Программирование на языке C++

Зариковская Наталья Вячеславовна

Лекция 3

Основные типы данных, идентификаторы и их внутреннее представление

- Как вам известно, язык C++ вырос из языка программирования C.
- Что из себя представляет язык C? Это предопределенный набор типов данных (тип char, int, float и т.д.), операции для работы с этими типами данных (например ==, + и т.д.) и небольшое множество операторов, которые образуют структуру управления (например, оператор цикла). Все это имеется и в C++ (так как C входит в него как подмножество), но кроме этого, в нем имеются механизмы, которые позволяют программисту определять новые типы данных, которые, возможно, лучше отражают реальные понятия, и определять операции над этими типами. В этих механизмах и заключается основная мощь этого языка. Но предопределенные типы данных, операции и структура управления составляют основу языка, без которой невозможно понять всего остального. Поэтому на протяжении нескольких лекций мы будем рассматривать именно такие предопределенные типы данных, предопределенные операции и операторы C++, то есть большей частью то, что входит в язык программирования C.

Основные символы языка и идентификаторы

- Первое, о чем нужно сказать - это об идентификаторах в C++. В первую очередь нужно подчеркнуть, что в C++ различаются прописные и строчные буквы (X и x будут разными именами). Это важное отличие C++ от многих других языков программирования, в первую очередь от Паскаля. Все ключевые слова C++ набираются строчными (маленькими) буквами и не могут быть использованы в качестве идентификаторов. Все остальные правила для идентификаторов должны быть вам уже знакомы - что идентификатор начинается с буквы, что знак подчеркивания считается буквой и т.д. - то же, что и в Паскале. Не рекомендуется начинать идентификаторы со знака подчеркивания - такие имена используются в стандартных библиотеках.
- Основными символами языка служат буквы латинского алфавита A-Z, a-z, арабские цифры 0-9, символы + = _ - ( ) * & % # ! |. , ; : ‘ / ? { } ~ \ [ ] ^, а также неотображаемые символы - пробел, перевод стоки, табуляция, забой.
- В языке C++ прописные и строчные символы обрабатываются как различные символы.
-

Основные символы языка и идентификаторы

- Как и любой язык программирования, составными элементами языка служат ЛЕКСЕМЫ. Лексема - неразделимая последовательность символов (в простейшем случае один символ), относящихся к базовому словарю, и распознаваемых компилятором. При написании программы лексемы могут быть разделены пробельными символами (пробел, символ табуляции, перевод строки, возврат каретки, перевод формата) или другими лексемами, такими как знаки операций. Последовательность пробельных символов рассматривается как один пробел.

Основные символы языка и идентификаторы

- Идентификатор - лексема, начинающаяся с буквы и состоящая из букв, цифр и знаков подчёр¬кивания. Языком C++ допускается произвольная длина идентификатора, однако, значимы только первые 31 символ. Идентификаторы, содержащие двойной знак подчёркивания, зарезервированы для системных целей.
- В языке C++, как и в других языках программирования, используются ключевые слова. Ключевые слова - это зарезервированные идентификаторы, которые наделены определенным смыслом и известны компилятору языка C++. К ключевым словам относятся:
- auto, double, int, struct, break, else, long, switch, register, tupedef, char, extern, return, void, case, float, unsigned, default, for, signed, union, do, if, sizeof, volatile, continue, enum, short, while, asm, fortran, near, far, cdecl, huge, paskal, interrupt.

- Ключевые слова не могут быть использованы в качестве идентификаторов при написании программы.
-

Комментарии и пробельные символы

- Следующие символы: пробел, табуляция, перевод строки, возврат каретки и комментарии рассматриваются компилятором только как разделители и в остальном на результат трансляции не влияют. Они называются пробельными символами. Где может быть хотя бы один такой символ - может быть их сколько угодно.
- В программе допускается использование комментариев. Комментарии это любая последовательность символов, заключенных между знаками начала -« /*» и конца - «*/» комментария или начинающихся знаком - «//» и заканчивающихся символом новой строки. Комментарии первого вида, ограниченные знаками - « /*» и «*/» не могут быть вложенными друг в друга. А комментарии, начинающиеся символом - «//» и заканчивающиеся символом новой строки могут быть вложенными в комментарий первого вида. Пример одного большого ком¬ментария может быть следующим:
- /* Вычисление суммы элементов вектора A[10]
- int S=0; //включает операцию обнуления
- For(int i=0; i<10; i++)
- S+=A[i]; //и операцию суммирования */
- Следует знать, что использование комментариев является признаком хорошего тона программирования, улучшая читабельность программы, и практически не влияет на длину рабочего кода программы.
-

Основные типы данных и их внутреннее представление

- Языком C++ поддерживаются следующие типы данных:
- предопределенные языком типы (основные);
- классы, определяемые пользователем;
- абстрактные типы данных, получаемые на основе использования классов и основных типов при решении пользователем конкретных задач.
- Любая константа, переменная, значение функции или выражения в C++ характеризуется своим типом. Тип этих объектов определяет множество допустимых значений, форму внутреннего представления, а так же множество допустимых операций. Поэтому все данные, используемые в программе, до их появления обязательно должны быть объявлены (описаны).
- Знание типов объектов также позволяет компилятору обнаруживать ошибки и избежать потерь времени при тестировании программы.

Определения и объявления

- В C++ переменная может быть объявлена и должна быть определена.
- Определение и объявление переменной различаются. Это очень важный момент.
- Определение переменной вызывает выделение памяти. Определение задает имя переменной и ее тип. Помимо этого может быть указано инициирующее значение для переменной. Должно быть одно и только одно определение переменной в программе. Переменная не может использоваться до ее определения.
- Объявление переменной объявляет, что переменная существует и определяется где - то в другом месте. Оно не является определением, не приводит к выделению памяти, а скорее уведомляет о том, что переменная определена где-то еще. В программе может быть несколько объявлений одной и той же переменной.
- В общем объявления и определения переменных могут называться описаниями.

Определения и объявления

- Оператор определения данных в общем виде может быть представлен:
- **[класс памяти] [тип] идентификатор;,**
- где **[класс памяти]** - определяет область видимости и время жизни (существования) идентификатора;
- **[тип]** - тип, заданный на момент определения, идентификатора.
- Язык C++ поддерживает четыре класса памяти: auto (автоматическая), extern (внешняя), register (регистровая), static(статическая).

Типы данных в C++. Константы и переменные

- Предопределённые (встроенные) типы - это типы, непосредственно поддерживаемые языком. К таким типам могут быть отнесены: простые (символ, целое число и вещественное число) и составные , для которых языком определены строгие правила их описания (указатели, вектор, многомерный вектор, перечисление, смесь, структура).

Целочисленные типы

- C++ поддерживает 5 целочисленных типов. Список и характеристики этих типов представлены в таблице

тип	размер бит	Десятичный min - max	Шестнадцатеричный min - max	Восьмеричный min - max
unsigned int	16	0-65535	0x8000 - 0xffff	00..0177777
short int	16	-32768 -32767	0x0000 -0x7fff	00..077777
int	16	-32768 -32767	0x0000- 0x7fff	00..077777
long	32	-2147483648 2147483648	Ox10000 -0x7fffffff	00..17777777777
unsigned long	32	0-4294967295	0x0000000-0xffffffff	00..37777777777

Целочисленные типы

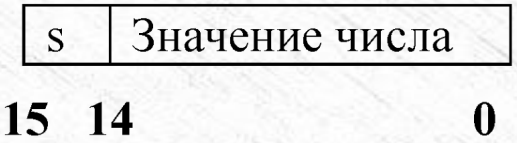
- Как видно из таблицы различные типы данных целого типа различают по количеству занимаемых в памяти бит и делят на беззнаковые (unsigned) и знаковые (signed). По умолчанию в объявлениях описатели short, int и long интерпретируются компилятором как signed. Беззнаковые данные в описаниях указываются явно.

- Например:
- int a,b; //описаны переменные a, b- signed int
- unsigned long l; //описана беззнаковая переменная l типа long

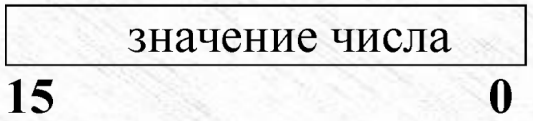
Целочисленные типы

- C++ поддерживает IEEE- стандарт внутреннего представления данных целого типа

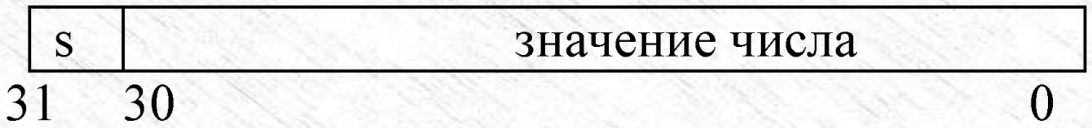
int



unsigned int



long int



long - unsigned

Целочисленные типы

- Старший бит знаковых чисел хранит знак числа. Если он равен 0, то число положительное и 1, если число отрицательное. Положительные числа хранятся и обрабатываются в прямом коде, а отрицательные в дополнительном коде.
- Следует заметить, что в языке C++ жёстко не фиксировано представление в памяти идентификаторов с модификаторами типа int и unsigned int. Размер памяти для переменной определяется длиной машинного слова (два или четыре байта) и зависит от используемой ПЭВМ.
- Язык C поддерживает явную форму инициализации переменной - через операцию присваивания.
- **Например**
- int n = 100 ;
- Язык C++ добавил к этому еще одну форму инициализации - неявную, при которой начальное значение располагается внутри скобок :
- int n(100);

Целочисленные типы

- При написании программы допускается использование символических констант. Константы в C++ могут быть заданы в десятичной, восьмеричной и шестнадцатеричной системах счисления. Тип и система счисления, в которой представлена символическая константа, определяется компилятором по ее записи (по умолчанию) или по использованным суффиксам (явно) Unsigned. Признаком используемой системы счисления для компилятора являются: для шестнадцатеричной системы счисления (C/C) - наличие в двух левых символах цифры 0 и буквы X(x); для восьмеричной C/C - наличие в качестве первого символа цифры 0; для десятичной C/C не удовлетворения двух вышеописанных условии.
- Признаком типа символической константы, задаваемой в явном виде, служит суффиксы U(u)- для констант типа unsigned; L(l)- для констант типа long. Разрешается комбинировать эти суффиксы в любом порядке.
- Отрицательные константы получаются применением операции «унарный минус» к соответствующей положительной константе.

Целочисленные типы

- **Примеры:**
- 8,27,-30201//десятичные имеют тип int, если не превышают максимально допустимого
- // для этого типа значения и тип long в противном случае
- 6U,30201U //unsigned int
- 6UL,30201UL //unsigned long
- 032,066,077 //восьмеричные
- 0x27,OX77A //шестнадцатеричные
- 0x27,0X77a //шестнадцатеричные длинные

Вещественные типы

- C++ поддерживает три вещественных типа. Список и характеристики этих типов представлены в таблице

ТИП	размер	диапазон представления чисел	диапазон бит отводимых под		знак	точность
	Бит (байт)	MIN-MAX	мантиссу	порядок	бит	10с/с
float	32 (4)	3.4E-38-3.4E 38	0/22	23/30	31	7
double	64 (8)	1.7E-308-1.7E 308	0/51	52/62	63	15
long double	80 (10)	3.4E-4932-3.4E 4932	0/63	64/78	79	19

Вещественные типы

- Вещественные числа (стандарт IEEE) состоят из знакового бита (s), «сдвинутого» порядка (p) и нормализованной мантиссы (m) (рисунок), представленной в двоичной системе счисления.

Число бит, выделяемых для хранения порядка и мантиссы, зависит от типа данных вещественного типа. С целью устранения необходимости хранения знака порядка и упрощения арифметических операций они хранится в «сдвинутом» виде. Это означает, что к истинному значению порядка в зависимости от типа вещественного числа прибавляется положительная константа-смещение (для float-127 для double-1023, для long double-16383).

Вещественные типы

Нормализованная мантисса - это способ хранения значения мантиссы с предполагаемой неявной единицей в старшем разряде. Это достигается, при нарушении нормализации, путем сдвига мантиссы влево до тех пор, пока старшей цифрой мантиссы не станет 1. Последнее позволяет не хранить, а неявно предполагать, наличие единицы в старшем разряде для типов float и double. Для типа long double отбрасывание старшей цифры мантиссы не производится.

Таким образом, формулы для расчета истинных значений вещественных (x) чисел имеют вид:
x=1.M*2^P (P-смещение), для типов float и double;
x=0.M*2^P (P-смещение), для long double.

Вещественные типы

При использовании констант в выражениях, компилятор отличает вещественные числа по наличию в записи десятичной точки, символа e или E. По аналогии с целочисленными константами информацию о типе символической константы компилятору поставляют: при наличии суффиксов f(F) константа интерпретируется как float или l(L) как long double; константы без использования суффиксов (по умолчанию) интерпретируются как double,

например:
9e-6 -double;
9e-6f -float;
9e-6l -long double.

При необходимости константа может быть представлена и инициализирована в разделе описания путем использования модификатора const перед типом, описываемой константы.

Например:
const long double pi=3.534653653;
const double e=2.7182;

Логические данные

Логические данные типа ‘’bool’’ введены в Borland C++ 5, занимают в памяти один байт и имеют два значения 1 (true) и 0 (false). Объявление логических переменных с инициализацией допускает использование двух предопределённых констант true (1) и false (0).

Пример:

```
bool dd=true; // dd=1
bool dd1=1;   // dd1=true.
```

Данные типа char

Данные типа char занимают в памяти 1 байт, определяются множеством значений кодовой таблицы ПЭВМ.
Код от 0 до 255 задает один из возможных символов кодовой таблицы (обычно ASCII- таблица).
Данные типа char могут рассматриваться компилятором как данные «целого» типа: данные со знаком (signed char); данные без знака (unsigned char).
Форма представления типа char как «целое» в IDE задается опцией компилятора Menu - Options Compile - Code Generation - Unsigned Chars (Signed chars).
Если тип char определен как Unsigned то диапазон изменения значений - от 0 до 255, и если signed то от -128 до 127

Данные типа char

двоичное представление	шестнадцатеричное представление	signed char	unsigned char
0000 0000	00	0	0
0000 0001		1	1
.....			
0111 1111	7f	127	127
1000 0000	80	-128	128
1000 0001	81	-127	129 .....
1111 1111	ff	-1	255

Как видно из таблицы для signed char старший бит определяет знак. Этот факт следует учитывать при операциях сравнения.

Например

объявление	операция сравнения	результат	интерпретация
char A=Ox80	(A < 0)	True	-128 < 0
unsigned char A=Ox80	(A < 0)	False	128 > 0

Данные типа char

В программе константа типа Char представляет собой символ, заключенный в одиночные кавычки - ‘A’, ‘F’, ‘$’ и т.д..
Например,
Char sim;
sim=’A’;.

Допускается использование символов, как целое, в выражениях.
Например:
int n=10*n+(s[i]-‘o’);

Следует знать, что использование символьных констант в сочетании с символом - обратная косая черта ‘\’, интерпретируется как управляющие
В языке C++ допускается представление символа его кодом в ASCII. Это достигается путем использования управляющей последовательности символов, состоящей из обратной косой черты, за которой следует внутренний код символа. Код символа может быть: в восьмеричной системе счисления, если первый символ ноль; в шестнадцатеричной системе счисления, если первый символ x (X); в десятичной системе счисления, если первый символ не 0 и не x.

Данные типа char

запись в программе	код символа		Выполняемая функция
	10 с/с	16 с/с	
'\o’	0	ox00	На экране не отображается. Рассматривается как ограничитель выводимой строки символов.
'\a’	7	ox07	Включает звуковой сигнал (BEL)
'\b’	8	ox08	Перемещение курсора на одну позицию влево (BS Забой)
'\t'	9	ox09	Горизонтальная табуляция (HT)
'\n’	10	ox0A	Перевод строки
'\v’	11	ox0B	Вертикальная табуляция (VT)
'\f’	12	ox0C	Прогон листа бумаги до начала следующей страницы
'\r’	13	ox0D	Возврат курсора на первый символ экрана (CR)
'\032’	26	ox1A	Курсор сдвигается на один символ влево
'\””	34	ox22	Символ двойной кавычки
'\”	39	ox27	Символ одиночной кавычки (апостроф)
‘\?’		0x3F	Вопросительный знак
‘\\’	92	ox5C	Символ обратной косой черты
‘\0ooo’	внутреннее представление символа, где ooo- значение кода символа в восьмеричной системе счисления		
‘\xhh’	внутреннее представление символа, где hh- значение кода символа в шестнадцатеричной системе счисления		

Данные типа char

Примеры:

```
char simvol='\0104'; //simvol= 'd'
char simvol='\x44';    //simvol='d'
char simvol='\68';      //simvol='d'
printf(''%c %c %c'','\0104','\x44','\68');// d d d
```