**注意区分：**

**=是赋值！**

**==才是相等！**

**不能连续大于/小于！！！**

**全局替换快捷键：ctrl+f (ind)**

**代码的组成结构**

**Identifiers: 标识符：下划线或字母起始**

**Delimiters/separators: 分隔符**

**包括：colon:冒号**

**semicolon： 分号**

**parentheses、brackets:括号**

**Literals: 常量**

**Comments:注释 用// /\* \*/ 表示**

**Preprocessing directives: 预处理命令**

**比如#include <iostream>**

**以#号开头，因为不是c++命令，所以不用分号结尾。**

**预处理是在编译器编译之前进行的操作。预处理过程扫描源代码，对其进行初步的转换，产生新的源代码提供给编译器。**

**所以预处理过程先于编译器对源代码进行处理。**

**等号的性质：**

**Int sum=0 ：initialization**

**Sum=0: assignment.**

**数据类型所占字节的数量：**

**Char/bool: 一个字节（byte）= 8个bit**

**Short: 两个字节**

**Int/long: 四个字节**

**Long long : 八个字节**

**Float: 四个字节（1+8+23）**

**Double: 八个字节（1+11+52）**

**指针与引用：**

|  |
| --- |
| int a **=** 10**;**  int **&**b **=** a**;**//&出现在声明处表引用，将b记作a的别名。  int**\*** aPtr **=** **&**a**;**//\*是指针，int 表示指针指向的空间类型为整数，此处为a的地址。  b **=** 5**;**  cout **<<** a **<<** **\***aPtr **<<** **\*(&**b**)** **<<** endl**;**//&此时是取地址符，\*用来间接访问。输出3个5.  **\***aPtr **=** 7**;**//通过地址进行间接访问，将a的值修改为7.  cout **<<** a **<<** **\***aPtr **<<** **\*(&**b**)** **<<** endl**;**//输出三个7. |

**值得一提的是，&&为右值引用。也就是可以实现**

|  |
| --- |
| double**&&** r1 **=** 3.14**;**  r1 **=** 2**;** |

**内置数组中的几种循环方式**

|  |
| --- |
| const int asize **=** 5**;**  int aArray**[**asize**];**//内置数组在声明外不可用等号直接对名称赋值。可以理解为常指针。  int**\*** pArray **=** aArray**;**//指向首地址值  **for** **(**int**&** x **:** aArray**)**//范围循环,&符号必要，优点是不会越界  **{**  x **=** 4**;**  **}**  **for** **(**int x **:** aArray**)**  **{**  cout **<<** x **<<** " "**;**  **}**  **for** **(**int i **=** 0**;** i **<** asize**;** i**++)**  **{**  aArray**[**i**]** **+=** i**;**  **}**  **for** **(**int i **=** 0**;** i **<** asize**;** i**++)**  **{**  cout **<<** **\*(**aArray **+** i**);**//加号表示偏移；可以换成pArray  **}**  **for** **(**int i **=** 0**;** i **<** asize**;** i**++)**  **{**  cout **<<** pArray**[**i**]** **<<** " "**;**//中括号其实同样是偏移  **}** |

**字符指针和字符数组之间的关系**

|  |
| --- |
| #include<string>  int main**()**  **{**  int b **=** 1**;**  int**\*** a **=** **&**b**;**  cout **<<** a **<<** endl**;**//输出的是地址值00000063E62FF474。  cout **<<** **\***a **<<** endl**;**//此时输出的就是1了。  const char**\*** aPtr **=** "student"**;**//const是因为右侧是常量，表示指针只有只读的权利,不加会报错  cout **<<** aPtr **<<** "" **<<** " " **<<** **\***aPtr **<<** " " **<<** aPtr **+** 3 **<<** " "**<<** **\*** **(**aPtr **+** 3**)** **<<** endl**;**  //由于字符指针常用，对其功能进行了拓展。aPtr输出整个字符串而不是地址.  // \*aPtr由于指针指向首地址，输出的是s.当然，也可以进行偏移。  //输出student s dent d  char bArray**[**10**];**  cin **>>** bArray**;**//相应地，输入输出功能也进行了拓展，逐个元素赋值。  cout **<<** bArray **<<** endl**;**  char c**[**10**]** **=** **{** 's'**,**'t'**,**'u'**,**'d'**,**'e'**,**'n'**,**'t'**,**'\0' **};**  //char c[10]="student;一定要加上结束符"\0",否则有可能报错  string s**;**  getline**(**cin**,** s**);**//按换行进行输入，由于此时上次cin的换行符仍在缓冲区中，故需要两次。  getline**(**cin**,** s**);**  cout **<<** s**;**  **}** |

**字符数组不可加（如循环时++A）**

**但是字符指针是可以的(但是要记得加const!)**

|  |
| --- |
| char ce**[**10**]** **=** "hello"**;**  const char**\*** p **=** ce**;**  p **=** "abc"**;**  cout **<<** p**;**//输出结果为"abc" |

**此处更改了\*p指向，并没有实际上改变ce！合理。**

**实参为数组，传递的是地址。如下例：**

**此处两个指针一有const一无const,细节。**

**图片包含 表格

描述已自动生成**

**逆向输出：**

**图片包含 文本

描述已自动生成**

**统计元素数量**

**图形用户界面, 应用程序

描述已自动生成**

**计算元素的个数**

**文本

描述已自动生成**

**类图：类的名字 | 数据成员（前面有+表示public） | 成员函数**

**图示

描述已自动生成using namespace std : 名空间“standard”**

**Cout: character out**

**escape sequence:**

**\n, \t ,\r ,\a, \\ ,\' ,\"**

**an identifier must not begin with a digit.**

**the integer must not exceed 2^31-1**

**exponentiation（指数）：pow（x,y)**

**类——面向对象**

**函数用Pascalcase；**

**变量用camelCase**

**形参parameter**

**实参argument**

**局部变量local variable**

**公域和私域定义一组形参/实参，实参传递到形参上（71页L33)**

**在调用函数时实参传递给形参**

**cin遇到空格就无法提取了，必须用getline(cin,变量名）**

**P75:局部变量nameOfCourse作为实际参数传递给形参name再赋值给courseName**

**return:将coursename值赋值给匿名对象（obj），即"getCourseName()"**

**除了引用类型之外，匿名对象都要开辟存储空间。**

**constructor构造函数（自动执行，设置初始值）**

**this指针指向当前变量（分别为gradebook1和gradebook2)**

|  |
| --- |
| Gradebook **(**string name**)**  **{**  **this->**coursename**=**name  **}**  GradeBook gradeBook1**(**“CS101 Introduction to C**++** Programming”**);**  GradeBook gradeBook2**(**“CS102 Data Structures in C**++**" ); |

**string写在前面和后面的区别是什么？**

**写在括号里是形参属性，写在前面是函数的属性（数据成员）**

**例如string getCourseName()就是匿名对象，性质为string**

**为什么要特地设定一个name变量？**

**类中的function必须有描述**

**头文件：.h 目的是为了class等代码的重用性**

**头文件中不能写using namespace std**

**1.include要重新包括一遍**

**2.将函数用分号代替，其他部分剪切到新建的cpp文件上；**

**3.新建cpp要包含头文件的定义（#include". h")不是<>哦！**

**validates the length of data member**

**membername.substr(0,25)**

**从第0个字符后开始取**

**Control structures:顺寻结构、选择结构（if,if/else,switch)、循环结构**

**void：不返回任何信息的函数类型**

**int main()的含义：**

**表示有某个int类型的返回值。（典型地，如设定的0，默认表示函数正常运行）**

|  |
| --- |
| switch语句  **switch()**  **{**  **case** where1**:** statement 1**;**  **break;**  **case** where2**:** statement 2**;**  **break;**  **default:** **default** statement**;**  **break;** //缺省情况  **}** |

**Switch括号里一定要是integral value,即具有整型值的表达式（整数，字符）！**

**1个case只会被匹配一次！如果没有break,会一直向下执行直到switch结束。**

**布尔值：bool（true:非零 / false:0）**

**三元运算：？：**

**int grade=90;**

**cout << ((grade > 60) ? ("passed!") : ("failed!"));**

**优先级: 算术、类、逻辑、条件、赋值、逗号**

**else与最近的if匹配；**

|  |
| --- |
| #include<iomanip>  double a **=** 0.012**;**  cout**<<**setprecision**(**位数**)<<**fixed**<<**a**;** |

**x++和++x的区别：后者是“加了再说”，先将x+1赋给x；**

|  |
| --- |
| //判断double型是否为整数：  #include <math.h>  **...**  const double EPS 1e-6  double a**;**  **...**  **if(**a **-** floor**(**a**)** **<** EPS**)**  //则为整数 |

|  |
| --- |
| //首次应用循环的嵌套（nesting):  int a**;**  **while** **(**cin **>>** a**)**  **{**  **for** **(**int i **=** 0**;** i **<** a**;** i**++)**  **{**  **for** **(**int j **=** 0**;** j **<** a**;** j**++)**  **{**  **if** **(**i **==** 0 **||** i **==** a **-** 1 **||** j **==** 0 **||** j **==** a **-** 1**)**  cout **<<** "\* "**;**  **else**  cout **<<** " "**;**  **}**  cout **<<** endl**;**  **}**  **}** |

**Counter controlled loop: counter 是必要的。**

**For 循环的内部原理：**

**表达式一只执行一次；**

**表达式二执行之后，执行循环体；最后执行表达式三；再回表达式二。**

**逗号表达式的值为最右端表达式的值；**

**第一步初始化可省略（提到前面）；**

**第二步省略则视为永真式；可用if判断+break跳出循环；**

**第三步省略：空语句。**

**循环体省略：表达式二循环**

|  |
| --- |
| int sum **=** 0**;**  **for** **(**int i **=** 1**;** i **<=** 100**,** sum **+=** i**++;)**  **{**  **}**  cout **<<** sum**;** |

**输出结果为0。**

**左侧参数在无副作用时被忽略，因此一直累加。**

**i超过2^31-1符号变为负。**

**某时刻sum+i=0,退出。**

**for (int i = 1; sum += i++, i <= 100;)**

**输出结果为5050。i<=100为false之后返回0，程序退出。**

**include<cmath>**

**pow(x,y)**

**注：iostream已经包含在cmath中了。**

**setw(int n)是c++中在输出操作中使用的字段宽度设置，设置输出的域宽，n表示字段宽度。只对紧接着的输出（两个<<之间）有效，紧接着的输出结束后又变回默认的域宽。**

**当后面紧跟着的输出字段长度小于n的时候，在该字段前面用空格补齐；当输出字段长度大于n时，全部整体输出。**

|  |
| --- |
| #include <iostream>  #include <iomanip>  **using** **namespace** std**;**  int main**()**  **{**  cout **<<** setw**(**3**)** **<<** 1 **<<** 1234 **<<** endl**;**  cout **<<** 1234 **<<** setw**(**7**)** **<<** 1234 **<<** 12345 **<<** endl**;**  cout **<<** 1234 **<<** setw**(**3**)** **<<** 12345 **<<** endl**;**  cout **<<** 1234 **<<** setw**(**4**)** **<<** 123456 **<<** 12 **<<** setw**(**5**)** **<<** 123 **<<** endl**;**  **return** 0**;**  **}**  //输出结果  1234  1234 123412345  123412345  123412345612 123 |

**setprecision fixed 直到被更改为止不会改变。**

**编译错误（compilation error）语法错误（syntax error)编译器无法通过**

|  |
| --- |
| **do**  **{**  **}while();** |

**^z与EOF:**

**Sentinel-controlled repetition,一般对应while 和 do…while…**

**哨兵值EOF：c++中表示-1，但直接输入并不能视为EOF。**

**windows系统上，需要按组合键ctrl+z。**

**break与continue语句。;**

**continue为终止本次循环,即跳过循环体中本次循环的剩余部分直接执行下一次循环。**

**break为直接跳出循环。**

**bool类型变量如果要输出true/false,必须要加上boolalpha**

|  |
| --- |
| bool a **=** 0**;**  cout **<<** boolalpha **<<** a**<<**endl**;**//输出为false  cout **<<** noboolalpha **<<** a **<<** endl**;**//输出为0 |

**函数签名只包括函数名和参数列表，函数声明还包括返回类型。**

**函数头是不包括分号的。**

**Pass-by-value/reference**

**变量前如果加了static，是静态变量，只会初始化一次。**

**默认参数一定要在函数的最末端。注意函数重载会出现问题。**

|  |
| --- |
| void ferror1**(**int x **,**int y**=**0**);**  void ferror1**(**int x**);**  ferror1**(**3**);** |

**此时不确定call的是谁。**

**引用类型不开辟存储空间。**

**int & ref\_z = z ;**

**(ref\_z与z共用空间）**

**函数原型中可以没有名字，因为形参只关注类型。**

**fabs为绝对值（需要包含cmath）**

**switch 如果其中一条语句之后没有break，则以下的语句（除了default）都将为真。**

**class结束之后一定要分号！**

**if和**

**else如果要对应，一定要写括号！**

**if (a % 2 == 0)cout << 1;**

**cout << endl;**

**else cout << 2;**

**如这种情况，if else就无法完全对应！**

**函数原型变量名可以省略，但是定义不可省略！**

**unsigned int box(int length=1, int width=1,int height=1)**

**要从最右端开始省略。**

**main函数之外的是全局变量，否则是局部变量。**

**若有同名，则优先调取局部变量。**

**左侧加入::则优先调取全局变量。**

**Function Overloading 函数重载**

**目的：创建同名的相似函数。（例如整形相加和double相加）。**

**int square(int a)**

**double square(double a)**

**返回函数的类型就是匿名对象的类型。**

**签名：函数名/形参的类型**

**函数模板：**

|  |
| --- |
| cout**<<**square**(**7**)<<**square**(**7.5**);**  template **<**typename T**>**  T maximum**(**T value 1**,** T value 2**,** T value 3**)** |

**这样就可以对于double，int都适用了。**

**举例：**

|  |
| --- |
| template**<**typename A**>**  A squareroot**(**A a1**)**  **{**  **return** sqrt**(**a1**);**  **}**  int main**()**  **{**  cout **<<** squareroot**(**1**)** **<<** " " **<<** squareroot**(**1.5**);**  **}** |

**Recursion 递归函数：自己call自己的函数。**

**指针**

**int a=7;**

**int \*aPtr=&a;**

**\*箭指**

**pass-by-value传值**

**内建数组**

**built-in arrays**

**type arrayName[arraySize]**

**若定义为整形，未定义的值为0；**

**逻辑型，false;**

**指针，空指针；**

**构造函数，缺省；**

**int ap[]其实是整形指针**

**（const）变量和(const)指针**

**其中，const类型的标识符不能被赋值，但可以被获取地址。**

**Const指针：常指针，指向的内容也不能改。**

**int \*p=&a,并非是将a的地址值赋给\*p,而是p指针！**

**指针、整形 4个字节；**

**double 8个字节**

**string 1个字节**

**sizeof运算符**

**直接输出字符指针或字符数组名都表示字符串**

**在读取到\0之前不会退出，所以一定记得加\*号！**

**Array是对象。**

**静态/动态数组**

**Sort(colors.begin(),colors.end())排序**

**Bool found= binary\_search(colors.begin(),colors.end(),”indigo”)检索**

**cout<<(found? :)**

**检索一个元素之前，一定要先排序。**

**Vector**

**输入函数：inputvector**

**Push\_back:在末尾加入一个元素**

**Pop\_back:删除最后一个元素**

**Erase：删除指定位置元素。**

**复制构造函数**

**try catch异常：**

**at函数可以继续处理接下来的命令。**

**pow的返回类型是double! 如果当作int用乘除运算中会出现问题！**

**七八章作业不能用built-in arrays!**

**函数原型相当于函数声明，包括函数类型、函数名、形参列表（其中形参名可以省略），且不需要函数体。**