



Семинар 1.

Лексические основы,
арифметические типы данных,
переменные и константы,
операторы,
линейный алгоритм



1. Лексические основы

1. Алфавит Си++
2. Идентификаторы и служебные слова
3. Константы-литералы
4. Перечисления
5. Комментарии



1.1. Алфавит

Алфавит в языке Си++ состоит из 96 символов. 91 – изображаемые:

- прописные и строчные буквы латинского алфавита

- десятичные цифры

- 29 спецсимволов:

" { } , | [] () + = / % \ ; ' : ? < = > _ ! & # ~ ^ . *



1.1. Алфавит

Неизображаемые символы:

- пробел
- горизонтальная табуляция
- вертикальная табуляция
- перевод строки
- начало новой строки

1.2. Идентификаторы и служебные слова



Идентификаторы необходимы для записи имён переменных и констант.

Идентификатор – последовательность произвольной длины из букв латинского алфавита, десятичных цифр и подчёркивания, начинающаяся не с цифры.

1.2. Идентификаторы и служебные слова



Примеры идентификаторов:
SUMMA, summa, variable1, var_1

Не являются идентификаторами:
9var, 90

1.2. Идентификаторы и служебные слова



Служебные (ключевые) слова – это идентификаторы, зарезервированные в языке. Служебные слова нельзя использовать в качестве произвольно выбираемых имён.

1.2. Идентификаторы и служебные слова



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

1.2. Идентификаторы и служебные слова



Служебные слова для альтернативного представления операций

and	&&	bitor		not_eq	!=	xor	^
and_eq	&=	compl	~	or		xor_eq	^=
bitand	&	not	!	or_eq	=		

1.2. Идентификаторы и служебные слова



Идентификаторы, начинающиеся с двух символов подчёркивания, резервируются для реализаций компиляторов Си++ и его стандартных библиотек. Идентификаторы, начинающиеся с одного символа подчёркивания используются в компиляторах Си++.



1.3. Константы-литералы

Константа = фиксированное значение

В Си++ существует несколько видов констант:

- константы-литералы
- именованные константы
- константы перечислений
- препроцессорные



1.3. Константы-литералы

Константы-литералы делятся на 5 групп:

- целые
- вещественные
- логические
- СИМВОЛЬНЫЕ
- строковые



1.3. Константы-литералы

Целые константы-литералы:

- десятичные – 0, 188, -10
- восьмеричные – 017 (15_{10}), 010 (8_{10})
- шестнадцатеричные – 0xED (237_{10})
- беззнаковые – 1u
- long – 1L



1.3. Константы-литералы

Вещественные константы (константы с плавающей точкой) могут включать 6 частей:

- целая часть
- точка
- дробная часть
- признак (символ) экспоненты e или E
- знак и показатель десятичной степени
- суффикс F (или f) или L (l)



1.3. Константы-литералы

Вещественные константы (константы с плавающей точкой):

12.5 12. .0 .13 1.23456F 1.23e-3

Вещественные константы без суффикса имеют тип `double`, с суффиксом `F` – `float`, с суффиксом `L` – `long double`



1.3. Константы-литералы

Логические константы:

- true (соответствует не 0)
- false (соответствует 0)



1.3. Константы-литералы

Символьные константы:

- обычные односимвольные (тип `char`). Пример: `'a'`, `'\n'`
- обычные мультисимвольные (тип `int`, зависит от реализации). Пример: `'abc'`
- широкие (тип `wchar_t`, зависит от реализации). Пример: `L'\n'`



1.3 Константы-литералы

Эскейп-последовательности – последовательности символов, начинающиеся со знака «\». Эскейп-последовательности бывают простыми, восьмеричными, шестнадцатеричными.

<code>\n</code>	<code>0x0A</code>	Новая строка
<code>\t</code>	<code>0x09</code>	Табуляция
<code>\\</code>	<code>0x27</code>	Обратная косая черта
<code>\ooo</code>	<code>ooo</code>	Символ, имеющий восьмеричный код <code>ooo</code>
<code>\xhhh</code>	<code>0xhhh</code>	Символ, имеющий шестнадцатеричный код <code>hhh</code>

```
cout << '\x21'; cout << '\041';
```



1.3. Константы-литералы

Строковые константы:

- существуют широкие строковые константы, начинающиеся с символа L:
L"hello"
- чаще применяются ординарные (или узкие) строковые константы:
"hello"
- тип строковой константы – char[]



1.4. Перечисления

Перечисления создаются с помощью служебного слова `enum`. Элементы перечисления – целочисленные константы (тип `int`). Пример:

```
enum {one = 1, two = 2, three = 3};  
enum {zero, one, two, three, four};  
enum {ten=10, three=3, four, five};  
enum {zero, nought=0, one, two, pair = 2,  
three};
```

1.4. Перечисления



Классы-перечисления:

```
enum class weekdays {  
    Mon,  
    Tue,  
    Wed,  
    Thu,  
    Fri,  
    Sat,  
    Sun  
};
```

```
int main()  
{  
    weekdays wd = weekdays::Mon;  
    ...
```



1.5. Комментарии

Комментарии служат средством для записи пояснений к различным участкам кода программы. Типы комментариев:

- `//` Однострочный комментарий
- `/*` Многострочный комментарий `*/`



Пример программы

Цель: создать программу для вывода
литеральных констант и занимаемого
ими места в памяти.