复试整理

2010年

一、简答题（60分）

1、简述你对“面向过程”和“面向对象”编程思想的认识与思考

答：面向过程是将解决问题的重点放在如何实现过程的细节方面，把数据和对数据进行操作的函数截然分开，以数据结构为核心，围绕着功能的实现或操作流程来设计程序，安全性较低、扩展升级麻烦，当问题的规模变大时，编程将很复杂；面向对象将客观事物看作具有属性和行为的对象，通过抽象找出同一类对象的共同属性（静态特征）和行为（动态特征），形成类。通过继承与多态可以很方便地实现代码重用，大大缩短了软件开发周期，并使软件风格统一。

2、ADT是什么？简述你对“数据抽象”和“信息隐藏”的认识

答：ADT即抽象数据类型，是基于已有类型而组合生成的复合数据类型，类正是抽象数据类型的描述形式。数据抽象：指以数据为中心，把数据及在数据上的操作作为一个整体来进行描述；信息隐蔽：通过封装，实现信息隐蔽；将类对外部的接口放在类的公有部分，数据放在类的私有部分中，防止用户在类外直接访问数据，并且屏蔽了类的实现细节，实现信息隐藏。

3、const和static有什么作用?

答：关键字const用于定义常量，可以保护被修饰的量，防止意外的修改，增强程序的健壮性，编译器一般不为普通const常量分配存储空间，而是将它们保存在符号表中，没了存储与读内存的操作，效率较高，同时节省了空间，避免了不必要的内存分配；static用于声明静态成员，它只有一个变量拷贝，供类的所有对象共享，静态类变量表示整个类范围中（所有类对象而非指定的类对象）共享的信息。

4、友元关系的利与弊

答：友元可以是一个函数也可以是一个类，友元提供了不同类的成员函数之间、类的成员函数和一般函数之间进行数据共享的机制，通过友元，一个一般函数或另一个类中的成员函数可以访问类中的私有成员和保护成员。友元的正确使用可以提供程序运行效率，实现信息共享，但同时也破坏了类的封装性和隐藏性，导致程序可维护性变差。

5、C++多态的实现

答：多态是指同样的消息被不同类型的对象接收时导致完全不同的行为，是对类的特定成员函数的再抽象。C++支持的多态有多种类型，重载（包括函数重载和运算符重载）和虚函数是其主要的方式。

6、STL是什么？组成部分和核心作用

答：标准模板库(Standard Template Library)，有三部分组成：容器(container)、迭代器(iterator)和算法(algorithm)。程序员可以重用STL容器、迭代器和算法实现通用的数据表示和操作，节约大量的开发时间和资源。

二、程序设计题（90分）

这篇博客中有所有题目的参考代码与解析：

[2010] http://blog.csdn.net/qq_32925781/article/details/79377073

2011年

一、简答题（50分）

1、简述你对C++中的数据类型和抽象数据类型（ADTs）的理解

答：数据类型是一组性质相同的具有一定范围的值集以及定义于这个值集上的一组操作。数据类型既有内部数据类型，如int，char，float，bool等，又有自定义外部数据类型，如枚举类型，结构类型，联合类型，数组类型、类类型。ADT即抽象数据类型，是基于已有类型而组合生成的复合数据类型，类正是抽象数据类型的描述形式。

2、请举例并写出相关代码，阐述C++在什么情况下必须进行运算符重载

答：只有当二元运算符最左边的操作数是该类的一个对象，或者当一元运算符的操作数是该类的一个对象时，才有必要进行运算符重载。

比如下面这个实例中必须对运算符+进行重载：

```
//运算符重载实例
#include <iostream>
using namespace std;

class MyClass{
private:
    int x;
    double y;

public:
    MyClass(int xx = 0,double yy = .0){
        x = xx;
        y = yy;
    }

    void print() const {
        cout << "x:" << x << " y:" << y << endl;
    }

    //以成员函数的方式重载+运算符
    MyClass & operator +(const MyClass & temp){
        x += temp.x;
        y += temp.y;
        return *this;
    }
};

int main(){
    MyClass a(1,1.1);
    MyClass b(10,2.342);
    MyClass c;

    c = a + b;      //二元运算符+的左边是MyClass类的对象
    c.print();

    return 0;
}
```

3、为什么说“继承是C++面向对象的一个主要特征之一”，请简要说明

答：继承是软件重用的一种形式，程序员创建的新类通过继承这一方式 ，从现有的类中吸收其数据和行为，再根据新类的特征，即对象的特征，赋予其新的数据和行为，并通过新的功能进一步提高，在面向对象的程序设计中，采用继承方式来组织设计系统的类，可以提高程序的抽象程度，更接近人的思维方式，使程序结构更清晰并降低编码和维护的工作量。

4、如何声明和使用虚函数，说明它在多态性中的作用和意义

答：声明虚函数时，在基类的函数原型前加关键字virtual，在派生类中重写函数；用名称和圆点成员选择运算符引用一个特定的对象以调用虚函数，被调用的虚函数是在编译时确定的；也可以通过基类指针调用函数，让程序在任何给定的时刻基于基类指针所指对象的对象类型，动态确定执行相应的派生类的函数。虚函数是多态性的主要实现方式，利用虚拟函数和多态性，程序员可以处理普遍性而让执行环境处理特殊性，即使在不知道一些对象的类型的情况下（只要这些对象属于同一继承层次并且通过一个共同的基类指针访问），程序员也可以命令各种对象表现出适合这些对象的行为。

5、请说明函数模板和模板函数的区别与联系

答：函数模板：以具体的类型为实参来生成函数体的模板；模板函数：函数模板实例化的结果，由函数模板将涉及的数据类型为参数来生成的模板化函数

函数模版就是数据类型参数化的函数定义，是一个函数族，代表的是一类函数。 当编译系统发现用指定数据类型调用函数模版时，就创建了一个模版函数，模版函数是一个实例化的具体函数。

二、编程题（100分）

这篇博客中有所有题目的参考代码与解析：

[2011] http://blog.csdn.net/qq_32925781/article/details/79389350

2012年

一、简答题

1、编写语句说明枚举类型是如何定义和使用的

答：Enum Week{Mon，Tue，Wed，Thu，Fri，Sat，Sun}；枚举类是对整数区间的自定义类型，一旦定义则不能改变，常常代替整数常量使用，可以使程序更清晰、更持久。在进入函数调用或其他模块时，常量需要初始化，而枚举类型是一种类型，无须定义实体，便可直接使用枚举符。默认对应着整数0,1,2...

当然也可以Enum Week{Mon=1，Tue，Wed，Thu，Fri，Sat，Sun}；这样定义，对应的整数就变成1,2,3...

2、程序改错

第一题：

```
 #include <iostream>
 using namespace std;
 int main()
 {
   for(int i=0;i<8;i++)
   {
      if(i%2==0)
          cout<<i+1<<endl;
      if(i%3==0)
          continue;
      if(i%5==0)
          break;
      cout<<"End of programming\n";
      //没懂这个程序想要干嘛，把上面这句cout删除?
   }
   cout<<"End of programming\n";
   return 0;
  }
```

第二题:

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

int main() {
    int c;
    if((c=cin.get())!=EOF){
        main();
        cout << c;
        //把上面这句语句改成 cout << char(c) ;
        //不然就会输出ASCII码
    }
    return 0;
}
```

第三题:

```
#include <iostream>
using namespace std;
void fun(int a[],int cur,int s)
{
    {
        if(cur < s)    //把这里的s改成s-1就能正确输出了
            fun(a,cur+1,s);
        cout<<a[cur]<<',';
    }
}
int main()
{
    int a[]={1,2,3,4,5,6,7,8,9,10};
    fun(a,0,10);
    return 0;
}
```

3、一个函数模板和模板函数的区别

同2011年第5题

4、为什么说“继承是面向对象的主要特征之一”，请简要说明

同2011年第3题

5、如何声明和使用虚函数，说明它在多态性中的作用和意义

同2011年第4题

二、编程题

这篇博客中有所有题目的参考代码与解析：

[2012] http://blog.csdn.net/qq_32925781/article/details/79399352

2013年

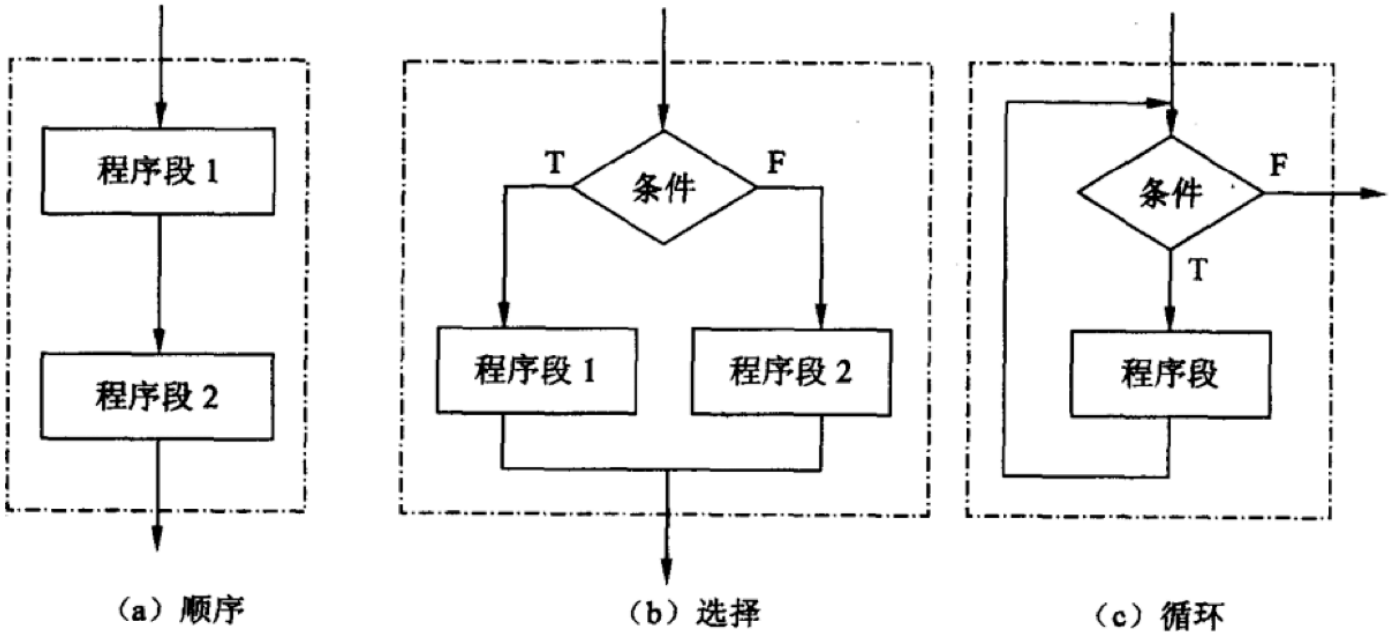
一、简答题

1、什么是逻辑错误?什么是语法错误?请举例说明。

答：语法错误是对语言规则的违背，当编译器不能正确识别语句时，便会导致语法错误，它们都是在编译阶段出现的，所以也叫编译错误，编译不能通过，比如语句末尾缺少分号等。逻辑错误是指算法上的错误，编译能通过，程序可以完成运行，但往往产生不正确的结果，比如循环语句结束条件没写，造成死循环。

2、控制语句有哪几种?请画出它的流程图或 UML 图。

顺序，选择，循环。UML就是统一建模语言，是用的最广泛的图形化表示方案。



3、什么是运算符重载?它如何增强了 C++的扩展性?

答：对已有的运算符赋予多重含义，使同一个运算符作用于不同类型的数据时导致不同的行为。同一个运算符以适应不同的数据类型，增强了C++的扩展性。

4、为什么说”继承是 C++面向对象的主要特征之一”?

同2011年第3题

5、类作用域和文件作用域的区别是什么?请详细说明。

答：函数原型作用域：在函数原型声明时形式参数的作用范围就是函数原型作用域。

块作用域：函数形参列表中形参的作用域，从形参列表中的声明处开始，到整个函数体结束之处为止；函数体内声明的变量，其作用域从声明处开始，一直到块结束的大括号为止。

类作用域：类X的成员M在以下几种情况下具有类作用域：

（1） 在X的成员函数内出现标识符M，同时在该函数中也没有声明同名的局部作用域标识符。

（2） 在表达式x.M或者x::M中，类X的对象x通过作用符号“.”,”::”访问的M也具有类作用域，这正是在程序中访问对象的成员的最基本的方法。

（3） 在prt->M这样的表达式中，其中prt为指向X类的一个对象的指针。

文件作用域：在任何函数外部声明的标识符的作用范域为文件作用域，可以从声明标识符到文件末尾处的任

何函数中访问标识符。

二、编程题

这篇博客中有所有题目的参考代码与解析：

[2013] http://blog.csdn.net/qq_32925781/article/details/79410775

2014年

一、简答题

1、指出错误并改正

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

void f(int *p) {
    if(p){
    *p *= 100;
    cout<<*p<<endl;
    }
}

int main()
{
    int p = 10;
    int * ptr;  //这里改成 int * ptr = &p;
    f(ptr);

    return 0;
}
```

2、说出下面程序输出什么

题目不全，答案待定。

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

int main()
{
    char  str[] = "THIS IS PROGRAMMING.";
    int length = 21;
    int j = 0;
    for(int k = 0; k < length; k++){
        if(str[j] < str[k])
            j = k;
    }
    int temp = j;
    str[j] = str[7];
    str[7] = str[13];
    str[13] = str[length];
    cout<< str <<endl;

    return 0;
}
```

3、程序输出结果

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

void fun(int i,int j,int*a) {
    *a=j-i;
}

int  main()
{
    int a,b,c;
    fun(20,9,&a);
    fun(9,a,&b);
    fun(a,b,&c);
    cout<<a<<","<<b<<","<<c<<endl;

    return 0;
}
```

输出结果是：

```
-11,-20,-9
```

4、是关于二维数组的，其实就是求矩阵行之和列之和

题目不全。

5、是问输出什么

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

class A
{
public:
    virtual void print(){
        cout<<"A::print"<<endl;

    }
};

class B: public A
{
public:
    void print()
    {
        cout<<"B::print"<<endl;
    }
};

class C: public B
{
public:
    void print(){
        cout<<"C::print"<<endl;
    }
};

int main() {
    B b;
    C c;
    A* a;
    a = &c;
    a->print();
    a = &b;
    a->print();

    return 0;
}
```

输出结果是：

```
C::print
B::print
```

二、编程题

这篇博客中有所有题目的参考代码与解析：

[2014] http://blog.csdn.net/qq_32925781/article/details/79413824

2015年

只有五道编程题

这篇博客中有所有题目的参考代码与解析：

[2015] http://blog.csdn.net/qq_32925781/article/details/79420284

2016年

一、填空

1.程序从哪里开始运行 main()函数

2.三种控制结构 顺序、选择、循环

3.可以初始化指针的值 nullptr、0、具体的地址

注：nullptr是C++11新引入的，写NULL也行。

4.对象之间看不到其他对象的具体操作的属性是 信息隐藏

注：信息隐藏包括但不限于private、protected及接口类这些手段。注意区分封装和信息隐藏之间的区别。

5.用户自定义类型的一个实例叫什么 对象

注：类是抽象数据类型(ADTs)的描述形式

二、简答

1.std::cin,std::cout,std::cerr的含义。

答：都是标准流对象

cin：istream实例，连接到标准输入设备

cout：ostream实例，连接到标准输出设备

cerr：ostream实例，连接到标准错误设备，输出是**无缓冲**的

补充一个 clog：ostream实例，连接到标准错误设备，输出是**有缓冲**的

2.判断对错、改错，题目不全

3.存储类说明符有哪些？作用分别是?

答：auto、static、extern、register

auto 标明一个变量具有自动存储时期，该说明符只能用在具有代码块作用域的变量声明中。比如一般的局部变量或是在函数原型中的参数。

register 只能用在具有代码块作用域的变量。请求一个变量存储在**寄存器**中快速使用，但是不能获得改变量的地址。register常常是不必要的。

static 使用static声明的局部变量仅被其声明所在的函数所知，static局部变量在函数返回到它的调用者后仍保留着变量的值。

extern 表明在声明一个在其他地方定义了的变量(该关键字用于全局变量)。

类型	特性
auto	具有代码块作用域，**自动存储期**，外部或者内部链接属性
register	具有代码块作用域，**自动存储期**，外部或者内部链接属性
static	具有代码块或者文件作用域、**静态存储期**、内部链接
extern	具有文件作用域、**静态存储期**、外部链接

4. * 运算符有哪些含义。

答：乘法、指针的间接运算符。

```
int *ptr;    //声明一个指针
*ptr = 10;   //表示指针操作数所指向的对象(内存块)
```

5.有哪些运算符不能重载

答:

```
.       .*(成员指针运算符)       ::        ?:
```

6.多态性如何实现。

答：多态是指同样的消息被不同类型的对象接收时导致完全不同的行为，是对类的特定成员函数的再抽象。C++支持的多态有多种类型，重载（包括函数重载和运算符重载）和虚函数是其主要的方式。

三、程序题

这篇博客中有参考代码与解析:

[2016] http://blog.csdn.net/qq_32925781/article/details/79423783

2017年

一、填空题(70分)

1.如果每次运行环境只能执行一条语句,但是有许多语句需要执行,那么 用花括号{}括起来 , 构成语句块 。

2.标识符的作用域 语句块作用域、函数原型作用域、函数作用域、类作用域、命名空间及全局命名空间作用域 。

3.用字符串“schedule”初始化一个字符数组的初始化语句

```
char s[] = "schedule";
char s[] = {"schedule"};
char s[] = {'s','c','h','e','d','u','l','e','\0'};

const char *sPtr = "schedule";  //补充一种指针式的字符数组
```

4. 哪几个运算符必须重载为成员函数 ()、[]、->、= 。

二、简答题（60分）

1． 什么是“else摇摆问题”，举例说明（10分）

答:

```
if (a>0)
   if(b>0)
     ...
else
   ...
```

这里的else应该与第二个if匹配而非第一个

2. 函数模板和函数重载的区别与联系（10分）

答：若一个函数的功能是对任意类型的数据作同样的处理，则将所处理的数据类型说明为参数，就可以把这个程序改写为函数模版。函数模版就是数据类型参数化的函数定义，代表的是一类函数。**其参数都是抽象的**。

用同一函数名定义多个函数，这些函数的参数个数、参数类型或参数顺序不同，这就是函数重载。**每个重载函数的参数是具体的**。但参数完全相同而返回值不同的函数不构成重载。

有一种特殊情况：重载模版函数。当编译器在处理重载模版函数的问题时，遵循的原则是：首选函数名、参数类型都匹配的具体函数，再找模版。

3.怎样区别虚函数和纯虚函数？两者都有什么作用（20分）

答:

```
virtual void fun();        //虚函数
virtual void fun() = 0;    //纯虚函数
```

虚函数用于基类与派生类的同名操作使其具有多态性，纯虚函数是用来定义抽象类的。

4.面向对象程序“接口与实现方法分离”，有什么优点（10分）

答：接口定义并标准化了人和系统等诸如此类事物彼此交互的方式。

成员函数的实现细节对客户代码是隐藏的(即信息隐藏)，使得程序员不会写出依赖类的实现细节的客户代码。

程序更容易修改，只要类的接口保持不变，类的实现的改变不会影响客户。

加快编译速度，提高了维护性，使得代码变得清晰

5.列出所有与字符串处理有关的头文件（10分）

答： string.h是C语言中字符串操作函数的头文件

cstring是C++对C语言中的strcpy之类的函数申明，包含cstring之后，就可以在程序中使用C语言风格的strcpy

之类的函数。

string是C++中string类模板的声明

CString是MFC中定义的字符串类，MFC中很多类及函数都是以CString为参数的

三、编程题（20分）

这篇博客中有参考代码与解析：

[2017] http://blog.csdn.net/qq_32925781/article/details/79423803

2018年

题型大变，没有简答了。

一、读程题(8*10分)

涉及字符大小比较、字符数组(指针式)、虚函数、继承等等。。。

二、填程序(25分)

1、判断闰年

2、读一行字符串到字符数组

3、求二维矩阵的鞍点

三、编程(45分)

1、输出1000以内的完数 格式是：完数=因子1+因子2+……因子n 比如说 6=1+2+3;

2、随机生成十个10到100000的数，转成字符串并输出至文件

3、设计一个employee类