东 南 大 学 考 试 卷（A 卷）

98

课程名称 计算机科学基础（下） 考试学期 13-14-3 得分

适用专业 信息工程 考试形式 闭卷 考试时间长度 120 分钟

题 目	一	二	三	四	总 分
得 分	20	10	40	28	
批阅人	郑	郑	缘	缘	

+20

一．选择题：（请将答案填写在下面表格中。每空 2 分，共 20 分）

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
C	B	A	C	D	A	B	D	C	D

1．下面对于友元函数描述正确的是_____(1)_____。

A．友元函数的实现必须在类的内部定义　　B．友元函数是类的成员

C．友元函数破坏了类的封装性和隐藏性　　D．友元函数不能访问类的私有成员

2．C++中的虚基类可以保证_____(2)_____。

A．限定该基类只通过一条路径派生出子类

B．当一个类多次从基类间接派生后，其基类只能被继承一次

C．当一个类多次从基类间接派生后，派生类对象能保留多份该基类的成员

D．允许该基类通过多条路径派生子类，派生类也就可以多次继承该基类

3．关于虚函数，下列描述中不正确的是_____(3)_____。

A．派生类的虚函数与基类的虚函数必须具有相同的函数名和返回值，而参数可以不同

B．虚函数必须是成员函数

C．基类中声明了虚函数后，派生类中其对应的函数可不必声明为虚函数

D．虚函数不可以是 static 类型的成员函数

4．在下面的类的定义中，横线处应填入的内容是_____(4)_____。

```
class F{
    double data;
public:
    void print(){cout<<data<<endl;}
    void setData(double d){data=d;}
    static int k;
};
_____k=0;
```

A. int　　B. static int　　C. int F::　　D. static int F::

5．下列叙述正确的是_____(5)_____。

A.虚函数是一个 static 类型的成员函数　　B.虚函数是一个非成员函数

C.包含虚函数的基类为虚基类　　D.包含纯虚函数的基类为抽象类

6．当定义派生类的对象时，调用构造函数的正确顺序是_____(6)_____。

A．先调用基类的构造函数，再调用派生类的构造函数

B．先调用派生类的构造函数，再调用基类的构造函数

C．调用基类的构造函数和调用派生类的构造函数的顺序无法确定

D．调用基类的构造函数和调用派生类的构造函数是同时进行的

7．判定一个顺序栈 st(最多元素为 MaxSize)为满的条件是_____(7)_____。

A. st->top!=-1　　B. st->top= =MaxSize

C. st->top==-1　　D. st->top!=MaxSize

8．对于动态分配内存空间描述正确的是_____(8)_____。

A．使用 new 运算符分配的内存空间的长度必需是常量

B．delete 运算符可以释放动态的存储空间和静态的存储空间

C．由 new 分配的内存空间是不连续的

D．delete 运算符只能释放由 new 分配的动态存储空间

9．下列有关类和对象的说法中，正确的是_____(9)_____。

A．类和对象没有区别

B．要为类和对象分配存储空间

C．对象是类的实例，为对象分配存储空间而不为类分配存储空间

D．类是对象的实例，为类分配存储空间而不为对象分配存储空间

10．实现运行时的多态性要使用_____(10)_____。

A.重载函数　　B.构造函数　　C.析构函数　　D.虚函数

+10

二．填空题(每空 2 分，共 10 分)：

1．下面的类定义中，是否有错？如有错请说明错在哪里。

```
class Line{
  private:
    int start_x=0,start_y=0;
```

```
    int end_x=0,end_y=0;
  public:
    int draw();
    int is_on_line(int x,int y);
};
```

有错，类内的数据成员不可初始化

2. 用某种排序方法对关键字序列（32,18,41,23,2,56,36,67）进行排序时，序列的变化情况如下：

18,32,23,2,41,36,56,67
18,23,2,32,36,41,56,67
18,2,23,32,36,41,56,67
2,18,23,32,36,41,56,67
2,18,23,32,36,41,56,67

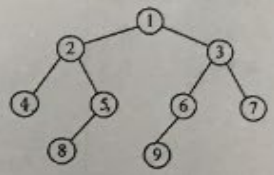
则所采用的排序方法是 冒泡排序。

3. 写出下面程序的输出结果：

```
double Quadratic(int A,int B,bool choose){
    if(A) {
            if((A- B)<0)    throw "Error";
            else    if(choose)    return A+B;
            else     return A*B;
    }
    else
        throw "B/A:except of deviding zero.";
}
int main(){
    try {
        cout<<"The answer is:"<<Quadratic(5,1,1)<<endl;
        cout<<"And"<<Quadratic(0,2,0)<<'\n';
        cout<<"Enter the next three number:";
    }
    catch(char *ptr) {
            cout<<ptr<<endl;
    }
    catch(int   answer) {   cout<<"The answer is:"<<answer<<'\n';    }
}
```

输出结果：
The answer is: 6
B/A: except of deviding zero

4.写出下面二叉树的后序遍历结果：48529673 1。



三. 读程序写结果（每空 2 分，共 40 分）

1.阅读以下程序，写出运行结果（12 分）。

```
#include<iostream>
#include<string>
using namespace std;
class Cplant
{
    int num;
    string name;
public:
    Cplant(int n=0,string str="\0")
    {
        num=n;
        name=str;
        cout<<"Constructor!\n";
    }
    Cplant(Cplant &obj)
    {
        num=obj.num;
        name=obj.name;
        cout<<"Copy constructor!\n";
    }
    ~Cplant()
    {
        cout<<"num:\t"<<num<<"\tname:\t"<<name<<endl;
        cout<<"desturctor!\n";
    }
};
int main()
{
    Cplant Pineapple(40,"Pineapple");
```

1. 输出结果：
Constructor!
Copy constructor!
~~40~~
num: 40 name: Pineapple
desturctor!
num: 40 name: Pineapple
desturctor!

```
    Cplant plant=Pineapple;
    return 0;
}
```

2.阅读以下程序，写出运行结果 （12 分）

```
#include<iostream>
using namespace std;
class point
{
    int x,y;
public:
    point(int a,int b):x(a),y(b)
    {cout<<"Abstract class!\n";}
    virtual double area()=0;
    void show()
    {cout<<x<<'\t'<<y<<endl;}
};
class circle:public point
{
    int r;
public:
    circle(int a,int b,int c):point(a,b),r(c)
    {cout<<"Derived class!"<<endl;}
    double area(){return 3.14*r*r;}
    void show(){cout<<r<<'\t'<<area()<<endl;}
};
int main()
{
    circle obj(0,1,2);
    point *bp=&obj;
    cout<<obj.area()<<endl;
    obj.show();
    cout<<bp->area()<<endl;
    bp->show();
    return 0;
}
```

2.输出结果：

Abstract class!
Derived class!
~~12.56~~ 12.56
2 12.56
12.56
0 1

3.阅读以下程序，写出运行结果(8 分)

```
#include<iostream>
#include<string>
```



```
using namespace std;
class panda
{
    double weight;
    static string color;
public:
    panda(double a,string str)
    {
        weight=a;
        color=str;
    }
    void setcolor(string str)
    {   color=str;   }
    void show()
    {   cout<<weight<<'\t'<<color<<endl;}
};

string panda::color = "dark";

int main()
{
    panda fatpanda(65,"white");
    fatpanda.show();
    panda slimpanda(40,"black");
    slimpanda.show();
    fatpanda.show();
    slimpanda.setcolor("black and white");
    fatpanda.show();
    return 0;
}
```

3.输出结果：

65 white
40 black
65 black
65 black and white

4.阅读以下程序，写出运行结果（8 分）

```
#include<iostream>
using namespace std;
class food
{
protected:
    char name[20];
public:
```

```
    food(char *n)
    {
        strcpy(name,n);
        cout<<"food\n";
    }
};
class noodle:virtual public food
{
protected:
    double width;
public:
    noodle(double w,char *n):food(n),width(w){cout<<"noodle\n";}
};
class filling:virtual public food
{
protected:
    bool Ismeat;
public:
    filling(bool m, char *n):food(n),Ismeat(m){cout<<"filling\n";}

};
class dumpling:public filling, public noodle
{
    char taste[10];
public:
    dumpling(char *t,double w,bool m,char *n):
                                noodle(w,"baozi"),filling(m,"huntun"),food(n)
    {
        strcpy(taste,t);
        cout<<"dumpling:\t"<<name<<'\t'<<width<<'\t'<<Ismeat<<'\t'<<taste<<endl;
    }
};
int main()
{
    dumpling favorite("delicious",20.5,true,"dumpling");
    return 0;
}
```

4.输出结果：

food
filling
noodle
dumpling: dumpling
dumpling: dumpling 20.5 1 delicious

四. 完善程序题 （每空 2 分，共 40 分） +28

1. 以下两个函数模板 reverseLinkedNodes 和 deleteLinkedNodes 分别实现单向节点链的反向与删除，其中：类模板 Node 的数据成员 data 代表承载的数据，而 next 是指向下一个节点的指针，如果其后无节点则域 next 值为 NULL；
reverseLinkedNodes 的形参 head 是指向该节点链的第一个节点的指针，函数反向该节点链并统计链所包含的节点个数后赋值给形参 size，同时返回值是反向后节点链的第一个节点的指针；
deleteLinkedNodes 的形参 head 是指向该节点链的第一个节点的指针，其删除该链上为所有节点动态分配的内存空间。
请完善程序。

```
template <typename ____X____> class Node
{
public:
    X data;
    Node * next;
};

template <typename T> Node <T> * reverseLinkedNodes(Node<T> * head, int &
size)
{
    int ____counter=0____;

    Node<T> * ____p____ = head;
    while (p != NULL) {
        counter++;
        Node<T> * q = p;
        p = p->next;
        ____q->next____ = head;
        ____head=q____;
    }
    size = counter;

    return ____head____;
}

template <typename T> void deleteLinkedNodes(Node<T> * head)
{
    Node<T> * p = head;
```

```
    while (p != NULL) {
        Node<int> * q = p;
        p = p->next;
        delete q;
    }
}
```

2. 函数 sort 应用冒泡算法实现对包含 length 个元素的整数数组 buffer 的降序排序，请完善程序。

```
void sort(int * buffer, int length)
{
    for (int i = 0; i < length - 1; i++) {
        bool swapped = false;
        for (j = length - 1; j >= i; j--) {
            if (buffer[j+1] > buffer[j]){
                int temp = buffer[j];
                buffer[j] = buffer[j + 1];
                buffer[j + 1] = temp;
                swapped = true;
            }
        }
        if (!swapped) break;
    }
}
```

3.以下三个类 MyTime、MyDate 和 MyDateTime 分别实现对时间（时分秒）、日期（年月日）以及日期时间（年月日时分秒）的输出，执行主函数后输出为:

2014 年 6 月 20 日
12 时 0 分 0 秒
2014 年 6 月 20 日
0 时 0 分 0 秒
2014 年 6 月 20 日
0 时 0 分 0 秒
2014 年 6 月 20 日 12 时 0 分 0 秒

请完善程序:


```
#include <iostream>
using namespace std;
class MyTime
{
    int hour, minute, second;
public:
    MyTime(int h, int m, int s) : hour (h), minute (m), second (s)
    {
        cout<<hour<<"时"<<minute<<"分"<<second<<"秒"<<endl;
    }
    MyTime(MyTime & source)
    {
        hour = source.hour; minute = source.minute; second = source.second;
        cout<<hour<<"时"<<minute<<"分"<<second<<"秒"<<endl;
    }
    friend ostream & operator<<(ostream & sink, MyTime & one)
    {
        sink<<one.hour<<"时"<<one.minute<<"分"<<one.second<<"秒";
        return sink;
    }
};

class MyDate
{
    int year, month, day;
public:
    MyDate(int y, int m, int d) : year (y), month (m), day (d)
    {
        cout<<year<<"年"<<month<<"月"<<day<<"日"<<endl;
    }
    MyDate(MyDate & source)
    {
        year = source.year; month = source.month; day = source.day;
        cout<<year<<"年"<<month<<"月"<<day<<"日"<<endl;
    }
    friend ostream & operator<<(ostream & sink, MyDate & one)
    {
        sink<<one.year<<"年"<<one.month<<"月"<<one.day<<"日";
        return sink;
    }
```

```
};

class MyDateTime : public MyDate
{
    MyTime time;
public:
    MyDateTime(int year = 2014, int month = 6, int day = 20, int hour = 12, int
minute = 0, int second = 0)
        : MyDate (year, month, day), time (hour, minute, second) { }

    friend ostream & operator<<(ostream & sink, MyDateTime & one)
    {
        MyDate &date = one;
        sink<<date;

        sink << one.time <<endl;
        return sink;
    }
};

int main(void)
{
    MyDateTime dt;
    MyDateTime dt2 (2014, 6, 20, 0, 0, 0);
    MyDateTime dt3 = dt2;
    cout<<dt;
    return 0;
}
```