DECLARATIONS

Character Set, Keywords, Identifiers, Constants, Variables

Character Set

- C uses the uppercase letters A to Z.
- C uses the lowercase letters a to z.
- C uses digits 0 to 9.
- C uses certain Special Characters.

Character Set

- The special characters are listed below:-

+	-	*	/	=	%	&	#
!	?	^	“	‘	~	\	|
<	>	(	)	[	]	{	}
:	;	.	,	_	WHITE SPACE		

Most versions of the language also allow certain other characters, such as @ and $, to be included within strings and comments.

White space includes new lines, tabs, space.

Identifiers

- Identifiers are the names that are given to various program elements, such as variables, functions and arrays.

- Identifiers consist of letters and digits, in any order, except that *the first character must be a letter*. Both UPPER and lowercase letters are permitted.

Identifiers

- Note: Upper case and lowercase letters are not interchangeable (i.e., an uppercase letter is not equivalent to the corresponding lowercase letter.)
- The underscore (_) character can also be included, and is considered to be a letter. An underscore is often used in the middle of an identifier. An identifier may also begin with an underscore.

Valid – Invalid Identifiers

- The following names are valid Identifiers:

x	y12	sum_1	_tempera
names	area	tax_rate	GOTO

- The following names are Invalid Identifiers:

1th	The first character must be a letter.
x”	Illegal Character (“).
rder-no	Illegal Character (-).
rror flag	Illegal Character (blank space).
goto	Reserved word (keyword).

Important - Identifiers

- Note: As identifier can be arbitrarily long. Some implementations of C recognize only the first eight characters, though most implementations recognize more (typically 31 characters).

Example - Identifiers

- The identifier file_manager and file_management are both grammatically valid. Some compilers may be unable to distinguish between them, however, because the first eight characters are same for each identifier. Therefore, only one of these identifiers should be used in a single C program.

Keywords

- There are certain reserved words, called keywords, that have standard, predefined meanings in C.

- These keywords can be used only for their intended purpose, they cannot be used as programmer defined identifiers.

Standard Keywords

- The standard keywords are:-

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

Standard Keywords

- Some compilers may also include some or all of the following keywords:

ada	**far**	**near**
fortran	**asm**	**pascal**
Entry	**huge**	

- Note that the keywords are all lowercase. Since uppercase and lowercase characters are not equivalent, it is possible to utilize an uppercase keyword as an identifier.

Constants

- There are four basic types of constants in C. They are:
  - Integer constants.
  - Floating-point constants.
  - Character constants.
  - String constants.
  - **Symbolic constants.**
- Non symbolic constants are also known as literals.

Constants

- Moreover, there are several different kinds of integer and floating-point constants, as discussed below:

  - Integer and Floating-point constants represent numbers. They are often referred to collectively as numeric – type constants.

Rules for Numeric – Type Constants

- Commas and blank spaces cannot be included within the constant.
- The constant can be preceded by minus (-) sign, if desired.
- The value of a constant cannot exceed specified minimum and maximum bounds. For each type of constant these bounds will vary from one C compiler to another.

Integer Constants

- An integer constant is an integer-valued constant. Thus, it consists of a sequence of digits.

- Integer constants can be written in three different number systems: decimal(base 10), octal(base 8) and hexadecimal(base 16).

Decimal – Integer Constant

- A decimal integer constant can consist of any combination of digits taken from the set 0 through 9. if the constant contains two or more digits, the first digit must be something other than 0.

Valid – Invalid Integer Constants

- Several valid decimal integer constants are shown below:

0	1	743	5280	32767	9999

- The following decimal integer constants are written incorrectly for the reasons stated.

12,245	Illegal Character (,).
36.0	Illegal Character (.).
10 20 30	Illegal Character (blank space).
123-45-6789	Illegal Character (-).
0900	the first digit cannot be zero

Octal – Integer Constant

- An octal integer constant can consist of any combination of digits taken from the set 0 through 7.

- However the first digit must be 0, in order to identify the constant as an octal number.

Valid – Invalid Integer Constants

- Several valid octal integer constants are shown below:

0	01	0743	077777

- The following octal integer constants are written incorrectly for the reasons stated.

743	Does not begin with 0.
05280	Illegal digit (8).
0777.777	Illegal character (.).

Hexadecimal – Integer Constant

- A hexadecimal integer constant must begin with either 0x or 0X. It can then be followed by any combination of digits taken from 0 to 9 and a through f (either upper or lowercase).

- Note that the letters a through f (or A through F) represent the (decimal) quantities 10 through 15, respectively.

Valid – Invalid Integer Constants

- Several valid hexadecimal integer constants are shown below:

0x	0X1	0X7FFF	0xabcd

- The following hexadecimal integer constants are written incorrectly for the reasons stated.

0X 12.34	Illegal character (.).
0BE38	Does not begin with 0x or 0X.
0x.4bff	Illegal character (.).
0XDEFG	Illegal character (G).

Floating-Point Constants

- A floating-point constant is a base-10 number either a decimal point or an exponent (or both).
- The interpretation of a floating-point constant with an exponent is essentially the same as scientific notation, except that the base 10 is replaced by the letter E or e. Thus, the number 1.2×10^{-3} would be written as 1.2E-3 or 1.2e-3. This is equivalent to 0.12e-2,or 12e-4, etc.

Valid – Invalid Floating-point Constants

- Several valid floating-point constants are shown below:

0.	1.	0.2	827.602
50000.	0.000743	12.3	315.0066
2E-8	0.006e-3	1.6667E+8	.12121212e12

- The following are not valid floating-point constants for the reasons stated.

1	Either a decimal point or an exponent must be present.
1,000.0	Illegal character (,).
2E+10.2	The exponent must be an Integer quantity.
3E 10	Illegal character (blank space) in the exponent.

Character Constants

- A character constant is a single character, enclosed in apostrophes (i.e., single quotation marks).
- Character constants have integer values that are determined by the computer's particular character set.
- Most computers, and virtually all personal computers, make use of ASCII (i.e., American Standard Code for Information Interchange) character set.

Valid Character Constants

- Several valid character constants are shown below:

‘A’	‘x’	‘3’	‘ ‘ Blank space

- Several character constants and their corresponding values, as defined by the ASCII character set, are shown below:

Constant	Value
‘A’	65
‘x’	120
‘3’	51

Constant	Value
‘?’	63
‘ ’	32

String Constants

- A string constant consists of any number of consecutive characters (including none), enclosed in (double) quotation marks.

Valid String Constants

- Several valid string constants are shown below:

“green”	“Washington, D.C. 20005”	“270-32-3456”
“$19.95”	“THE CORRECT ANSWER IS:”	“2*(I+3)/J”
“ “	“Line 1\n Line 2\n Line 3”	“ “

- Note that the string constant “Line 1 \n Line 2\n Line 3”extends over three lines because of newline characters that are embedded within the string.

Symbolic Constants

- A symbolic constant is a name that substitutes for a sequence of characters. The characters may represent a numeric constant, a character constant or a string constant.
- Thus a symbolic constant allows a name to appear in place of a numeric constant, a character constant or a string.
- When a program is compiled, each occurrence of a symbolic constant is replaced by its corresponding character sequence.

Symbolic Constants

- Symbolic constants are usually defined at the beginning of a program.
- A symbolic constant is defined by writing

```
# define name text
```

Or

```
const  data-type name = value;
```

where name represents a symbolic name, typically written in uppercase letters, and text represents the sequence of characters that is associated with the symbolic name.

Variables

- A variable is an identifier that is used to represent some specified type of information within a designated portion of the program.

- In its simplest form, a variable is an identifier that is used to represent a single data item i.e., a numerical quantity or character constant.

Variables

- A given variable can be assigned different data items at various places within the program.
- Thus, the information represented by the variable can change during the execution of the program.
- However, the data type associated with the variable cannot change.

Escape Sequences

- Certain non printing characters, as well as the backlash (\ ) and the apostrophe (‘), can be expressed in the terms of escape sequences or backslash character constants.

Escape Sequences

- An escape sequence always begin with a backward slash and is followed by one or more special characters.
- For example, a line feed (LF), which is referred to as a newline in C can be represented as \n.
- Such escape sequences always represent single characters, even though they are written in terms of two or more characters.

Escape Sequences

- The following are the commonly used escape sequences:

Character	Escape Sequence	ASCII Value
bell (alert)	\a	007
backspace	\b	008
horizontal tab	\t	009
vertical tab	\v	011
newline (line feed)	\n	010
form feed	\f	012
carriage return	\r	013
quotation mark (“)	\”	034
apostrophe	\’	039
question mark (?”	\?	063
backslash	\\	092
null	\0	000

Practice Questions

- List 10 valid and 10 invalid identifiers other than those mentioned in your textbook and this presentation.
- Separate the valid decimal, octal and hexadecimal literals from the following:-
  - 00.5, 0.0.5, 0.8E+0.8, 786, 13B, 0xLPU, 0x87e3fa
- Give the alternative declaration of following symbolic constants:
  - # define CSE101 C
  - const int CSE = 101
  - const char city =‘J’;
  - int const city =‘N’;

Practice Questions

- Comment on the following code:

  – #define int 100

  and **how does this declaration behave for the following**:-

  ✓ int x;

  ✓ int;

  ✓ printf(“%d”, x);