ЛЕКСЕМЫ ЯЗЫКА С++

Алфавита С++:

- A-Z, a-z, _;
- 0-9;
- специальные символлы: *, ? и т. д;
- пробельные символы: пробел, табуляция и переход на новую строку.

Лексема – это минимальная единица языка (без пробельных символов), имеющая самостоятельное значение.

Типы лексем:

- имена (идентификаторы);
- ключевые (зарезервированные) слова;
- числовые и символьные константы;
- знаки операций;
- разделители (скобки, точка, запятая, пробельные символы).

Простые переменные в C++ (идентификаторы)

Отслеживается:

- где хранится информация;
- какое значение сохранено;
- разновидность сохраненной информации.

Объект – место в памяти, имеющее тип, который определяет вид информации, разрешенной для хранения.

Переменная – именованный объект.

1. *Объявление* переменной:

```
int day;
```

int
day ?

2. *Инициализация* переменной:

```
day = 7;
```

int
day 7

```
int day = 7;
```

3. *Доступ* к переменной:

```
cout << day;
```

Инициализация переменных:

а) в стиле C:

```
int day = 7;
day = 7;
```

б) в стиле C++:

```
int day = (7);
int day = {7};
int day (7);
int day {7};
```

в) последовательность объявлений/инициализаций:

```
int a, b, c;
a =b = c =7;
```

г) НО!

```
int day = "7";
// ошибка компиляции
```

Основные *типы данных*:

```
int    number_of_steps = 39;  // int — для целых чисел
float  flying_time = 3.5;     // float — для чисел с плавающей точкой
char   decimal_point = '.'; // char — для символов
string name = "Annemarie";    // string — для строк
bool   tap_on = true;         // bool — для логических переменных
```

Имена, назначаемые переменным

ОСМЫСЛЕННЫЕ!!!

```
float cost_of_trip; // snakeCase
float costOfTrip;   // camelCase
float x;
float cot;
```

Идентификатор –последовательность символов, начинающаяся с буквы или знака подчеркивания, за которыми следуют (или нет) буквы, цифры или знаки подчеркивания.

Правила именования:

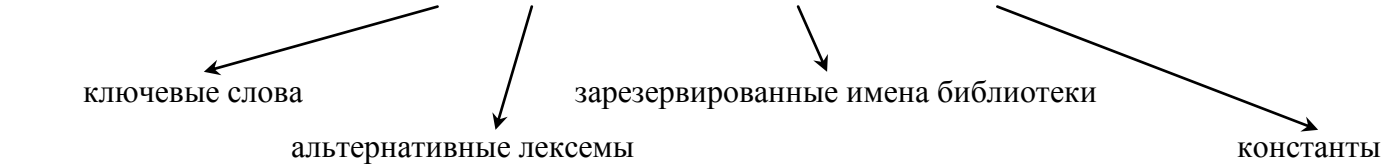
- `A-Z, a-z, 0-9, _`;
- первым символ – не `0-9`;
- `A-Z ≠ a-z`;
- `veryVeryLongName`.

Разрешённые имена	Запрещённые имена
`Day`	`4four`
`person7`	`step-to-step`
`very_log_variable_in_C`	`one#two#three`

НЕ используются в качестве идентификаторов зарезервированные слова C++:

- Ключевые слова C++;
- `_имя, __имя;`
- константы

Зарезервированные слова C++



Ключевые слова C++

Ключевые слова (keywords) – это идентификаторы, используемые самим языком для выражения языковых конструкций (формирующие словарь языка программирования).

```
alignas         constexpr       friend          reinterpret_c   true
alignof         continue        goto            ast             try
asm             decltype        if              requires        typedef
auto            default         inline          return          typeid
bool            delete          int             short_          typename
break           do              long            signed          union
case            double          mutable         sizeof          unsigned
catch           dynamic_cast    namespace       static          using
char            else            new             static_assert   virtual
char32_t        enum            noexcept        static_cast     void
charl6_t        explicit        nullptr         struct          volatile
class           export          operator        switch          wchar_t
compl           extern          private         template        while
concept         false           protected       this
const           float           public          thread_local
const_cast      for             register        throw
```

Альтернативные лексемы

```
and         &&                  not         !
and_eq      &=                  not_e       !=
bitand      &                   or          ||
bitor       |                   or_eq       |=
compl       ~                   xor         ^
                                xor_eq      ^=
```

Зарезервированные имена библиотеки C++

имена, зарезервированные для использования библиотекой C++.

```
<climits> - CHAR_BIT
<cmath> - sin
```

```
__gink
__Lynx
_lynx
```

```
int sin = 7;
sin(1.0);
```

Числовые и символьные константы

Символьные и строковые литералы

Строковые литералы – `"Hello"`.

Символьные литералы:

1. Единичные печатные символы:
   - `'A'` соответствует 65, коду ASCII для символа A;
   - `'a'` соответствует 97, коду ASCII для символа a;
   - `' '` соответствует 32, коду ASCII для символа пробела;

2. Управляющие последовательности:

Название символа	Символ ASCII	Код C++	Десятичный код ASCII	Шестнадцатеричный код ASCII
Новая строка	NL (LF)	\n	10	0xA
Вертикальная табуляция	VT	\v	11	0xB
Забой	BS	\b	8	0x8
Возврат каретки	CR	\r	13	0xD
Предупреждение	BEL	\a	7	0x7
Обратная косая черта	\	\\	92	0x5C
Знак вопроса	?	\?	63	0x3F
Одинарная кавычка	'	\'	39	0x27
Двойная кавычка	"	\"	34	0x22

Использование в строковых литералах:

– по коду C++

```
char alarm = '\a';
cout << alarm << "Don't do that again!\a\n";
cout << "Ben \"Buggsie\" Hacker\nwas here!\n";
```

– вместо манипуляторов

```
cout << endl; // использование манипулятора endl
cout << '\n'; // использование символьной константы
cout << "\n"; // использование строки
```

– для улучшения читаемости кода

```
cout << endl << endl << "What next?" << endl << "Enter a number:" << endl;
cout << "\n\nWhat next?\nEnter a number:\n";
```

– по восьмеричным (\0) или шестнадцатеричным (\x) кодам

```
cout << "Hello\x20world!";
cout << "Hello\040world!";
cout << "Hello world!";
```

Целочисленные литералы

+57 **57** **-57**

57 10 **057** 8 **0x57** или **0X57** 16 **0b1101** 2

1-9

Пример 1

```
cout << 57 << ' ' << 057 << ' ' << 0x57 << ' ' << 0b1101;
//      57           47           87             13
```

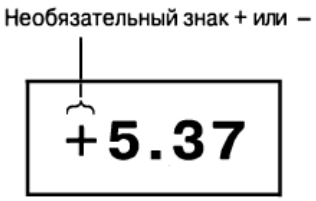
Пример 2

```
cout << 57 << oct << 47 << hex << 87;
//      57         57          57     - в десятичной системе

cout << showbase << 57 << oct << 057 << hex<< 0x57;
//                    57           057          0X57

noshowbase
```

Числа с плавающей точкой.

стандартный — экспоненциальный

Стандарный:



5,37

Вывод вещественных чиел в формате с фиксированной запятой:

```
cout << fixed << 13.7;  // 13.700000
cout << fixed << setprecision(4) << 13.7; // 13.7000
```

Экспоненциальный:

$5.37E+16 = 5.37 * 10^{16}$

d.ddd**E+**n – перемещение десятичной точки на n позиций вправо
d.ddd**E-**n – перемещение десятичной точки на n позиций влево

Пример.

```
2.52e+8      // можно использовать E или e; знак + необязателен
8.33E-4      // порядок может быть отрицательным
7E5          // то же, что и 7.0E+05
-18.32e13    // перед записью может стоять знак + или -
+5.98E24     // масса Земли в килограммах
9.11e-31     // масса электрона в килограммах
7.2 E6       // недопустимый формат
```

Вывод вещественных чиел в формате с плаваущей запятой:

```
cout << scientific << 13.7;  // 1.370000e+001
cout << scientific << uppercase << 13.7;
// 1.370000E+001
cout << scientific << setprecision(4) << 13.7;
// 1.3700e+001
```