.print();
}
double B::get_x() const
{ return x;
}
A B::get_a1() const
{ return a1;
}
A B::get_a2() const
{ return a2;
}

int main()
{ A p(7), q(p);
  q.print();
  B b(2.3, p, q), c = b;
  c.print();
  return 0;
}
```

Тест 4

Намерете резултата от изпълнението на програмата:

```
#include <iostream>
#include <cassert>
#include <string>
using namespace std;
class A
{ public:
      A(char* = "Aleksandar", char* = "Vassil");
      ~A();
      A(const A&);
      A& operator=(const A&);
      void print() const;
  private:
     char* s1;
     char* s2;
};

A::A(char* s, char* t)
{ cout  << "A(" << s << ", " << t << ")\n";
  s1 = new char[strlen(s)+1];
  assert(s1 != NULL);
  strcpy(s1, s);
  s2 = new char[strlen(t)+1];
  assert(s2 != NULL);
  strcpy(s2, t);
}
A::~A()
{ cout << "~A()\n";
  delete [] s1;
  delete [] s2;
}
A::A(const A& s)
{ cout << "A(const A&)\n";
  s1 = new char[strlen(s.s1)+1];
  assert(s1 != NULL);
  strcpy(s1, s.s1);
  s2 = new char[strlen(s.s2)+1];
  assert(s2 != NULL);
  strcpy(s2, s.s2);
}
A& A::operator=(const A& s)
{ cout << "A::operator=()\n";
  if (this != &s)
   { delete [] s1;
     delete [] s2;
```

```
      s1 = new char[strlen(s.s1)+1];
      assert(s1 != NULL);
      strcpy(s1, s.s1);
      s2 = new char[strlen(s.s2)+1];
      assert(s2 != NULL);
      strcpy(s2, s.s2);
   }
   return *this;
}
void A::print() const
{ cout << s1 << " " << s2 << endl;
}

int main()
{ A a1("Ivan", "Georgi"), a2(a1), a3(a2);
  a1.print();
  a2.print();
  a3.print();
  a3 = a1;
  a3.print();
  a2.print();
  return 0;
}
```

Тест 5

Намерете резултата от изпълнението на програмата:

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

class A
{ public:
     A(char* ="Ivanka", double = 1.0);
     ~A();
     A(const A&);
     A& operator=(const A&);
     void print() const;
  private:
     char* st;
     double x;
};

A::A(char* s, double y)
{ cout  << "A(" << s << ", " << y << ")\n";
   st = new char[strlen(s)+1];
   assert(st != NULL);
   strcpy(st, s);
   x = y;
}
A::~A()
{ cout << "~A()\n";
   delete [] st;
}
A::A(const A& s)
{ cout << "A(const s)\n";
   st = new char[strlen(s.st)+1];
   assert(st!=NULL);
   strcpy(st, s.st);
   x = s.x;
}

A& A::operator=(const A& s)
{ cout << "A::operator=()\n";
   if (this != &s)
   { delete st;
      st = new char[strlen(s.st)+1];
```

```
      strcpy(st, s.st);
      x = s.x;
    }
    return *this;
}
void A::print() const
{ cout << st << " " << x << endl;
}

class B
{ public:
       B(double, const A&);
       B(const B&);
       B& operator=(const B&);
       void print() const;
  private:
     double x;
     A a;
};

B::B(double d, const A& e) : a(e)
{ cout << "B::B(d, e)\n";
  x = d;
}
B::B(const B& p) : a(p.a)
{ cout << "B::B(const B& p)\n";
  x = p.x;
}
B& B::operator=(const B& p)
{ cout << "B::operator=()\n";
  if (this != &p)
    { x = p.x;
      a = p.a;
    }
   return *this;
}
void B::print() const
{ cout << x << endl;
  a.print();
}
int main()
{ A a1("Penka", 1.5), a2("Donka");
  B b(1, a1), c(2, a2), d(c);
  b.print(); c.print(); d.print();
  d = b; d.print();
  return 0;
}
```

Тест 6

За класа complex

```
class complex
{ public:
      complex(double = 0.0, double = 0.0);
      double get_Re() const;    // намира реалната част
      double get_Im() const;    // намира имагинерната част
      void print() const;
  private:
     double re,            // реална част
            im;            // имагинерна част
};
```

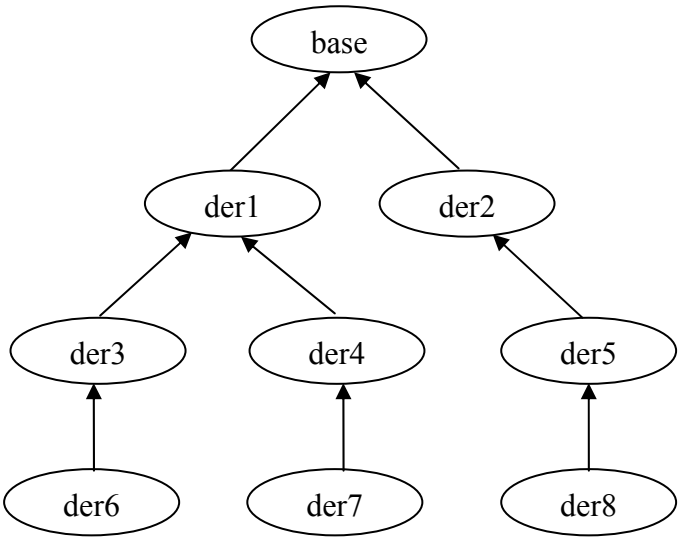
предефинирайте операторите:
а) +, който събира две комплексни числа;
б) +, който събира комплексно с реално число;
в) +, който събира реално с комплексно число;
г) !=, който проверява дали две комплексни числа са различни;
д) >, който проверява дали реално число е по-голямо от комплексно (ако double x и complex c, x е по-голямо от c, ако |x| > |c|).

Тест 7

Задача 1. За йерархията по-долу, определете:

а) прекия достъп на компонентите на всеки от класовете.

б) възможностите за достъп на обектите: b, d1, d2, d3, d4, d5, d6, d7 и d8 до компонентите на класовете.

<table>
<tr>
<td> 
</td>
<td>class base
{ public: void b1();
protected: void b2();
private: int b3;
} b;

class der1 : public base
{ public: void d11();
protected: void d12();
private: int d13;
} d1;

class der2 : protected base
{ public: void d21();
protected: void d22();
private: int d23;
} d2;</td>
</tr>
<tr>
<td>class der3 : protected der1
{ public: void d31();
protected: void d32();
private: int d33;
} d3;</td>
<td>class der4 : private der1
{ public: void d41();
protected: void d42();
private: int d43;
} d4;</td>
</tr>
<tr>
<td>class der6 : public der3
{ public: void d61();
protected: void d62();
private: int d63;
} d6;</td>
<td>class der7 : protected der4
{ public: void d71();
protected: void d72();
private: int d73;
} d7;</td>
</tr>
<tr>
<td>class der5 : public der2
{ public: void d51();
protected: void d52();
private: int d53;
} d5;</td>
<td>class der8 : protected der5
{ public: void d81();
protected: void d82();
private: int d83;
} d8;</td>
</tr>
</table>

Тест 8

Какъв е резултатът от изпълнението на програмата?

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

class FI
{ public:
      void init(int x, int y)
      { bx = x;
        by = y;
      }
      void display() const
      { cout << " FI::bx= " << bx << endl
             << " FI::by= " << by << endl;
      }
  protected:
      int bx;
  protected:
      int by;
};
class SE : public FI
{ public:
      void init(int x, int y, int z, int t)
      { FI::init(x, y);
        bx = z;
        by = t;
      }
      void display() const
      { FI::display();
        cout << " SE::bx = " << bx << endl
             << " SE::by = " << by << endl;
      }
  protected:
      int bx;
  private:
      int by;
};
class THI : public SE
{ public:
      void init(int x, int y, int z, int t, int p, int q)
      { bx = p;
        by = q;
        FI::init(x, y);
        SE::init(x, y, z, t);
```

```
      }
      void display() const
      { FI::display();
        SE::display();
        cout << " THI::bx = " << bx << endl
             << " THI::by = " << by << endl;
      }
   protected:
      int bx;
   private:
      int by;
};

int main()
{ FI a; SE b; THI c;
  a.init(0, 1); b.init(2, 3, 4, 5);
  c.init(1, 2, 3, 4, 5, 6);
  a.display(); b.display(); c.display();
  b.FI::init(1, 2);
  b.FI::display();
  b.display();
  c.SE::init(2, 3, 4, 5);
  c.FI::init(6, 7);
  c.display();
  return 0;
}
```

Тест 9

Намерете резултата от изпълнението на програмата.

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

class A
{ public:
      A(int x)
      { cout << "A(" << x << ")\n";
        a = x;
      }
      A& operator=(const A &x)
      { cout << "A::operator=()\n";
         if(this != &x) a = x.a;
         return *this;
      }
      void display() const
      { cout << "A: " << a << endl;
      }
   private: int a;
};

class B : public A
{ public:
      B(int x = 0) : A(3)
      { cout << "B(" << x << ")\n";
        y = x;
      }
      B(const B& p) : A(8)
      { cout << "B(const B&)\n";
        y = p.y-1;
      }
      B& operator=(const B& x)
      { cout << "B::operator=()\n";
        if (this != &x)
        { A::operator=(x);
          y = x.y+1;
        }
        return *this;
      }
      void display() const
      { A::display();
        cout << "B: " << y << endl;
      }
  private:
     int y;
 };
```

```
class C : public A
{ public:
      C(int x = 1) : A(x+2)
      { cout << "C(" << x << ")\n";
        y = x;
      }
      C(const C& p) : A(0)
      { cout << "C(const C&)\n";
        y = p.y+1;
       }
      C& operator=(const C& x)
      { cout << "C::operator=()\n";
        if (this != &x)
        { A::operator=(x);
          y = x.y+2;
        }
        return *this;
      }
      void display() const
      { A::display();
        cout << "C: " << y << endl;
      }
   private:
      int y;
};

class D : public B
{ public:
      D(int x = 5)
      { cout << "D(" << x << ")\n";
        y = x;
      }
      D(const D& p) : B(p)
      { cout << "D(const D&)\n";
        y = p.y+1;
      }
      void display() const
      { B::display();
        cout << "D: " << y << endl;
      }
  private:
      int y;
};

int main()
{ B x(6), y = x;
  C z(4), u;
  D v(2), t = v;
  cout << "x: "; x.display();
  cout << "y: "; y.display();
```

```
    y = x; cout << "y: "; y.display();
    cout << "z: "; z.display();
    cout << "u: "; u.display();
    u = z; cout << "u: "; u.display();
    cout << "v: "; v.display();
    cout << "t: "; t.display();
    t = v; cout << "t: "; t.display();
    return 0;
}
```

Тест 10

Намерете резултата от изпълнението на програмата. Напишете стойностите на обекта p след изпълнението на $p = q$.

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

class A
{ public:
    A(int a = 0, double b = 0)
    { n = a; x = b;
      cout << "A: " << n << ", " << x << endl;
    }
    A(const A& p)
    { n = p.n+1; x = p.x+1.5;
      cout << "A(const C&)\n";
      cout << "A.n: " << n << endl << "A.x: " << x << endl;
    }
    A& operator=(const A& p)
    { cout << "A::operator=()\n";
      if (this != &p)
      { n = p.n + 1; x = p.x + 1.5;
        cout << "A.n: " << n << endl << "A.x: " << x << endl;
       }
       return *this;
    }
  private:
    int n;
    double x;
};

class B
{ public:
    B(int a = 1, double b = 1)
    { n = a; x = b;
      cout << "B: " << n << ", " << x << endl;
    }
  private:
    int n;
    double x;
};
```

```
class C
{ public:
      C(int a = 2, double b = 2)
      { n = a; x = b;
        cout << "C: " << n << ", " << x << endl;
      }
      C(const C& p)
      { n = p.n + 1; x = p.x + 1.5;
        cout << "C(const C&)\n";
        cout << "C.n: " << n << endl << "C.x: " << x << endl;
      }
   private:
      int n;
      double x;
};

class der : B, protected C, public A
{ public:
      der(int x = 0, int y = 0, int z = 0): A(x), B(y, z), C(z, x)
      { n = z; m = x-y;
        cout << "der: " << n << ", " << m << endl;
      }
      der(const der& p) : C(p), A(p), a(p)
      { cout << "der(const der&)\n";
        n = p.n + 1; m = p.m + 1;
        cout << "der.n: " << n << endl  << "der.m: " << m << endl;
      }
   private:
      int n, m;
      A a;
      B b;
      C c;
};

int main()
{ der p, q(1,2,3);
  der r = p;
  p = q;
  return 0;
}
```

Тест 11

Разгледайте програмата:

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

class base
{ public:
      virtual void f()
      { cout << "base::f() \n";
      }
      base()
      { cout << "base()\n";
        f(); g(); h();
      }
      void usual()
       { cout << "usual()\n";
         f(); g(); h();
       }
   protected:
      virtual void g()
      { cout << "base::g()\n";
      }
  private:
     virtual void h()
     { cout << "base::h()\n";
     }
};

class der1 : public base
{    void f()
     { cout << "der1::f()\n";
       base::f();
     }
  protected:
     void g()
     { cout << "der1::g()\n";
     }
  public:
      der1()
      { cout << "der1()\n";
      }
       ~der1()
      { cout << "~der1()\n";
        f(); g(); h();
      }
     void h()
     { cout << "der1::h()\n";
     }
};
```

```
class der2 : public der1
{ protected:
      void f()
      { cout << "der2::f()\n";
      }
   public:
      der2()
      { cout << "der2()\n";
        f();
        g();
        h();
     }
     void g()
      { cout << "der2::g()\n";
        base::g();
      }
   private:
      void h()
      { cout << "der2::h()\n";

      }
};

int main()
{ base b; der1 d1; der2 d2;
  base *p = &d1;
  der1 *q = &d2;
  b.f();
  b.base();
  p->f();
  p->g();
  p->h();
  q->f();
  q->g();
  q->h();
  q->base();
  p = &d2;
  p->f();
  p->g();
  p-> usual();
  q->usual();
  return 0;
}
```

а) Намерете и обяснете грешките в процедурата *main* на горната програма.
б) Кои връзки в програмата се разрешават статично и кои динамично?
в) Какъв е резултатът от изпълнението на програмата след отстраняване на неправилните обръщения към член-функции?