

# 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		!	"	#	<del>\$</del>	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	<del>@</del>	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	<del>0</del>	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

# Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.

# Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.

# Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.
- книга. чете Иван? интерес на

# Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.
- книга. чете Иван? интерес на
- $\langle \text{изречение} \rangle ::= \langle \text{подлог} \rangle \langle \text{сказуемо} \rangle [ \langle \text{определение} \rangle ] \langle \text{допълнение} \rangle .$

## Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.
- книга. чете Иван? интерес на
- $\langle \text{изречение} \rangle ::= \langle \text{подлог} \rangle \langle \text{сказуемо} \rangle [ \langle \text{определение} \rangle ] \langle \text{допълнение} \rangle .$
- $\langle \text{подлог} \rangle ::= \langle \text{собствено\_съществително} \rangle | \langle \text{нарицателно\_съществително} \rangle \langle \text{пълен\_член} \rangle$



## Синтаксис

- Правила за построяване на текст
- Иван чете интересна книга.
- Студентът пише програма.
- книга. чете Иван? интерес на
- $\langle \text{изречение} \rangle ::= \langle \text{подлог} \rangle \langle \text{сказуемо} \rangle [ \langle \text{определение} \rangle ] \langle \text{допълнение} \rangle .$
- $\langle \text{подлог} \rangle ::= \langle \text{собствено\_съществително} \rangle | \langle \text{нарицателно\_съществително} \rangle \langle \text{пълен\_член} \rangle$
- $\langle \text{пълен\_член} \rangle ::= \text{ът} | \text{ят} | \text{та} | \text{то}$

## Синтаксис

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

## Синтаксис

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

## 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>

# Синтактичен анализ — пример 1

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.

# Синтактичен анализ — пример 1

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <собствено\_съществително> <сказуемо> <определение>  
<допълнение>.

## Синтактичен анализ — пример 1

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <собствено\_съществително> <сказуемо> <определение>  
<допълнение>.
- **Иван** <глагол> <определение> <допълнение>.



# Синтактичен анализ — пример 1

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <собствено\_съществително> <сказуемо> <определение>  
<допълнение>.
- **Иван** <глагол> <определение> <допълнение>.
- **Иван** **чете** <определение> <нарицателно\_съществително>.

# 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**.

# 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>

## Синтактичен анализ — пример 2

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.

## Синтактичен анализ — пример 2

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<допълнение>.

## Синтактичен анализ — пример 2

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<допълнение>.
- **Студент**<пълен\_член> <глагол> <допълнение>.

## Синтактичен анализ — пример 2

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<допълнение>.
- **Студент**<пълен\_член> <глагол> <допълнение>.
- **Студентът** <глагол> <нарицателно\_съществително>.



## 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**.

## 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>

## Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.

## Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<собствено\_съществително>.

## Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<собствено\_съществително>.
- Програма<пълен\_член> <глагол> Иван.

## Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<собствено\_съществително>.
- Програма<пълен\_член> <глагол> Иван.
- Програмата гледа Иван.

## Синтактичен анализ — пример 3

- <изречение>
- <подлог> <сказуемо> [ <определение> ] <допълнение>.
- <нарицателно\_съществително><пълен\_член> <сказуемо>  
<собствено\_съществително>.
- Програма<пълен\_член> <глагол> Иван.
- Програмата гледа Иван.





## 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!

## 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

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

## Мета-език на Backus-Naur

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
- $\langle \text{цяло\_число\_без\_знак} \rangle ::= \langle \text{цифра} \rangle \{ \langle \text{цифра} \rangle \}$

## Мета-език на Backus-Naur

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
- $\langle \text{цяло\_число\_без\_знак} \rangle ::= \langle \text{цифра} \rangle \{ \langle \text{цифра} \rangle \}$
- $\langle \text{цяло\_число} \rangle ::= [ + | - ] \langle \text{цяло\_число\_без\_знак} \rangle$

## 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

## Мета-език на Backus-Naur

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
- $\langle \text{цяло\_число\_без\_знак} \rangle ::= \langle \text{цифра} \rangle \{ \langle \text{цифра} \rangle \}$
- $\langle \text{цяло\_число} \rangle ::= [ + | - ] \langle \text{цяло\_число\_без\_знак} \rangle$ 
  - $-15, 2, +412$
- $\langle \text{латинска\_буква} \rangle ::= A \mid B \mid \dots \mid Y \mid Z \mid a \mid b \mid \dots \mid y \mid z$

# Мета-език на Backus-Naur

- $\langle \text{цифра} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
- $\langle \text{цяло\_число\_без\_знак} \rangle ::= \langle \text{цифра} \rangle \{ \langle \text{цифра} \rangle \}$
- $\langle \text{цяло\_число} \rangle ::= [ + | - ] \langle \text{цяло\_число\_без\_знак} \rangle$ 
  - $-15, 2, +412$
- $\langle \text{латинска\_буква} \rangle ::= A \mid B \mid \dots \mid Y \mid Z \mid a \mid b \mid \dots \mid y \mid z$
- $\langle \text{идентификатор} \rangle ::= \_ \mid \langle \text{латинска\_буква} \rangle$   
 $\{ \langle \text{латинска\_буква} \rangle \mid \langle \text{цифра} \rangle \mid \_ \}$



# 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)

- $\langle \text{идентификатор} \rangle ::= \_ \mid \langle \text{латинска\_буква} \rangle$   
 $\{ \langle \text{латинска\_буква} \rangle \mid \langle \text{цифра} \rangle \mid \_ \}$

# 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

# 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

# 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

# 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)

# 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')

# 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!")



# 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 (+, -, \*, /)

# 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> */`

# Коментари

- `<коментар> ::= //<текст_на_един_ред> | /* <текст> */`
- Компиляторът игнорира:

# Коментари

- $\langle \text{коментар} \rangle ::= // \langle \text{текст\_на\_един\_ред} \rangle \mid /* \langle \text{текст} \rangle */$
- Компиляторът игнорира:
  - коментари

# Коментари

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

# 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!



# Променливи

Променливата е именувана област в паметта.

Различно от променлива в математиката!

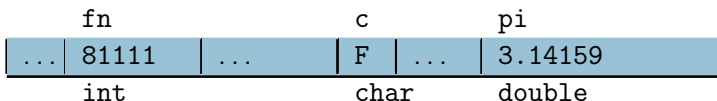
- Име (идентификатор)
- Място в паметта (адрес)
- Тип
- Стойност

# Променливи

Променливата е именувана област в паметта.

Различно от променлива в математиката!

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



## Дефиниция и присвояване

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

# 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;`

$$(((2+5)+8)+11)+15$$

# Изход на екрана

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

## Изход на екрана

- `cout << <израз> {<< <израз>};`

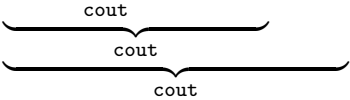
- `((cout << a) << b) << c;`

Diagram illustrating the output of the code `((cout << a) << b) << c;` using curly braces to group the output:

```

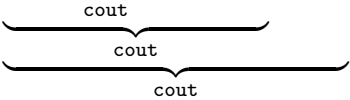

      cout
    ┌───────────┐
  ┌───────────────────┐
┌───────────────────────────┐
cout
  
```

## Изход на екрана

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

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



## 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;`

# Вход от клавиатурата

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

# Вход от клавиатурата

- `cin >> <идентификатор> {>> <идентификатор>};`

- `((cin >> a) >> b) >> c;`

Diagram illustrating the nested structure of the input statement `((cin >> a) >> b) >> c;` using curly braces:

- The innermost expression `cin >> a` is grouped by a brace labeled `cin`.
- The expression `(cin >> a) >> b` is grouped by a brace labeled `cin`.
- The entire expression `((cin >> a) >> b) >> c` is grouped by a brace labeled `cin`.

# Вход от клавиатурата

- `cin >> <идентификатор> {>> <идентификатор>};`

- `((cin >> a) >> b) >> c;`

Diagram illustrating the nested structure of the code `((cin >> a) >> b) >> c;` using curly braces to show the scope of each `cin` object:

- The innermost `cin` is associated with `a`.
- The middle `cin` is associated with `a` and `b`.
- The outermost `cin` is associated with `a`, `b`, and `c`.

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

## Вход от клавиатурата

- `cin >> <идентификатор> {>> <идентификатор>};`

- `((cin >> a) >> b) >> c;`

Diagram illustrating the nested structure of the input statement `((cin >> a) >> b) >> c;` using curly braces:

- The innermost expression `cin >> a` is grouped by a brace labeled `cin`.
- The expression `(cin >> a) >> b` is grouped by a brace labeled `cin`.
- The entire expression `((cin >> a) >> b) >> c` is grouped by a brace labeled `cin`.

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

- ~~`cin >> 15;`~~

# Константи

- `const` <тип> <идентификатор> = <израз>;

# Константи

- `const` <тип> <идентификатор> = <израз>;
- стойността на константите:



# Константи

- `const` <тип> <идентификатор> = <израз>;
- стойността на константите:
  - **трябва** да бъде зададена при дефиниране

# Константи

- `const` <тип> <идентификатор> = <израз>;
- стойността на константите:
  - трябва да бъде зададена при дефиниране
  - не може да се променя след това

# Константи

- `const` <тип> <идентификатор> = <израз>;
- стойността на константите:
  - **трябва** да бъде зададена при дефиниране
  - **не може** да се променя след това
- Примери:

# 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;`

# 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

# Типове

- Класификация на видовете данни
- Носят **семантична** информация

# Типове

- Класификация на видовете данни
- Носят **семантична** информация
- Помагат за проверка на коректност



# Типове

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

# Типове

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

# 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

$$\begin{aligned} &!(a \&\&b) \Leftrightarrow !a \vee !b \\ &!(a \vee b) \Leftrightarrow !a \&\&!b \end{aligned}$$

## Символен тип (char)

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

10	\n	LF
13	\r	CR
8	\t	TAB

# Целочислен тип (int)

- Множество от стойности:  $[-2^{31}; 2^{31} - 1]$   $2^{32} \rightarrow 4$  байта
- модификатори
  - short:  $[-2^{15}; 2^{15} - 1]$   $2^{16} \rightarrow 2$  байта
  - long:  $[-2^{63}; 2^{63} - 1]$   $2^{64} \rightarrow 8$  байта
  - 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!

$$\pi = 3.14159\dots$$

$$\mathbb{N} = \{0, 1, 2, \dots\}$$

$$\mathbb{Z} = \{\dots, -1, 0, 1, \dots\} \quad (-)$$

$$\mathbb{Q} = \left\{ \frac{p}{q} \mid p \in \mathbb{Z}, q \in \mathbb{N} \setminus \{0\} \right\} \quad (1)$$

$$\mathbb{R} = \sum_{n=N}^{\infty} a_n \cdot 10^n = a_N \cdot 10^N + a_{N+1} \cdot 10^{N+1} + \dots \quad (\text{lim})$$

$$+ a_1 \cdot 10 + 0 + a_{-1} \cdot 10^{-1} +$$

$$+ a_{-2} \cdot 10^{-2} + \dots$$

## Числа с плаваща запетая

- Внимание: това не са реални числа!
  - А какво са реални числа?

$$r = \sum_{n=-\infty}^N a_n \cdot 10^n$$

$a_N a_{N-1} \dots a_2 a_1 a_0, a_{-1} a_{-2} \dots$

$$101,01_{(2)} = 2^2 + 2^0 + 2^{-2} = 5,25$$

# Числа с плаваща запетая

- **Внимание: това не са реални числа!**
  - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**

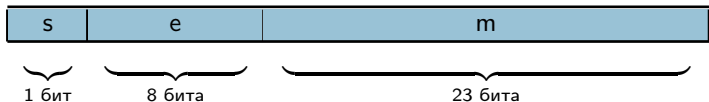
# Числа с плаваща запетая

- **Внимание: това не са реални числа!**
  - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**
- Представяне в паметта

# Числа с плаваща запетая

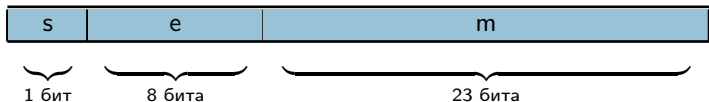
- **Внимание:** това не са реални числа!
  - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**
- Представяне в паметта
  - $f = (-1)^s \cdot m \cdot 2^e$

$$\begin{aligned}
 & 3 \overbrace{14159}^{23} 26535 \\
 & 3.14159 \\
 & = (-1)^0 \cdot 314159 \cdot 10^{-5}
 \end{aligned}$$



# Числа с плаваща запетая

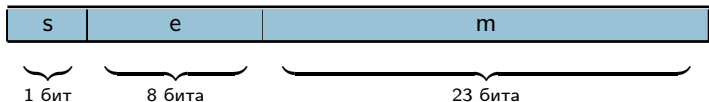
- **Внимание: това не са реални числа!**
  - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**
- Представяне в паметта
  - $f = (-1)^s \cdot m \cdot 2^e$



- $s \in \{0, 1\}$  — знак

# Числа с плаваща запетая

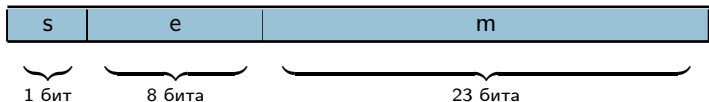
- **Внимание: това не са реални числа!**
  - А какво са реални числа?
- Още ще ги наричаме и **дробни числа**
- Представяне в паметта
  - $f = (-1)^s \cdot m \cdot 2^e$



- $s \in \{0, 1\}$  — знак
- $m \in [0; 2^{23} - 1]$  — мантиса

# 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

$$\left( 2^{23} \cdot (-1) \cdot 2^{127} \right)$$

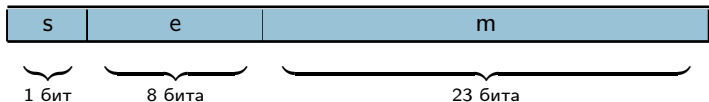
$$1 \cdot 2^{-126}$$



# 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

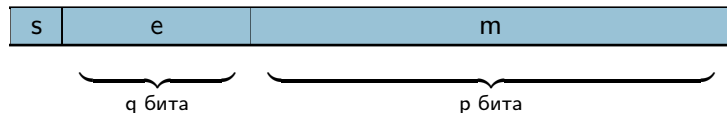
- $f = (-1)^s \cdot m \cdot 2^e$



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

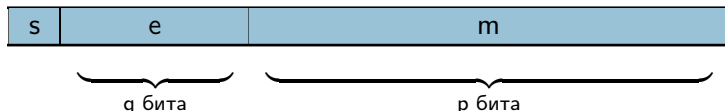
## Общо представяне на числа с плаваща запетая

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



- $s \in \{0, 1\}$  — 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!**

*12.10<sup>-2</sup> → 0.12*

# Математически функции

```
#include <cmath>
```

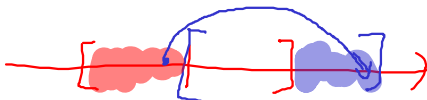
- `abs(x)`, `fabs(x)`
- `sin(x)`, `cos(x)`, `tan(x)`, `asin(x)`, `acos(x)`, `atan(x)`
- `exp(x)`, `log(x)`, `log10(x)`
- `ceil(x)`, `floor(x)`
- `sqrt(x)`, `pow(x, n)`

Handwritten notes illustrating mathematical functions and their relationships:

- Red annotations above `log(x)` and `log10(x)` include  $e^x$  and  $\ln x$ .
- Red annotations next to `ceil(x)` and `floor(x)` show the ceiling and floor functions:  $\lceil \cdot \rceil$  and  $\lfloor \cdot \rfloor$ .
- Red annotations next to `pow(x, n)` show the relationship between logarithms:  $\log_2 5 = \frac{\ln 5}{\ln 2}$ .
- Red annotations below `sqrt(x)` show the relationship between roots and powers:  $\sqrt[n]{8} = \text{pow}(8, 0.2)$ .

# 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 `||`