天津大学研究生院 2007年招收硕士生入学试题

题号：528

考试科目：数据结构与程序设计

页数：8

(一) 数据结构

1 写出下列广义表的存储结构（给出一种方式即可）(5 分)

((),(e),(a,(b,c,d)))

2 (1)试分别画出具有三个结点的树和三个结点的二叉树的所有不同形态。(5 分)

(2)针对(1)中各种形态的二叉树分别写出先序,中序和后序遍历的序列。(3 分)

3. (1)写出堆排序的思想(3 分)

(2)给出向堆中加入数据 4, 2, 5, 8, 3, 6, 10, 14 时，每加入一个数据后堆的变化。(4 分)

4. (1)写出拓扑排序的算法(不要求编程)。(5 分)

(2)举例说明拓扑有序序列产生的全过程(至少要有六个顶点)。(5 分)

5. 编程：假设以数组 Q[m]存放循环队列中的元素，同时以 rear 和 length 分别指示环形队列中的队尾位置和队列中所含元素的个数。试给出该循环队列的队空条件和队满条件，并写出相应的初始化(initqueue)，插入(enqueue)和删除(dlqueue)元素的操作。(10 分)

6. 编程： 若用二叉链表作为二叉树的存储表示，试编写递归算法：(10 分)

(1) 统计二叉树中叶结点的个数。

(2) 以二叉树为参数，交换每个结点的左子女和右子女。

(二)：程序设计

一、阅读程序并写出结果：(第 1, 2, 3 题 6 分；第 4 题 7 分)

```
1. #include<iostream.h>

class original
{
      public:
          original(int i=0,int j=0){x0=i;y0=j;}
          virtual void set( )=0;
          virtual void draw( )=0;
      protected:
          int x0,y0;
};

class son1:public original
{
      public:
          son1(int i=0,int j=0,int m=0,int n=0):original(i,j)
          { x1=m;y1=n;              }
          void set( ){cout<<"son1::set("<<x1++<< ")called.\n";}
          void draw( ){cout<<"son1::draw( "<<--y1<<")called.\n";}
      protected:
          int x1,y1;
};

class son2:public original
{
      public:
          son2(int i=0,int j=0,int p=0,int q=0):original(i,j)
          { x2=p;y2=q;   }
          void set( ){cout<<"son2::set("<<++x2<<" )called.\n";}
          void draw( ){cout<<"son2::draw("<<y2--<<" )called.\n";}
      protected:
          int x2,y2;
};

void drawobj(original *p)
```

```
{     p->draw( ); }

void setobj(original *p)
{     p->set( ); }

void main( )
{
      son1 *s1obj=new son1;
      son2 *s2obj=new son2;
      drawobj(s1obj);
      drawobj(s2obj);
      cout<<"another one!"<<endl;
      setobj(s1obj);
      setobj(s2obj);
      cout<<"\nRedraw the objects\n";
      drawobj(s1obj);
      drawobj(s2obj);
}
```

Son1::draw(4) called.
Son2::draw(0) called.
Call another!
Son1:: set(0) called
Son2:: Set(1) called.
Redraw the objects
Son1::draw(-2) called
Son2::draw(-1) called

```
2. #include<iostream.h>

class A
{
   public:
      A( ){a=0;cout<<"A's default constructor called.\n";}
      A(int i){a=i;cout<<"A's constructor called.\n";}
      ~A( ){cout<<"A's destructor called.\n";}
      void Print( ) const {cout<<a<<",";}
      int Geta( ){return a;}
   private:
      int a;
};

class B:public A
{
   public:
      B( ){b=0;cout<<"B's destructor called.\n";}
```

```
          1  2    (
        B(int i,int j,int k);
        ~B( ){cout<<"B's destructor called.\n";}
        void Print( );
    private:
        int b;
        A aa;
};
             ,  )
B::B(int i,int j,int k):A(i),aa(j)
{
    b=k;
    cout<<"B's constructor called.\n";
}

void B::Print( )
{
    A::Print( );
    cout<<b<<","<<aa.Geta( )<<endl;
}

void main( )
{
    B bb;
    bb=B(1,2,5);
    bb.Print( );
}
```

A's def
B's [illegible]
A's [illegible]
B's con
1 , 5 , 2
B's des
A's des

```
3. #include<iostream.h>

class A
{
          3  6
    public:  1  2
        A(int i, int j)
        { a = i; b = j;}
        void Move(int x,int y)
        { a += x; b += y;}
        void Show( )
        {
       cout << "(" << a << "," << b << ")" << endl;
        }
    private:
        int a,b;
};

class B:public A
{
    public:  3   4   5   6
        B(int i,int j,int k,int l):A(i,j),x(k),y(l)
        { }
        void Show( ) { cout << x << "," << y << endl;}
        void fun( ) { Move(3,5); }
        void f1( ) { A::Show( ); }
        int x,y;
};

void main( )
{
    A c(1,2);
    c.Show( );      (1, 2)
    B d(3,4,5,6);
    d.fun( );       (6, 9)
    d.A::Show( );
    d.B::Show( );   (5,6)
    d.f1( );        (6,9)
```

```
4. #include<iostream.h>
#include<string.h>

class base
{
    public:     4
        base(int st);
        ~base( );
    private:
        char string[10];
};

base::base(int st)
{
    int i,str1='A';
    char string[]="ABCDEFGHI";
    for(i=st;i<8;i++)
    string[i]=str1++;
    string[9]='\0';
    cout<<"Constructor called for "<< string<< endl;
}

base::~base( )
{
    char string[ ]="xxxxxxxxx";
    cout<<"Destructor called for "<< string<< endl;
}

void fun( )
{
    base b2(7);
    cout<<"In fun"<<endl;
}

void main( )
{
    base b1(4);
    cout<<"before calling fun"<<endl;
    fun( );
    cout<<"after calling fun"<<endl;

}
```

Con ABCD ABCD

Con - ABABCDEF

In

Des