C Token

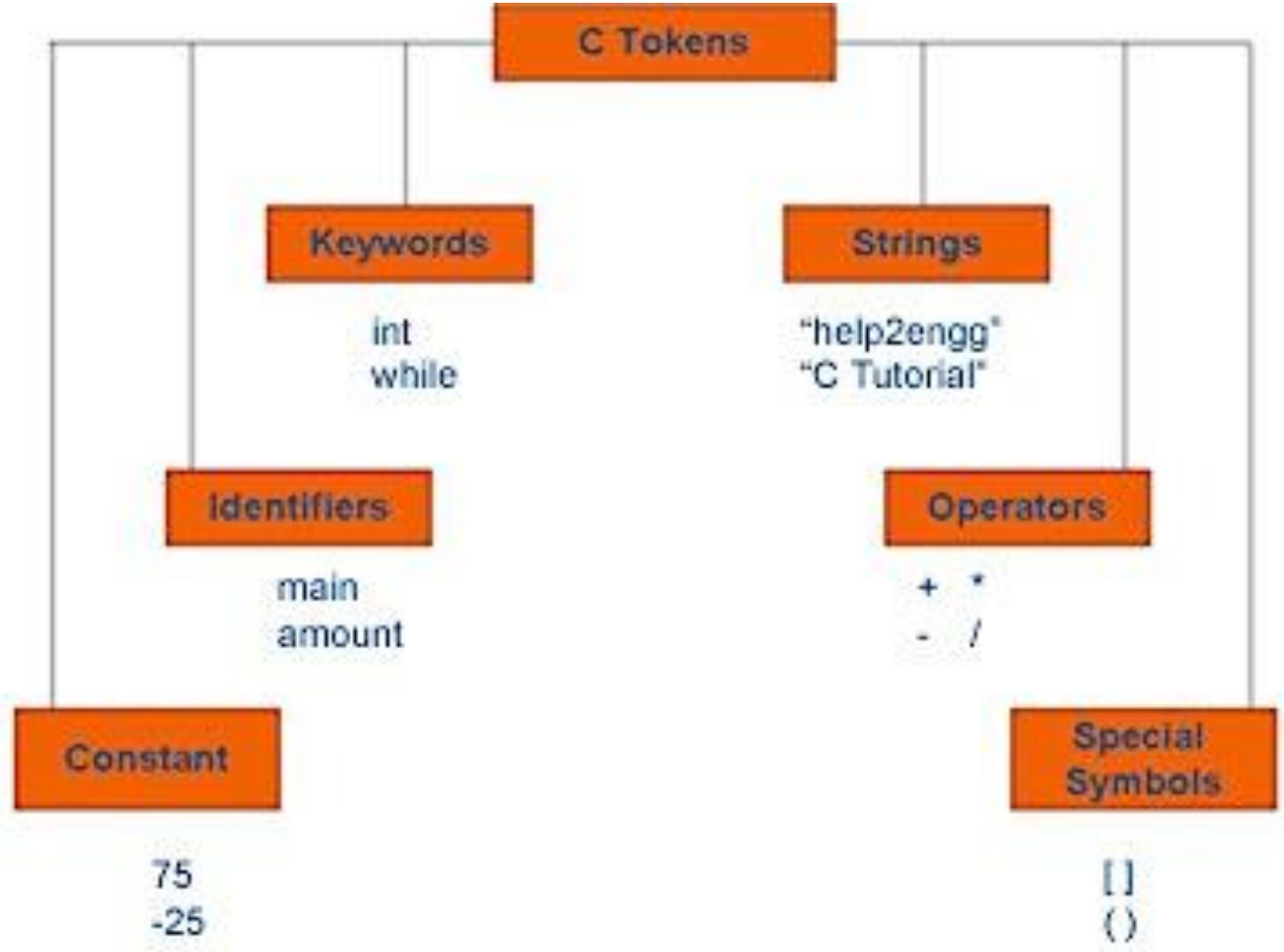
Professor Dr. M. Ismail Jabiullah
Professor
Department of CSE
Daffodil International University
Bangladesh

C tokens

- C tokens are the basic buildings blocks in C language which are constructed together to write a C program.
- Each and every smallest individual units in a C program is known as C token or a lexical unit.
- C tokens can be classified in six types as follows:

①	Keywords	(e.g., int, while, do …),
②	Identifiers	(e.g., main, total, my_var …),
③	Constants	(e.g., 10, 20 , - 25.5 …),
④	Strings	(e.g., “total”, “hello” , “DIU” …),
⑤	Special symbols	(e.g., (), {} , [] …),
⑥	Operators	(e.g., +, /, - , * …)

C tokens


C Tokens
Keywords
int
while
Strings
"help2engg"
"C Tutorial"
Identifiers
main
amount
Operators
+ *
- /
Constant
75
-25
Special Symbols
[]
()

C tokens example program

```
int main()
{
    int x, y, total;
    x = 10, y = 20;
    total = x + y;
    printf("Total %d \n", total);
    return 0;
}
```

- where,
  - Main, x, y, total – identifier
  - {,}, (,) – special symbols
  - Int , return – keyword
  - 10, 20 – constant
  - =,+ – operator
  - "Total 30" – String
  - main, {, }, (,), int, x, y, total etc– all these are various tokens

C Keywords

- There are some reserved words in C language whose meaning are predefined in C compiler, those are called C keywords.
- Each keyword is meant to perform a specific function in a C program.
- Each keywords has fixed meaning and that cannot be changed by user.
- We cannot use a keyword as a variable name.
- Since upper case and lowercase characters are not considered same in C, we can use an uppercase keyword as an identifier. But it is not considered as good programming practice.

Keywords

- There are 32 keywords in C which are given below. keywords are all lowercase.

Keywords in C Language

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
continue	for	signed	void
do	if	static	while
default	goto	sizeof	volatile
const	float	short	unsigned

Identifiers

- Each program elements in a C program are given a name called identifiers.
- Names given to identify variables, functions etc. are examples for identifiers.
  - e.g., ***int x ;*** here **x** is a name given to an integer variable.
- **Rules for constructing identifier name in C:**
  - An identifier can be composed of letters (both uppercase and lowercase letters), digits and underscore only.
  - The first character of identifier should be either a letter or an underscore(not any digit). But, it is discouraged to start an identifier name with an underscore though it is legal. It is because, identifier that starts with underscore can conflict with system names. In such cases, compiler will complain about it.
  - Punctuation and special characters are not allowed except underscore.
  - Identifiers should not be keywords.
  - Identifiers are case sensitive.
  - There is no rule for the length of an identifier. However, the first 31 characters of an identifier are discriminated by the compiler. So, the first 31 letters of two identifiers in a program should be different.

Constants

- Constants in C refer to fixed values that do not change during the execution of a program.
- Example: 1, 2.5 , "Programming is fun." etc. are the example of constants.
- In C, constants can be classified as follows:
  - Numeric constants
    - Integer constant (Ex: 102, - 5)
    - Real constant (Ex: 3.14, 5.5)
  - Character constants
    - Single character constants (Ex: ´A` , ´;` , ´5`)
    - String constants (Ex: “Hello” , “5+4”)
  - Backslash character constants (Ex: \n, \r)

Integer constants

- Integer constants are the numeric constants (constant associated with number) without any fractional part or exponential part.
- There are three types of integer constants in C language:
  - decimal constant(base 10),
  - octal constant(base 8) and
  - hexadecimal constant(base 16).
- For example:
  - Decimal constants: 0, -9 , 22 etc
  - Octal constants: 021, 077, 033 etc
  - Hexadecimal constants: 0x7f, 0x2a, 0x521 etc
- Note: Every octal constant starts with 0 and hexadecimal constant starts with 0x in C programming.

Real Constants

- Real constant, also called Floating point constants are the numeric constants that has either fractional form or exponent form.
- For example:
  - -2.0
  - 0.0000234
  - -0.22E-5
- **Note:** Here, E-5 represents 10^{-5}. Thus, -0.22E-5 = -0.0000022.

Single Character Constants

- Single character constants are the constant which use single quotation around characters.
- For example: 'a', 'l', 'm', 'F' etc.
- All character constants have an equivalent integer value which are called ASCII Values.

String constants

- A string is a sequence of characters enclosed in double quotes.
- The sequence of characters may contain letters, numbers, special characters and blank spaces.
- String constants are the constants which are enclosed in a pair of double-quote marks.
- For example:
  - "good" // string constant
  - "" // null string constant
  - " " // string constant of six white space
  - "x" // string constant having single character.

Backslash Character Constant

- Sometimes, it is necessary to use newline(enter), tab, quotation mark etc. in the program which either cannot be typed or has special meaning in C programming.
- In such cases, backslash character constant (escape sequence) are used.
- For example: \n is used for newline.
- The backslash(\ ) causes "escape" from the normal way the characters are interpreted by the compiler.

List of Escape Sequences

Escape Sequences

Escape Sequences	Character
\b	Backspace
\f	Form feed
\n	Newline
\r	Return
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\'	Single quotation mark
\"	Double quotation mark
\?	Question mark
\0	Null character

Special Symbols

- The following special symbols are used in C having some special meaning and thus, cannot be used for some other purpose.
- [] () {} , ; : * ... = #
- **Braces{}:** These opening and ending curly braces marks the start and end of a block of code containing more than one executable statement.
- **Parentheses():** These special symbols are used to indicate function calls and function parameters.
- **Brackets[]:** Opening and closing brackets are used as array element reference. These indicate single and multidimensional subscripts.

Operators

- C operators are symbols that triggers an action when applied to C variables and other objects. The data items on which operators act upon are called operands.
- Depending on the number of operands that an operator can act upon, operators can be classified as follows:
  - **Unary Operators:** Those operators that require only single operand to act upon are known as unary operators.
  - **Binary Operators:** Those operators that require two operands to act upon are called binary operators.
  - **Ternary Operators:** These operators requires three operands to act upon.
- There are many operators, some of which are single characters ~ ! @ % ^ & * - + = | / : ? < >
- While others require two characters ++ -- << >> <= += -= *= /= == |= %= &= ^= || && !=
- Some even require three characters <<= >>=
- The multiple-character operators can not have white spaces or comments between the characters.