

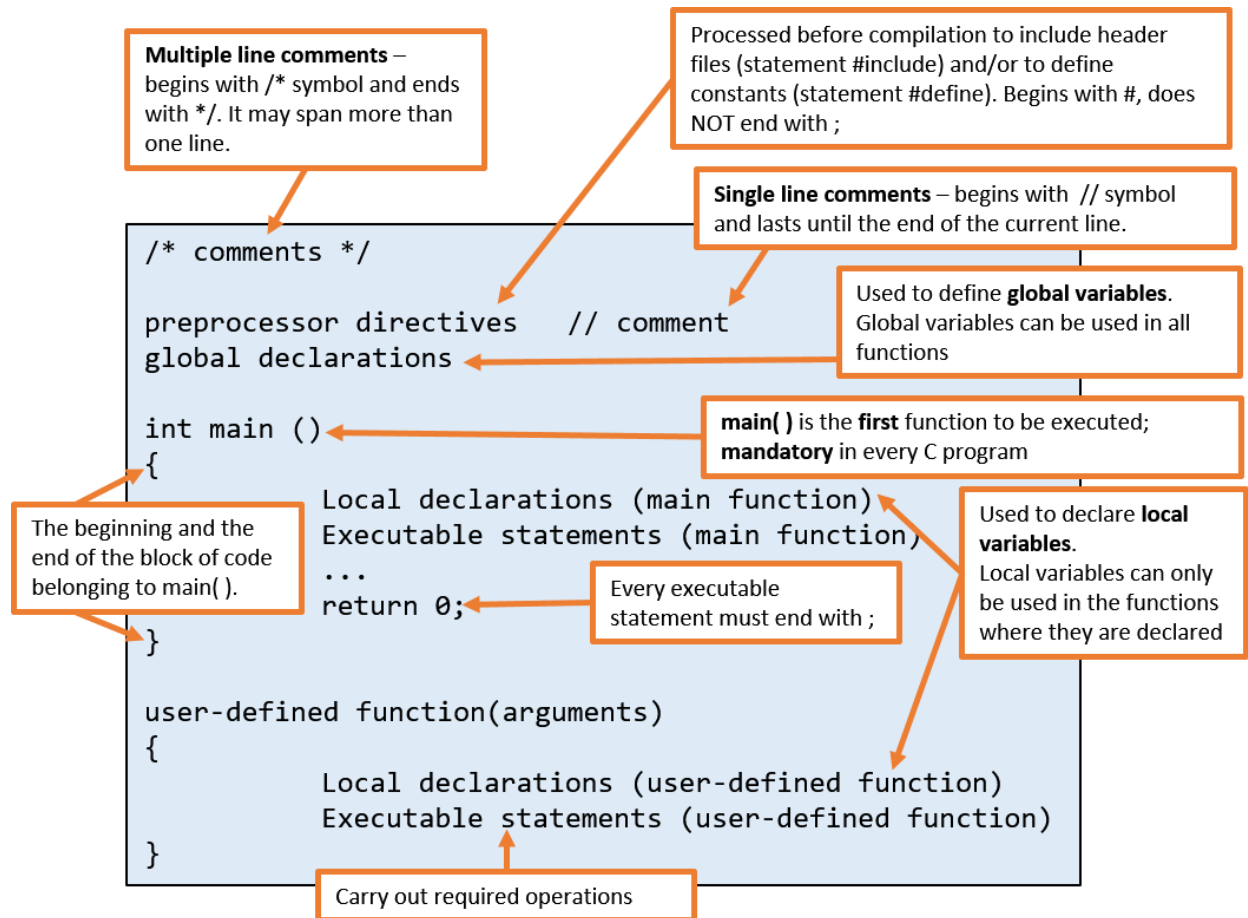
Lecture 1. C Programming Language

- Developed at AT&T Laboratories in 1972 by Dennis Ritchie for the UNIX operating system
- Building block for many known programming languages
- Used for:
 - Operating systems
 - Compilers
 - Assemblers
 - Editors
 - Network drivers
 - Utilities
 - Embedded systems
 - General-purpose programs

Features of C programming language

- C is a structured programming language
- C is highly portable language – C program is platform independent. This applies to both, choice of operating system and hardware platform.
- C brings together the features of high-level programming languages and the bit manipulation capability.
- C is fast – Compilation and execution are faster than with most other programming languages.
- C can extend itself by adding functions to its collection of build-in functions.
- C is easy to learn – It has only 32 keywords.

Structure of a C program



- Comments are optional; they are ignored by the compiler
- Preprocessor directive `#include` is used to access header files (files with extension `.h`) which contain a set of macro definitions and declarations of common C functions (build-in function). The actual executable code is stored in C libraries.
- It can be **only one** `main()` function in every C program.
- Global/local declarations state the program's need for memory.
- `{ }` are used to group statements together and to define the body of the function. For every open bracket `{`, there must be a closing bracket `}`.
- Statement `return 0;` is used to end the program and to return 0 to the operating system indicating normal termination. Non-zero value indicates abnormal termination and it is usually 1.
- C is case-sensitive language

C Language Elements

C Character Set

Alphabets	Uppercase: A – Z Lowercase: a – z
Digits	0 – 9
Special characters	- ~ ' ! @ # % ^ & * () _ - + = \ { } [] : ; " ' < > , . ? /
White space characters	blank space, new line, horizontal tab, carriage return, form feed

C Keywords (Reserved Words)

- All keywords are in lowercase letters
- Cannot be used for identifiers (user-defined names)

S.No	Keyword	Meaning
1	auto	Used to represent automatic storage class
2	break	Unconditional control statement used to terminate switch & looping statements
3	case	Used to represent a case (option) in switch statement
4	char	Used to represent character data type
5	const	Used to define a constant
6	continue	Unconditional control statement used to pass the control to the beginning of looping statements
7	default	Used to represent a default case (option) in switch statement
8	do	Used to define do block in do-while statement
9	double	Used to present double datatype
10	else	Used to define FALSE block of if statement
11	enum	Used to define enumerated datatypes
12	extern	Used to represent external storage class
13	float	Used to represent floating point datatype
14	for	Used to define a looping statement
15	goto	Used to represent unconditional control statement
16	if	Used to define a conditional control statement
17	int	Used to represent integer datatype
18	long	It is a type modifier that alters the basic datatype
19	register	Used to represent register storage class
20	return	Used to terminate a function execution
21	short	It is a type modifier that alters the basic datatype
22	signed	It is a type modifier that alters the basic datatype
23	sizeof	It is an operator that gives size of the memory of a variable
24	static	Used to create static variables - constants
25	struct	Used to create structures - Userdefined datatypes
26	switch	Used to define switch - case statement
27	typedef	Used to specify temporary name for the datatypes
28	union	Used to create union for grouping different types under a name
29	unsigned	It is a type modifier that alters the basic datatype
30	void	Used to indicate nothing - return value, parameter of a function
31	volatile	Used to creating volatile objects
32	while	Used to define a looping statement

“32 Keywords in C Programming Language with their Meaning.” n.d. Online Image. BSC btechsmartclass.
23 Jan, 2017. <http://www.btechsmartclass.com/CP/c-keywords.htm>

Identifiers

Identifier is a user defined name given to variable, function, constant and/or other program entity.

Rules for creating identifiers:

- Identifiers are case sensitive (varName is not the same as varname)
- Can only consist of letters (uppercase and lowercase), digits, and underscore (_)
- Cannot start with a digit (0-9)
- Underscore can be used as the first character (example: _result)
- Cannot contain any special character other than underscore (_)
- Cannot use reserved words (keywords)
- The identifier length limit is compiler dependent.
- Given names should be descriptive, meaningful, and unique.

Function printf()

printf() is a build-in C function used to perform output operations such as displaying data on the screen, sending data to printer or file.

printf() is defined in "stdio.h" header file. When printf() is used in the program, the "stdio.h" header must be included as follows:

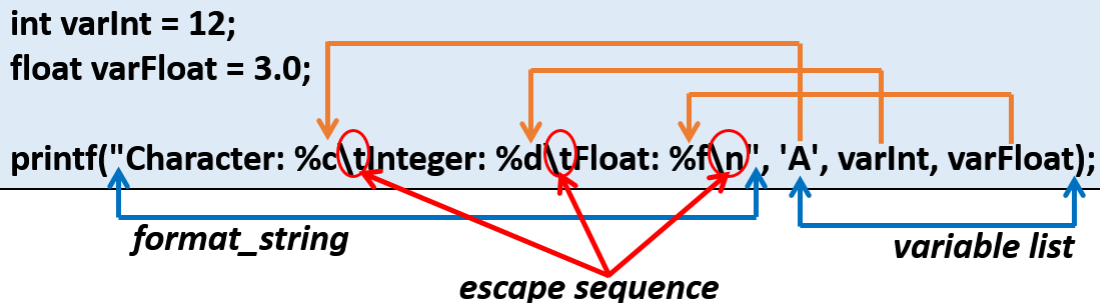
```
#include<stdio.h>
```

Syntax: `printf("format_string", variable1, variable2, variable3, ...);`

where

format_string is a string that can contain 3 types of elements:

- Plain characters that will be displayed on the screen (unchanged)
- **Format specifiers** (conversion specifiers) used as placeholders, which will be replaced by the values of variables listed after the *format_string*.
 - Format specifier begins with %, followed by the data type code. Data type code must match the variable data type. Example: %c for a variable of character data type
 - For each listed variable, there should be one format specifier.
 - Format specifiers are replaced in sequential order.
- **Escape sequences** used to control the cursor or insertion point



The diagram shows a code snippet with annotations. The code is: `int varInt = 12; float varFloat = 3.0; printf("Character: %c\tInteger: %d\tFloat: %f\n", 'A', varInt, varFloat);`. Annotations include: orange arrows pointing from `varInt` and `varFloat` to their respective format specifiers `%d` and `%f`; a blue arrow pointing from the string literal to the label *format_string*; and a red arrow pointing from the `\t` escape sequence to the label *escape sequence*.

Output:
Character: A Integer: 12 Float: 3.000000

Examples:

```
int size = 7;
float average = 80.5;
printf("Welcome!\n");
printf("a=%d b=%d\n\n", 3, size);
printf("Test Results:\nAverage=%.2f", average);
```

Output:
Welcome!
a=3 b=7

Test Results:
Average =
80.50

Format Specifiers

Format specifier	Description	Supported data types
%c	Character	char unsigned char
%d	Signed Integer	short unsigned short int long
%e or %E	Scientific notation of float values	float double
%f	Floating point	float
%g or %G	Similar as %e or %E	float double
%i	Signed Integer	short unsigned short int long
%l or %ld or %li	Signed Integer	long
%lf	Floating point	double
%Lf	Floating point	long double
%lu	Unsigned integer	unsigned int unsigned long
%o	Octal representation of Integer.	short unsigned short int unsigned int long
%p	Address of pointer to void	void *
%s	String	char *
%u	Unsigned Integer	unsigned int unsigned long
%x or %X	Hexadecimal representation of Unsigned Integer	short unsigned short int unsigned int long
%%	Prints % character	

Escape Sequences

Escape Sequence	Name	Description
\a	Alert	Sounds a beep
\b	Back space	Backs up 1 character
\f	Form feed	Starts a new screen of page
\n	New line	Moves to the beginning of next line
\r	Carriage return	Moves to the beginning of current line
\t	Horizontal tab	Moves to the next tab position
\v	Vertical tab	Moves down a fixed amount
\\	Back slash	Prints a back slash
\'	Single quotation	Prints a single quotation
\"	Double quotation	Prints a double quotation
\?	Question mark	Prints a question mark

References

- Tan, H.H., and T.B. D'Orazio. *C Programming for Engineering & Computer Science*. USA: WCB McGraw-Hill. 1999. Print.
- Hock-Chuan, Chua. *C programming Tutorial*. Programming notes, n.d. Web. 23 Jan, 2017. <https://www3.ntu.edu.sg/home/ehchua/programming/index.html>
- BSC btechsmartclass. n.d. Web. 23 Jan, 2017. <http://www.btechsmartclass.com/CP/computer-languages.htm>