第五章：

循环结构：

1.while循环：在循环开始之前先判断条件，只有条件为真时才执行循环体。

2. do-while循环：先执行循环体，然后再判断条件，至少会执行一次循环体。

3.for循环：在循环开始之前初始化变量，然后判断条件，执行循环体，最后更新变量。

关系表达式：

1.关系运算符用于比较两个值的大小关系，

2.关系表达式的结果是一个布尔值，即`true`或`false`。

3.关系表达式可以用于条件判断，控制循环的执行。

逻辑表达式：

1.逻辑运算符用于组合多个关系表达式的结果，例如逻辑与“&&”、逻辑或“||”、逻辑非“!”。

2.逻辑表达式的结果也是一个布尔值。

3.逻辑运算符有短路求值的特点，即如果能够确定整个表达式的结果，就不再计算后面的表达式。

循环控制语句：

1.break`语句用于立即退出当前循环。

2.continue`语句用于跳过当前循环的剩余部分，进入下一次循环。

第六章

1. 分支语句：

if语句：根据条件的真假执行不同的代码块。

switch语句：根据表达式的值选择不同的分支进行执行。

2. 条件运算符：

条件运算符（三元运算符）`?:`可以根据条件的真假选择不同的值进行赋值。

3.逻辑运算符：

逻辑与`&&`：当所有条件都为真时，结果为真。

逻辑或`||`：当至少有一个条件为真时，结果为真。

逻辑非`!`：对条件取反

4. 嵌套和多重选择：

可以在分支语句中嵌套其他分支语句，实现多重选择的逻辑。

第七章

1. 函数的定义和声明：

函数是一段完成特定任务的代码块，可以通过函数名来调用执行。

函数的定义包括函数的返回类型、函数名、参数列表和函数体。

函数的声明是指在使用函数之前提前声明函数的原型，以便编译器知道函数的存在。

2. 函数的参数传递：

函数可以通过参数来接收外部传入的数据，参数可以是基本类型、数组、结构体等。

参数可以按值传递、按引用传递或按指针传递。

1. 函数的返回值：

函数可以有返回值，返回值的类型在函数定义时指定。

使用`return`语句将函数的结果返回给调用者。

4. 函数的重载：

函数重载是指在同一个作用域内定义多个同名函数，但它们的参数列表不同。

通过函数重载，可以根据不同的参数类型和个数来选择调用不同的函数。

5. 默认参数：

函数可以设置默认参数，当调用函数时没有提供相应的参数时，将使用默认值。

1. 内联函数：

内联函数是指在函数定义前加上`inline`关键字，告诉编译器将函数的代码插入到调用处，而不是通过函数调用的方式执行。

第八章

1.递归函数：

递归函数是指在函数的定义中调用函数本身的情况。

递归函数需要有一个终止条件，以避免无限递归。

递归函数可以解决一些问题，如阶乘、斐波那契数列等。

2. 局部变量、静态局部变量和全局变量：

局部变量是在函数内部定义的变量，其作用域仅限于函数内部。

静态局部变量是在函数内部定义的静态变量，其作用域仅限于函数内部，但其生命周期会延长到整个程序运行期间。

全局变量是在函数外部定义的变量，其作用域为整个程序，可以被多个函数访问。

3. 函数指针：

函数指针是指向函数的指针变量，可以用于调用函数。

函数指针的声明需要指定函数的返回类型和参数列表。

函数指针可以作为函数的参数，也可以作为函数的返回值。

4. 函数对象：

函数对象是指重载了函数调用运算符`()`的对象，可以像函数一样被调用。

函数对象可以保存状态，可以作为函数的参数或返回值。

第九章

1. 数组：

数组是一组相同类型的数据元素的集合，可以通过索引访问和操作数组中的元素。

数组的大小在定义时确定，且不可改变。

数组的索引从0开始，最后一个元素的索引为数组大小减一。

2. 字符串：

字符串是由字符组成的序列，以空字符`\0`结尾。

C++中的字符串可以使用字符数组或`string`类来表示和操作。

字符串的常见操作包括拼接、比较、查找等。

3. 动态内存分配：

动态内存分配是指在程序运行时根据需要分配和释放内存空间。

使用`new`运算符可以在堆上分配内存空间，使用`delete`运算符释放内存空间。

动态内存分配可以用于创建数组和对象，以及处理变长数据。

4. 多维数组：

多维数组是指包含多个维度的数组，可以通过多个索引来访问和操作数组中的元素。

多维数组可以是二维、三维或更高维度的。

5. 字符串和C-风格字符串的转换：

可以使用C++标准库中的函数来进行字符串和C-风格字符串之间的转换，如“strcpy”、“strlen”、“to\_string”等。