**Перечисления**

Средство C++ enum представляет собой альтернативный по отношению к const способ создания символических констант. Он также позволяет определять новые типы, но в очень ограниченной манере. Синтаксис enum подобен синтаксису

структур. Например, рассмотрим следующий оператор:

*enum spectrum {red, orange, yellow, green, blue, violet, indigo, ultraviolet};*

**Этот оператор делает две вещи.**

• Объявляет имя нового типа — spectrum; при этом spectrum называется перечислением, почти так же, как переменная struct называется структурой.

• Устанавливает red, orange, yellow и т.д. в качестве символических констант для целочисленных значений 0-7. Эти константы называются перечислителями.

По умолчанию перечислителям присваиваются целочисленные значения, начиная с 0 для первого из них, 1 — для второго и т.д. Это правило по умолчанию можно переопределить, явно присваивая целочисленные значения.

Имя перечисления можно использовать для объявления переменной с этим типом перечисления:

*spectrum band; // band — переменная типа spectrum*

**Единственными допустимыми значениями**, которые можно присвоить переменной типа перечисления без необходимости приведения типов, являются значения, указанные в определении этого перечисления.

*band = blue; // правильно, blue - перечислитель*

*band = 2000; // неправильно, 2000 — не перечислитель*

*spectrum* ограничена только восемью допустимыми значениями.

Для перечислений определена только операция присваивания. В частности, арифметические операции не предусмотрены:

*band = orange; // правильно*

*++band; // неправильно (операция ++ обсуждается в главе 5)*

*band = orange + red; // неправильно, но довольно хитро*

**Установка значений перечислителей**

Конкретные значения элементов перечислений можно устанавливать явно посредством операция присваивания:

*enum bits {one = 1, two = 2, four = 4, eight = 8};*

**Присваиваемые значения должны быть целочисленными**. Можно также явно устанавливать только некоторые из перечислителей:

*enum bigstep{first, second = 100, third};*

В этом случае first получает значение 0 по умолчанию. Каждый последующий неинициализированный перечислитель увеличивается на единицу по сравнению с предыдущим. Поэтому third имеет значение 101.

**Допускается указывать одно й тоже значение для нескольких перечислителей:**

*enum {zero, null = 0, one, numero\_uno = 1};*

Здесь zero и null имеют значение 0, a one и numero\_uno — значение 1.

**Диапазоны значений перечислителей**

Каждое перечисление имеет диапазону и с помощью приведения к типу переменной перечисления можно присвоить любое целочисленное значение в пределах этого диапазона, даже если данное значение не равно ни одному из перечислителей.

*enum bits {one = 1, two = 2, four = 4, eight = 8};*

*bits myflag;*

В таком случае показанный ниже оператор является допустимым:

*myflag = bits (6); // правильно, потому что 6 находится в пределах диапазона*