Tokens

Programming in C

Shirley B. Chu
shirley.chu@delasalle.ph

College of Computer Studies
De La Salle University

October 26, 2021

Programming Language

A programming language is a formal language that is used to give instructions to a computer in the form of a program or code.

Like any natural language, C has its own set of vocabulary and grammar. The words in C are called tokens and the grammar is called syntax.

Terms

White spaces refer to blanks (space), tabs, newlines, and comments.

Terms

White spaces refer to blanks (space), tabs, newlines, and comments.

A **token** is a series of contiguous characters that the compiler treats as a unit.

Terms

White spaces refer to blanks (space), tabs, newlines, and comments.

A **token** is a series of contiguous characters that the compiler treats as a unit.

- literals

Terms

White spaces refer to blanks (space), tabs, newlines, and comments.

A **token** is a series of contiguous characters that the compiler treats as a unit.

- literals
- identifiers

Terms

White spaces refer to blanks (space), tabs, newlines, and comments.

A **token** is a series of contiguous characters that the compiler treats as a unit.

- literals
- identifiers
- keywords

Literals

- used to represent fixed values

Literals

- used to represent fixed values
- Classifications:
  - Numeric
    - whole numbers (integers)
    - real numbers (floating-point numbers)

Literals

- used to represent fixed values
- Classifications:
  - Numeric
    - whole numbers (integers)
    - real numbers (floating-point numbers)
  - Non-numeric
    - characters
    - strings

Integers

✓ consists of a sequence of digits

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign
  - ✗ comma

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign
  - ✗ comma
  - ✗ decimal point

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign
  - ✗ comma
  - ✗ decimal point

Note that, in C:

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have + or − preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign
  - ✗ comma
  - ✗ decimal point

Note that, in C:

- a series of digits that starts with a zero 0 is considered *octal*.

Integers

- ✓ consists of a sequence of digits
- ✓ may be signed, i.e. may have `+` or `-` preceding the number
- ✓ none of the following
  - ✗ space/s in between digits or sign
  - ✗ comma
  - ✗ decimal point

Note that, in C:

- a series of digits that starts with a zero `0` is considered *octal*.
- a series of digits preceded by `0x` or `0X` is considered *hexadecimal*.

Floating-point Numbers

- consists of an integer part, a decimal point, and a fraction part.

  `3.5`, `-27.5`, `4.0`

Floating-point Numbers

- consists of an integer part, a decimal point, and a fraction part.
  `3.5`, `-27.5`, `4.0`

- may be expressed using the scientific notation.
  `2.25e4` to mean 2.25×10^4, equivalent to 22500.0
  `2E-4` to mean 2×10^{-4}, equivalent to 0.0002

Character Literals

- consist of a single character enclosed in single quotation marks.

```
'x', 'a', '9'
```

[1] http://www.asciitable.com/

Character Literals

- consist of a single character enclosed in single quotation marks.

  `'x'`, `'a'`, `'9'`

- has an equivalent numeric value (a.k.a. *ASCII value*[1])

  ASCII value of 'A' is 65, 'B' is 66, . . . and 'a' is 97, 'b' is 98, ...

[1] http://www.asciitable.com/

Character Literals

- consist of a single character enclosed in single quotation marks.

  `'x'`, `'a'`, `'9'`

- has an equivalent numeric value (a.k.a. *ASCII value*[1])

  ASCII value of 'A' is 65, 'B' is 66, . . . and 'a' is 97, 'b' is 98, ...

- may be a special character (a.k.a. *escape sequence*)

[1] http://www.asciitable.com/

Character Literals

- consist of a single character enclosed in single quotation marks.

  `'x'`, `'a'`, `'9'`

- has an equivalent numeric value (a.k.a. *ASCII value*[1])

  ASCII value of 'A' is 65, 'B' is 66, . . . and 'a' is 97, 'b' is 98, ...

- may be a special character (a.k.a. *escape sequence*)
  - enclosed in single quotation marks

[1] http://www.asciitable.com/

Character Literals

- consist of a single character enclosed in single quotation marks.

  `'x'`, `'a'`, `'9'`

- has an equivalent numeric value (a.k.a. *ASCII value*[1])

  ASCII value of 'A' is 65, 'B' is 66, . . . and 'a' is 97, 'b' is 98, ...

- may be a special character (a.k.a. *escape sequence*)
  - enclosed in single quotation marks
  - preceded by a blackslash \

Special Character	Meaning
`'\n'`	newline
`'\t'`	tab
`'\b'`	backspace
`'\''`	single quote

[1] http://www.asciitable.com/

String Literals

- consists of zero or more characters enclosed in double quotation marks.

```
"hello there!"
```

String Literals

- consists of zero or more characters enclosed in double quotation marks.

```
"hello there!"
```

- Escape sequences within a string must be preceded by a backslash.

```
"Good morning!\n"
```

String Literals

- consists of zero or more characters enclosed in double quotation marks.

```
"hello there!"
```

- Escape sequences within a string must be preceded by a backslash.

```
"Good morning!\n"
```

- double quotation mark within a string literal must be preceded by a backslash

```
"She said, \" hello there!\""
```

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

```
nice
```

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

  `nice` ✓

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

```
nice          ✓
idno.
```

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

```
nice          ✓
idno.         ✗
```

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

`nice`	✓	`first_digit`
`idno.`	✗	

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

`nice`	✓	`first_digit`	✓
`idno.`	✗		

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters A–Z, a–z, digits 0–9, or underscores _

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

  `friend4`

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓
`3rdno`	

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓
`3rdno`	✗

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	
`3rdno`	✗		

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗		

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

  `case`

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗	`Case`

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗	`Case`	✓

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗	`Case`	✓

- cannot be a defined identifier in the C standard library

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗	`Case`	✓

- cannot be a defined identifier in the C standard library

  `printf`

Identifiers

Identifiers are used to give names to memory locations (data), and subroutines.

Rules on naming identifiers:

- consists of letters `A`–`Z`, `a`–`z`, digits `0`–`9`, or underscores `_`

`nice`	✓	`first_digit`	✓
`idno.`	✗	`email addr`	✗

- is case-sensitive

  `idno` and `Idno` are different identifiers.

- cannot begin with a digit

`friend4`	✓	`_1stdigit`	✓
`3rdno`	✗	`a_`	✓

- cannot be one of the reserved words

`case`	✗	`Case`	✓

- cannot be a defined identifier in the C standard library

`printf`	✗

Identifiers

- Variables

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.
  - Its value is **fixed** throughout the duration of the program

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.
  - Its value is **fixed** throughout the duration of the program
- Keywords[2]

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.
  - Its value is **fixed** throughout the duration of the program
- Keywords[2]
  - Keywords are reserved words of the programming language

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.
  - Its value is **fixed** throughout the duration of the program
- Keywords[2]
  - Keywords are reserved words of the programming language
  - These words have special meanings, and are used by the language for processing.

[2] https://en.cppreference.com/w/c/keyword

Identifiers

- Variables
  - Variable can hold a value.
  - Its value **may change** at any point within the duration of the program.
- Constants
  - Constant can hold a value.
  - Its value is **fixed** throughout the duration of the program
- Keywords[2]
  - Keywords are reserved words of the programming language
  - These words have special meanings, and are used by the language for processing.
  - These words **cannot** be redefined by the programmer.

[2] https://en.cppreference.com/w/c/keyword

☺ Thank you! ☺