Понятие типа данных

Тип данных определяет:
- внутреннее представление данных в памяти компьютера;
- множество значений, которые могут принимать величины этого типа;
- операции и функции, которые можно применять к величинам этого типа.

Все типы языка C++ можно разделить на основные и составные. В языке C++ определено шесть основных типов данных для представления целых, вещественных, символьных и логических величин. На основе этих типов программист может вводить описание составных типов. К ним относятся массивы, перечисления, функции, структуры, ссылки, указатели, объединения и классы.

Основные типы данных

Основные (стандартные) типы данных часто называют арифметическими, поскольку их можно использовать в арифметических операциях. Для описания основных типов определены следующие ключевые слова:

int (целый);
char (символьный);
wchar_t (расширенный символьный);
bool (логический);
float (вещественный);
double (вещественный с двойной точностью).

Первые четыре типа называют целочисленными (целыми), последние два – типами с плавающей точкой. Код, который формирует компилятор для обработки целых величин, отличается от кода для величин с плавающей точкой. Существует четыре спецификатора типа, уточняющих внутреннее представление и диапазон значений стандартных типов:

short (короткий);
long (длинный);
signed (знаковый);

unsigned (беззнаковый).

Целый тип (int).

Размер типа int не определяется стандартом, а зависит от компьютера и компилятора. Для 16-разрядного процессора под величины этого типа отводится 2 байта, для 32-разрядного – 4 байта.
Для точного определения количества байт следует написать тестовую программу и включить в нее операцию:

s=sizeof(int);

Значение переменной s будет равно количеству байт занимаемому объектами указанного в скобках типа. Для определения диапазона значений целого типа данных следует воспользоваться формулой:
$Ds = -2^{8s-1} \ldots + 2^{8s-1} - 1$ для знаковых типов (int – знаковый тип) и
$Dus = 0 \ldots 2^{8s} - 1$ для беззнаковых (unsigned) типов.
Спецификатор short перед именем типа указывает компилятору, что под число требуется отвести 2 байта независимо от разрядности процессора. Спецификатор long означает, что целая величина будет занимать 4 байта. Таким образом, на 16-разрядном компьютере эквиваленты int и short int, а на 32-разрядном – int и long int.
Внутреннее представление величины целого типа – целое число в двоичном коде. При использовании спецификатора signed старший бит числа интерпретируется как знаковый (0 – положительное число, 1 – отрицательное). Спецификатор unsigned позволяет представлять только положительные числа, поскольку старший разряд рассматривается как часть кода числа. Таким образом, диапазон значений типа int зависит от спецификаторов.
По умолчанию все целочисленные типы считаются знаковыми, то есть спецификатор signed можно опускать.
Константам, встречающимся в программе, приписывается тот или иной тип в соответствии с их видом. Если этот тип по каким-либо причинам не уст-раивает программиста, он может явно указать требуемый тип с помощью суффиксов L, l (long) и U, u (unsigned). Например, константа 32L будет иметь тип long и

занимать 4 байта. Можно использовать суффиксы L и U одновременно, например, Ox22UL или 05Lu.
Типы short int, long int, signed int и unsigned int можно сокращать до short, long, signed и unsigned соответственно.

Символьный тип (char).

Под величину символьного типа отводится количество байт, достаточное для размещения десятичного кода любого символа из набора символов для данного компьютера, что и обусловило название типа. Как правило, это 1 байт. Тип char, как и другие целые типы, может быть со знаком или без знака. В величинах со знаком можно хранить значения в диапазоне от -128 до 127. При использовании спецификатора unsigned значения могут находиться в пределах от 0 до 255. Этого достаточно для хранения любого символа из 256-символьного набора ASCII. Величины типа char применяются также для хранения целых чисел, не превышающих границы указанных диапазонов.

Логический тип (bool).

Величины логического типа могут принимать только значения true и false, являющиеся зарезервированными словами. Внутренняя форма представления значения false – 0 (нуль). Любое другое значение интерпретируется как true. При преобразовании к целому типу true имеет значение 1.

Вещественный тип (float, double и long double).

Стандарт C++ определяет три типа данных для хранения вещественных значений: float, double и long double. Вещественные типы данных хранятся в памяти компьютера иначе, чем целочисленные. Внутреннее представление вещественного числа состоит из двух частей – мантиссы и порядка. В IBM PC-совместимых компьютерах величины типа float занимают 4 байта, из которых один двоичный разряд отводится под знак мантиссы, 8 разрядов под порядок и 23 под мантиссу.
Для величин типа double, занимающих 8 байт, под порядок и мантиссу отводится 11 и 52 разряда соответственно. Длина мантиссы определяет точ-ность числа, а длина порядка – его

диапазон.
Спецификатор long перед именем типа double указывает, что под величину отводится 10 байт.
Диапазон значений вещественных типов определяется с помощью тестовой программы, в которой следует каким-либо образом узнать значения следующих констант:

FLT_MIN…FLT_MAX – диапазон типа float,
DBL_MIN…DBL_MAX – диапазон типа double,
LDBL_MIN…LDBL_MAX – диапазон типа long double.

Эти константы находятся в библиотеке <float.h> и следует убедиться, что она подключена.
Константы с плавающей точкой имеют по умолчанию тип double. Можно явно указать тип константы с помощью суффиксов F, f (float) и L, 1 (long). Например, константа 2E+6L будет иметь тип long double, а константа 1.82f – тип float.
В стандарте ANSI диапазоны значений для основных типов не задаются, определяются только соотношения между их размерами, например:

sizeof(float) < sizeof(double) < sizeof(long double)
sizeof(char) < sizeof(short) < sizeof(int) <, sizeof(long)

Различные виды целых и вещественных типов, различающиеся диапазоном и точностью представления данных, введены для того, чтобы дать программисту возможность наиболее эффективно использовать возможности конкретной аппаратуры, поскольку от выбора типа зависит скорость вычислений и объем памяти. Но оптимизированная для компьютеров какого-либо одного типа программа может стать не переносимой на другие платформы, поэтому в общем случае следует избегать зависимостей от конкретных характеристик типов данных.

Тип void

Кроме перечисленных, к основным типам языка относится тип void, но множество значений этого типа пусто. Он используется для определения функций, которые не возвращают значения (такие функции в языке Паскаль называют процедурами, но в языке Си процедур нет), для указания пустого списка

аргументов функции, как базовый тип для указателей и в операции приведения.