阅读内容和要求：C++ Primer Plus （5-9章）阅读并总结关键语法点

总结：

第五章：

1. for循环的组成部分包括循环的初始化（如i=0）、循环测试部分（如i<5）、循环体部分（用{}括起最好）、循环更新部分（如i++），其中初始化、测试和更新操作构成了控制部分，由分号隔开，控制部分后面的语句是循环体。（循环在i=0时结束）
2. for循环是入口条件循环。这意味着在每轮循环之前都计算测试表达式的值，当测试表达式为false时将不会执行循环体。
3. <<运算符的优先级比表达式中使用的运算符高。
4. cout在显示bool值之前把它们转换成int，但是cout.setf(ios:: boolalpha)函数调用设置了一个标记，该标记命令cout显示true和false，而不是1和0。
5. 表达式的副作用是指判定表达式的值这种操作改变了内存中数据的值。
6. 可以在for循环的初始化部分声明变量。如：for (int i=0;i<5;i++)。
7. 检测不等通常比检测相等好，有时候变量的值不一定会等于检测值。
8. 递增运算符（++）和递减运算符（--）将循环计数加1或减1。前缀版本++a意思是先将a的值加1，然后使用新的值来计算表达式；后缀版本a++意思是使用a的当前值计算表达式，然后将a的值加1。对于类而言，前缀版本的效率比后缀版本高。
9. 前缀递增、前缀递减和解除引用运算符的优先级相同，以从右到左的方式结合。后缀递增和后缀递减的优先级相同，但比前缀运算符的优先级高，这两个运算符以从左到右的方式结合。如：

\*++pt从右到左结合，先将++应用于pt，然后将\*应用于被递增后的pt；

++\*pt，意味着对pt取得的值加1；

(\*pt)++，意味着首先对指针解除引用，得到值，在将值递增，指针指向位置不变；

\*pt++中++的优先级更高，所以先对指针递增，但是\*pt++的值仍是原来的，当语句执行结束后，pt的值才是递增后的新地址。

1. +=运算符将两个操作数相加，并将结果赋给左边的操作数。这意味着左边的操作数必须能够被赋值。同理有-=、\*=、/=和%=。
2. 逗号运算符的优先级最低。
3. 关系运算符的优先级比算术运算符低。
4. 不能用关系运算符比较字符串，但可以比较字符，因为字符实际上是整型。
5. strcmp()包含在头文件cstring中，用来比较字符串。
6. 可以对字符变量使用递增运算符和递减运算符。
7. while循环没有初始化和更新部分，只有测试条件和循环体。它也是一种入口条件循环。
8. 通常使用for循环来为循环计数，无法预知循环次数时，使用while循环。
9. 循环的指导原则：

* 指定循环终止的条件；
* 在首次测试之前初始化条件；
* 在条件被再次测试之前更新条件。

1. clock()返回程序开始执行后所用的系统时间，包含在头文件ctime中。ctime中定义了一个符号常量CLOCKS\_PER\_SEC，该常量等于每秒钟包含的系统时间单位数。所以将系统时间除以它可以得到秒数，或者秒数乘以它可以得到以系统时间单位为单位的时间。并且ctime将clock\_t作为clock()返回类型的别名，所以可以将变量声明为clock\_t，编译器将它转换为其他合适的类型。

20.c++有两种为类型建立别名的方式，一种是使用预处理器，如#define BYTE char，第二种是使用关键字typedef创建别名，如typedef char byte。

21.do while是出口条件循环，这种循环首先执行循环体，然后再判定测试表达式，条件为假则循环终止，条件为真则进行新一轮的执行和测试。它至少执行一次。

22.基于范围的for循环可以实现对容器的每一个元素执行相同的操作。

23.cin读取时忽略空格和换行符。cin的输入被缓冲，只有在用户按下回车键后，输入的内容才会发送给程序。cin.get(ch)读取输入中的下一个字符，包括空格，并赋给变量ch。

24.c++没有提供二维数组类型，用户可以创建每个元素本身都是数组的数组。

第六章：

1.使用if语句来实现选择是否执行某个操作。如果条件为真，if语句将引导程序执行语句或语句块，如果条件为假，程序将跳过这条语句或语句块。if else让程序决定执行两个语句或语句块中的哪一个。

2.‖运算符（OR）的优先级比关系运算符低。它也是个顺序点，因此先修改左侧的值，再对右侧的值进行判定。如果左侧的表达式为真，那么就不用判定右侧，整个表达式为真。&&（AND）同理，但如果左侧表达式为假，那么整个表达式为假。！表示取反，优先级高于所有的关系运算符和算数运算符。AND运算符的优先级高于OR运算符。

3.cctype中使用isalpha()检查字符是否为字母字符，使用isdigits()测试字符是否是数字字符，使用isspace()测试字符是否为空白，使用ispunct()测试字符是否是标点符号。

4.执行到switch语句时，程序跳到值所标记的那一行，要让程序执行完一组特定的语句后停止，必须使用break语句。

5.switch语句将int值和枚举量标签进行比较时，将枚举量提升为int。

6.break可以在switch语句或任何循环中使用，使得程序跳到switch或循环后面的语句处执行。continue语句用于循环中，让程序跳过循环体中余下的代码并开始新一轮循环。

第七章：

1. c++的返回值类型不能是数组，但是数组可以作为结构或者对象组成部分来返回。
2. 原型描述了函数到编译器的接口。函数原型是一条语句，必须以分号结束。
3. 在c++d的函数声明中，括号为空与在括号中使用关键字void是等效的。不指定参数列表时应该使用省略号。
4. 如果提供了一个不正确类型的参数，c++会自动转换成原型中指定的类型，条件是两者都是算术类型。
5. 在函数中声明的变量是局部变量，为该函数私有。
6. 形参和其他局部变量的主要区别是，形参从调用的函数里获得自己的值，其他变量是从函数中获得自己的值。
7. 为防止函数无意中修改数组的内容，可以在声明形参时使用关键字const。
8. 指针类型制定了列数，因此列数不作为独立的函数参数进行传递。
9. 将字符串作为参数传递时实际传递的是字符串第一个字符的地址，这意味着字符串函数原型应该将其表示字符串的形参声明为char \*类型。
10. 形参是指针而不是结构，因此需要使用间接成员运算符（->），而不是成员运算符（.）。
11. 获取函数的地址只要使用函数名即可，函数名后面不跟参数。
12. (\*pt)(int)意味着pt是一个指向函数的指针。

第八章：

1. &符号指示变量的地址，在c++中还可以用来声明引用。声明引用时必须将其初始化。
2. &&声明右值引用。
3. 方法self()设置各种格式化状态。所有这些设置都将一直保持不变，直到再次调用相应的方法设置它们。方法width()设置下一次输出操作使用的字段宽度，只在下一个值的显示有效，然后恢复默认设置。
4. 函数重载指的是可以有多个同名的函数，它们完成相同的工作，但是使用不同的参数列表。

第九章：

1. 组织程序的一种策略：

* 头文件：包含结构声明和使用这些结构的函数的原型；
* 源代码文件：包含与结构有关的函数的代码；
* 源代码文件：包含调用与结构相关的函数的代码。

1. 头文件常包含的内容：

* 函数原型；
* 使用#define或const定义的符号常量；
* 结构声明；
* 类声明；
* 模板声明；
* 内联函数。

1. 作用域为局部的变量只在定义它的代码块中可用，作用域为全局的变量在定义位置到文件结尾之间都可用。自动变量的作用域为局部，静态变量的作用域取决于它如何被定义。
2. 动态内存的分配和释放是用new和delete进行的。
3. 静态持续变量：

* 外部链接性：在代码块的外面声明它；（外部变量或全局变量）
* 内部链接性：在代码块的外面声明它，并使用static限定词；
* 没有链接性：在代码块的内部声明它，并使用static限定词。

1. 如果要在多个文件使用外部变量，只需在一个文件中包含该变量的定义，但在使用该变量的其他所有文件中必须使用关键字extern声明它。
2. 声明区域是可以在其中进行声明的区域，潜在作用域从声明点开始，到其声明区域的结尾。
3. 名称空间可以是全局的，也可以位于另一个名称空间中，但不能位于代码块中，因此，在默认情况下，在名称空间中声明的名称的链接性为外部的，除非引用了常量。
4. 名称空间的一些指导原则：

* 使用在已命名的名称空间中声明的变量，而不是使用外部全局变量或者静态全局变量；
* 如果开发了一个函数库或类库，将其放在一个名称空间中；
* 仅将编译指令using作为一种将旧代码转换为使用名称空间的权宜之计；
* 不要在头文件中使用using编译指令；
* 导入名称时，首选使用作用域解析运算符或using声明的方法；
* 对于using声明，首选将其作用域设置为局部而不是全局。