ISO C++98/03关键字共63个，此处严格按标准原文排版：

asm	do	if	return	typedef
auto	double	inline	short	typeid
bool	dynamic_cast	int	signed	typename
break	else	long	sizeof	union
case	enum	mutable	static	unsigned
catch	explicit	namespace	static_cast	using
char	export	new	struct	virtual
class	extern	operator	switch	void
const	false	private	template	volatile
const_cast	float	protected	this	wchar_t
continue	for	public	throw	while
default	friend	register	true	
delete	goto	reinterpret_cast	try	

1. asm

asm (指令字符串)：允许在 C++ 程序中嵌入汇编代码。

2. auto

auto（自动，automatic）是存储类型标识符，表明变量"自动"具有本地范围，块范围的变量声明（如for循环体内的变量声明）默认为auto存储类型。

3. bool

bool（布尔）类型，C++ 中的基本数据结构，其值可选为 true（真）或者 false（假）。C++ 中的 bool 类型可以和 int 混用，具体来说就是 0 代表 false，非 0 代表 true。bool 类型常用于条件判断和函数返回值。

4. break

break（中断、跳出），用在switch语句或者循环语句中。程序遇到 break 后，即跳过该程序段，继续后面的语句执行。

5. case

用于 switch 语句中，用于判断不同的条件类型。

6. catch

catch 和 try 语句一起用于异常处理。

7. char

char（字符，character）类型，C++ 中的基本数据结构，其值一般为 0~255 的 int。这 256 个字符对应着 256 个 ASCII 码。char 类型的数据需要用单引号 ' 括起来。

8.class

class（类）是 C++ 面向对象设计的基础。使用 class 关键字声明一个类。

9. const

const（常量的，constant）所修饰的对象或变量不能被改变，修饰函数时，该函数不能改变在该函数外面声明的变量也不能调用任何非const函数。在函数的声明与定义时都要加上const，放在函数参数列表的最后一个括号后。在 C++ 中，用 const 声明一个变量，意味着该变量就是一个带类型的常量，可以代替 #define，且比 #define 多一个类型信息，且它执行内链接，可放在头文件中声明；但在 C 中，其声明则必须放在源文件（即 .c 文件）中，在 C 中 const 声明一个变量，除了不能改变其值外，它仍是一具变量。如:

```
const double pi(3.14159);
或
const double pi = 3.14159;
```

10. const_cast

```
const_cast<type_id> (expression)
```

该运算符用来修改类型的 const 或 volatile 属性。除了 const 或 volatile 修饰之外， type_id 和 expression 的类型是一样的。常量指针被转化成非常量指针，并且仍然指向原来的对象；常量引用被转换成非常量引用，并且仍然指向原来的对象；常量对象被转换成非常量对象。

11. continue

continue（继续）关键字用于循环结构。它使程序跳过代码段后部的部分，与 break 不同的是，continue 不是进入代码段后的部分执行，而是重新开始新的循环。因而它是"继续循环"之意，不是

break（跳出）。

12. default

default（默认、缺省）用于 switch 语句。当 switch 所有的 case 都不满足时，将进入 default 执行。default 只能放在 switch 语句所有的 case 之后，并且是可选的。

13. delete

delete（删除）释放程序动态申请的内存空间。delete 后面通常是一个指针或者数组 []，并且只能 delete 通过 new 关键字申请的指针，否则会发生段错误。

14. do

do-while是一类循环结构。与while循环不同，do-while循环保证至少要进入循环体一次。

15. double

double（双精度）类型，C++ 中的基本数据结构，以双精度形式存储一个浮点数。

16. dynamic_cast

dynamic_cast（动态转换），允许在运行时刻进行类型转换，从而使程序能够在一个类层次结构安全地转换类型。dynamic_cast 提供了两种转换方式，把基类指针转换成派生类指针，或者把指向基类的左值转换成派生类的引用。

17. else

else 紧跟在 if 后面，用于对 if 不成立的情况的选择。

18. enum

enum（枚举）类型，给出一系列固定的值，只能在这里面进行选择一个。

19. explicit

explicit（显式的）的作用是"禁止单参数构造函数"被用于自动型别转换，其中比较典型的例子就是容器类型。在这种类型的构造函数中你可以将初始长度作为参数传递给构造函数。

20. export

为了访问其他编译单元（如另一代码文件）中的变量或对象，对普通类型（包括基本数据类、结构和类），可以利用关键字 extern，来使用这些变量或对象时；但是对模板类型，则必须在定义这些模板类对象和模板函数时，使用标准 C++ 新增加的关键字 export（导出）。

21. extern

extern（外部的）声明变量或函数为外部链接，即该变量或函数名在其它文件中可见。被其修饰的变量（外部变量）是静态分配空间的，即程序开始时分配，结束时释放。用其声明的变量或函数应该在别的文件或同一文件的其它地方定义（实现）。在文件内声明一个变量或函数默认为可被外部使用。在 C++ 中，还可用来指定使用另一语言进行链接，这时需要与特定的转换符一起使用。目前仅支持 C 转换标记，来支持 C 编译器链接。使用这种情况有两种形式：

```
extern "C" 声明语句
extern "C" { 声明语句块 }
```

22. false

false（假的），C++ 的基本数据结构 bool 类型的值之一。等同于 int 的 0 值。

23. float

float（浮点数），C++ 中的基本数据结构，精度小于 double。

24. for

for 是 C++ 中的循环结构之一。

25. friend

friend（友元）声明友元关系。友元可以访问与其有 friend 关系的类中的 private/protected 成员，通过友元直接访问类中的 private/protected 成员的主要目的是提高效率。友元包括友元函数和友元类。

26. goto

goto（转到），用于无条件跳转到某一标号处开始执行。

27. if

if（如果），C++ 中的条件语句之一，可以根据后面的 bool 类型的值选择进入一个分支执行。

28. inline

inline（内联）函数的定义将在编译时在调用处展开。inline 函数一般由短小的语句组成，可以提高程序效率。

29. int

int（整型，integer），C++ 中的基本数据结构，用于表示整数，精度小于 long。

30. long

long（长整型，long integer），C++ 中的基本数据结构，用于表示长整数。

31. mutable

mutable（易变的）是 C++ 中一个不常用的关键字。只能用于类的非静态和非常量数据成员。由于一个对象的状态由该对象的非静态数据成员决定，所以随着数据成员的改变，对像的状态也会随之发生变化。如果一个类的成员函数被声明为 const 类型，表示该函数不会改变对象的状态，也就是该函数不会修改类的非静态数据成员。但是有些时候需要在该类函数中对类的数据成员进行赋值，这个时候就需要用到 mutable 关键字。

32. namespace

namespace（命名空间）用于在逻辑上组织类，是一种比类大的结构。

33. new

new（新建）用于新建一个对象。new 运算符总是返回一个指针。由 new 创建

34. operator

operator（操作符）用于操作符重载。这是 C++ 中的一种特殊的函数。

35. private

private（私有的），C++ 中的访问控制符。被标明为 private 的字段只能在本类以及友元中访问。

36. protected

protected（受保护的），C++ 中的访问控制符。被标明为 protected 的字段只能在本类以及其继承类和友元中访问。

37. public

public（公有的），C++ 中的访问控制符。被标明为 public 的字段可以在任何类

38.register

register（寄存器）声明的变量称着寄存器变量，在可能的情况下会直接存放在机器的寄存器中；但对 32 位编译器不起作用，当 global optimizations（全局优化）开的时候，它会做出选择是否放在自己的寄存器中；不过其它与 register 关键字有关的其它符号都对32位编译器有效。

39. reinterpret_cast

用法：

```
reinpreter_cast<type-id> (expression)
```

type-id 必须是一个指针、引用、算术类型、函数指针或者成员指针。它可以把一个指针转换成一个整数，也可以把一个整数转换成一个指针（先把一个指针转换成一个整数，在把该整数转换成原类型的指针，还可以得到原先的指针值）。

40. return

return（返回）用于在函数中返回值。程序在执行到 return 语句后立即返回，return 后面的语句无法执行到。

41. short

short（短整型，short integer），C++ 中的基本数据结构，用于表示整数，精度小于 int。

42. signed

signed（有符号），表明该类型是有符号数，和 unsigned 相反。数字类型（整型和浮点型）都可以用 signed 修饰。但默认就是 signed，所以一般不会显式使用。

43. sizeof

由于 C++ 每种类型的大小都是由编译器自行决定的，为了增加可移植性，可以用 sizeof 运算符获得该数据类型占用的字节数。

44. static

static（静态的）静态变量作用范围在一个文件内，程序开始时分配空间，结束时释放空间，默认初始化为 0，使用时可改变其值。静态变量或静态函数，只有本文件内的代码才可访问它，它的名字（变量名或函数名）在其它文件中不可见。因此也称为"文件作用域"。在 C++ 类的成员变量被声明为 static（称为静态成员变量），意味着它被该类的所有实例所共享，也就是说当某个类的实例修改了该静态成员变量，其修改值为该类的其它所有实例所见；而类的静态成员函数也只能访问静态成员（变量或函数）。类的静态成员变量必须在声明它的文件范围内进行初始化才能使用，private 类型的也不例外。

45. static_cast

用法：

```
static_cast < type-id > ( expression )
```

该运算符把 expression 转换为 type-id 类型，但没有运行时类型检查来保证转换的安全性。它主要有如下几种用法：

- ① 用于类层次结构中基类和子类之间指针或引用的转换。进行上行转换（把子类的指针或引用转换成基类表示）

是安全的；进行下行转换（把基类指针或引用转换成子类表示）时，由于没有动态类型检查，所以是不安全的。

- ② 用于基本数据类型之间的转换，如把 int 转换成 char，把 int 转换成 enum。这种转换的安全性也要开发人员来保证。
- ③ 把空指针转换成目标类型的空指针。
- ④ 把任何类型的表达式转换成void类?

注意 static_cast 不能转换掉 expression 的 const、volitale、或者 __unaligned 属性。

46. struct

struct（结构）类型，类似于 class 关键字，与 C 语言兼容（class 关键字是不与 C 语言兼容的），可以实现面向对象程序设计。

47. switch

switch（转换）类似于 if-else-if 语句，是一种多分枝语句。它提供了一种简洁的书写，并且能够生成效率更好的代码。但是，switch 后面的判断只能是int（char也可以，但char本质上也是一种int类型）。switch 语句最后的 default 分支是可选的。

48. template

template（模板），C++ 中泛型机制的实现。

49. this

this 返回调用者本身的指针。

50. throw

throw（抛出）用于实现 C++ 的异常处理机制，可以通过 throw 关键字"抛出"一个异常。

51. true

true（真的），C++ 的基本数据结构 bool 类型的值之一。等同于 int 的非 0 值。

52. try

try（尝试）用于实现 C++ 的异常处理机制。可以在 try 中调用可能抛出异常的函数，然后在 try 后面的 catch 中捕获并进行处理。

53. typedef

typedef（类型定义，type define），其格式为：

```
typedef  类型 定义名;
```

类型说明定义了一个数据类型的新名字而不是定义一种新的数据类型。定义名表示这个类型的新名字。

54. typeid

指出指针或引用指向的对象的实际派生类型。

55. typename

typename（类型名字）关键字告诉编译器把一个特殊的名字解释成一个类型。在下列情况下必须对一个 name 使用 typename 关键字：

1. 一个唯一的name（可以作为类型理解），它嵌套在另一个类型中的。
2. 依赖于一个模板参数，就是说：模板参数在某种程度上包含这个name。当模板参数使编译器在指认一个类型时产生了误解。

56. union

union（联合），类似于 enum。不同的是 enum 实质上是 int 类型的，而 union 可以用于所有类型，并且其占用空间是随着实际类型大小变化的。

57. unsigned

unsigned（无符号），表明该类型是无符号数，和 signed 相反。

58. using

表明使用 namespace。

59. virtual

virtual（虚的），C++ 中用来实现多态机制。

60. void

void（空的），可以作为函数返回值，表明不返回任何数据；可以作为参数，表明没有参数传入（C++中不是必须的）；可以作为指针使用。

61. volatile

volatile（不稳定的）限定一个对象可被外部进程（操作系统、硬件或并发线程等）改变，声明时的语法如下：

```
int volatile nVint;
```

这样的声明是不能达到最高效的，因为它们的值随时会改变，系统在需要时会经常读写这个对象的值。因此常用于像中断处理程序之类的异步进程进行内存单元访问。

62. wchar_t

wchar_t 是宽字符类型，每个 wchar_t 类型占 2 个字节，16 位宽。汉字的表示就要用到 wchar_t。