

Основни елементи на C++

Трифон Трифонов

Увод в програмирането,
спец. Компютърни науки, 1 поток, 2018/19 г.

11–18 октомври 2018 г.

Азбука

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	0	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	

Изображения: wikipedia.org

Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.
- книга. чете Иван? интерес на
- $\langle \text{изречение} \rangle ::= \langle \text{подлог} \rangle \langle \text{сказуемо} \rangle [\langle \text{определение} \rangle] \langle \text{допълнение} \rangle .$
- $\langle \text{подлог} \rangle ::= \langle \text{собствено_съществително} \rangle | \langle \text{нарицателно_съществително} \rangle \langle \text{пълен_член} \rangle$
- $\langle \text{пълен_член} \rangle ::= \text{ът} | \text{ят} | \text{та} | \text{то}$
- $\langle \text{сказуемо} \rangle ::= \langle \text{глагол} \rangle$
- $\langle \text{определение} \rangle ::= \langle \text{прилагателно} \rangle$
- $\langle \text{допълнение} \rangle ::= \langle \text{собствено_съществително} \rangle | \langle \text{нарицателно_съществително} \rangle$

Синтактичен анализ — пример 1

- <изречение>
- <подлог> <сказуемо> [<определение>] <допълнение>.
- <собствено_съществително> <сказуемо> <определение> <допълнение>.
- **Иван** <глагол> <определение> <допълнение>.
- **Иван чете** <определение> <нарицателно_съществително>.
- **Иван чете** <прилагателно> **книга**.
- **Иван чете интересна книга**.

Синтактичен анализ — пример 2

- <изречение>
- <подлог> <сказуемо> [<определение>] <допълнение>.
- <нарицателно_съществително><пълен_член> <сказуемо>
<допълнение>.
- **Студент**<пълен_член> <глагол> <допълнение>.
- **Студентът** <глагол> <нарицателно_съществително>.
- **Студентът** <глагол> **програма**.
- **Студентът** **пише** **програма**.

Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [<определение>] <допълнение>.
- <нарицателно_съществително><пълен_член> <сказуемо>
<собствено_съществително>.
- Програма<пълен_член> <глагол> Иван.
- Програмата гледа Иван.



- Освен да е построено правилно, изречението трябва да има смисъл!
- **Семантика:** смисъл, значение на текст

Мета-език на Backus-Naur

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
- $\langle \text{цяло_число_без_знак} \rangle ::= \langle \text{цифра} \rangle \{ \langle \text{цифра} \rangle \}$
- $\langle \text{цяло_число} \rangle ::= [+ | -] \langle \text{цяло_число_без_знак} \rangle$
 - $-15, 2, +412$
- $\langle \text{латинска_буква} \rangle ::= A \mid B \mid \dots \mid Y \mid Z \mid a \mid b \mid \dots \mid y \mid z$
- $\langle \text{идентификатор} \rangle ::= _ \mid \langle \text{латинска_буква} \rangle$
 $\{ \langle \text{латинска_буква} \rangle \mid \langle \text{цифра} \rangle \mid _ \}$
 - $a, name, X1, _Data15$

Основни думи на C++ (tokens)

- $\langle \text{идентификатор} \rangle ::= _ \mid \langle \text{латинска_буква} \rangle$
 $\{ \langle \text{латинска_буква} \rangle \mid \langle \text{цифра} \rangle \mid _ \}$
- запазени думи
- стандартни идентификатори
- литерали
 - числови (1, -5, +2.34, 1e-02, 012, 0x123)
 - символни ('a', '\t')
 - низови ("hello", "yes!")
- операции (+, -, *, /)
- разделители (: ; , () [] { } < >)

Коментари

- `<коментар> ::= //<текст_на_един_ред> | /* <текст> */`
- Компиляторът игнорира:
 - коментари
 - празни символи (интервал, табулация, нов ред)
- Пример:

```
int sum = 0; // нулираме сумата
/*
```

*вече сме готови да започнем пресмятането
последователно ще натрупваме поредните числа в sum
докато не ги изчерпим всичките*

```
*/
```

```
...
```

Променливи

Променливата е именувана област в паметта.

Различно от променлива в математиката!

- Име (идентификатор)
- Място в паметта (адрес)
- Тип
- Стойност

	fn		c		pi
...	81111	...	F	...	3.14159
	int		char		double

Дефиниция и присвояване

$$\langle \text{дефиниция} \rangle ::= \langle \text{тип} \rangle \langle \text{идентификатор} \rangle [= \langle \text{израз} \rangle] \{ , \\ \langle \text{идентификатор} \rangle [= \langle \text{израз} \rangle] \};$$

$$\langle \text{присвояване} \rangle ::= \langle \text{идентификатор} \rangle = \langle \text{израз} \rangle ;$$

Примери:

- `double x;`
- `int a, b = 15;`
- `a = b + 5;`
- `x = a * (b - 3);`
- ~~`double y = double x;`~~

Изход на екрана

- `cout << <израз> {<< <израз>};`
- `((cout << a) << b) << c;`

- `cout << "a + b = " << a + b << endl;`
- ~~`cout << "a = " << 2;`~~

Вход от клавиатурата

- `cin >> <идентификатор> {>> <идентификатор>};`
- `((cin >> a) >> b) >> c;`

- ~~`cin >> a + b;`~~
- ~~`cin >> 15;`~~

Константи

- `const` <тип> <идентификатор> = <израз>;
- стойността на константите:
 - трябва да бъде зададена при дефиниране
 - не може да се променя след това
- Примери:
 - `const int FINGERS = 10;`
 - ~~`FINGERS = FINGERS + 2;`~~

Типове

- Класификация на видовете данни
- Носят **семантична** информация
- Помагат за проверка на коректност
- Множество от допустими стойности
- Операции
- Вградени функции

Класификация на типовете

- Скаларни (атомарни)
 - интегрални
 - булев (bool)
 - целочислен (int)
 - символен (char)
 - изброен (enum)
 - други
 - числа с плаваща запетая (float, double)
 - указател (T*)
 - псевдоним (T&)
- Съставни
 - масив ([])
 - низ (char[])
 - структура (struct)
 - клас (class)
 - обединение (union)

Логически тип (bool)

- Множество от стойности: {false, true}
- <булева_константа> ::= true | false
- логически операции

Конюнкция

&&	false	true
false	false	false
true	false	true

Дизюнкция

	false	true
false	false	true
true	true	true

Отрицание

!	
false	true
true	false

Символен тип (char)

- Множество от стойности
 - signed char: [-128; 127]
 - unsigned char: [0; 255]
- Литерали
 - '<символ>'
 - '\\<контролен_символ>'

Целочислен тип (int)

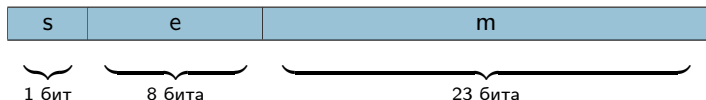
- Множество от стойности: $[-2^{31}; 2^{31} - 1]$
- модификатори
 - short: $[-2^{15}; 2^{15} - 1]$
 - long: $[-2^{63}; 2^{63} - 1]$
 - unsigned: $[0; 2^x - 1]$, където ($x = 16, 32, 64$)

Целочислен тип (`int`)

- аритметични операции
 - едноместни операции за знак (+, -)
 - двуместни аритметични операции
 - $a + b$ (събиране)
 - $a - b$ (изваждане)
 - $a * b$ (умножение)
 - a / b (частно)
 - $a \% b$ (остатък)
- операции за сравнение (предикати)
 - $a == b$ (равно)
 - $a != b$ (различно)
 - $a < b$ (по-малко)
 - $a > b$ (по-голямо)
 - $a <= b$ (по-малко или равно)
 - $a >= b$ (по-голямо или равно)

Числа с плаваща запетая

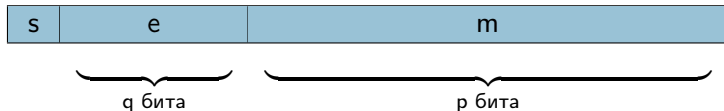
- **Внимание: това не са реални числа!**
 - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**
- Представяне в паметта
 - $f = (-1)^s \cdot m \cdot 2^e$



- $s \in \{0, 1\}$ — знак
- $m \in [0; 2^{23} - 1]$ — мантиса
- $e \in [-126; 127]$ — експонента
- машинна нула: $(-2^{-127}; 2^{-127})$

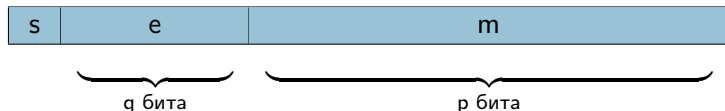
Общо представяне на числа с плаваща запетая

$$f = (-1)^s \cdot m \cdot 2^e$$



- $s \in \{0, 1\}$ — знак
- $m \in [0; 2^p - 1]$ — мантиса
 - p — точност, брой двоични цифри на мантисата
- $e \in [e_{min}; e_{max}]$ — експонента
 - $e_{min} = -e_{max} + 1$
 - $e_{max} = 2^{q-1} - 1$
 - q — обхват на експонентата
 - $e_{min} - 1$ и $e_{max} + 1$ са запазени за служебно ползване
- общо използвани битове: $p + q + 1$
- машинна нула: $(-2^{e_{min}-1}; 2^{e_{min}-1})$

Типове дробни числа



тип	размер	точност (p)	обхват (q)
float	4 байта = 32 бита	23 бита	8 бита
double	8 байта = 64 бита	52 бита	11 бита
long double	16 байта = 128 бита	112 бита	15 бита

Дробни литерали

- [`<цяло_число>`]. [`<цяло_без_знак>`][`(E|e)<цяло_число>`]
- Примери: 1, 2.34, 12e-2, 10.14E+03, .23
- Операции:
 - всички за целочислен тип **без %**
 - / е **дробно деление**, а не частно!
 - сравненията == и != са **ненадеждни!**

Математически функции

```
#include <cmath>
```

- `abs(x)`, `fabs(x)`
- `sin(x)`, `cos(x)`, `tan(x)`, `asin(x)`, `acos(x)`, `atan(x)`
- `exp(x)`, `log(x)`, `log10(x)`
- `ceil(x)`, `floor(x)`
- `sqrt(x)`, `pow(x, n)`

Преобразуване на типове

- `bool` → `char` → `short` → `int` → `long` → `float` → `double`
- `unsigned char` → `unsigned short` → `unsigned` → `unsigned long`
- обратната посока може да доведе до **загуба на информация**
- експлицитно преобразуване на типове:
`<преобразуване> ::= (<тип>) <израз>`

Приоритет на операциите

- 1 Обръщания към функции
- 2 Скоби
- 3 `!`, `+`, `-` (едноместни)
- 4 `*`, `/`, `%`
- 5 `+`, `-` (двуместни)
- 6 `<<`, `>>`
- 7 `<`, `<=`, `>`, `>=`
- 8 `==`, `!=`
- 9 `&&`
- 10 `||`