C Programming

Trainer : Nisha Dingare

Email : nisha.dingare@sunbeaminfo.com

Escape Sequence :

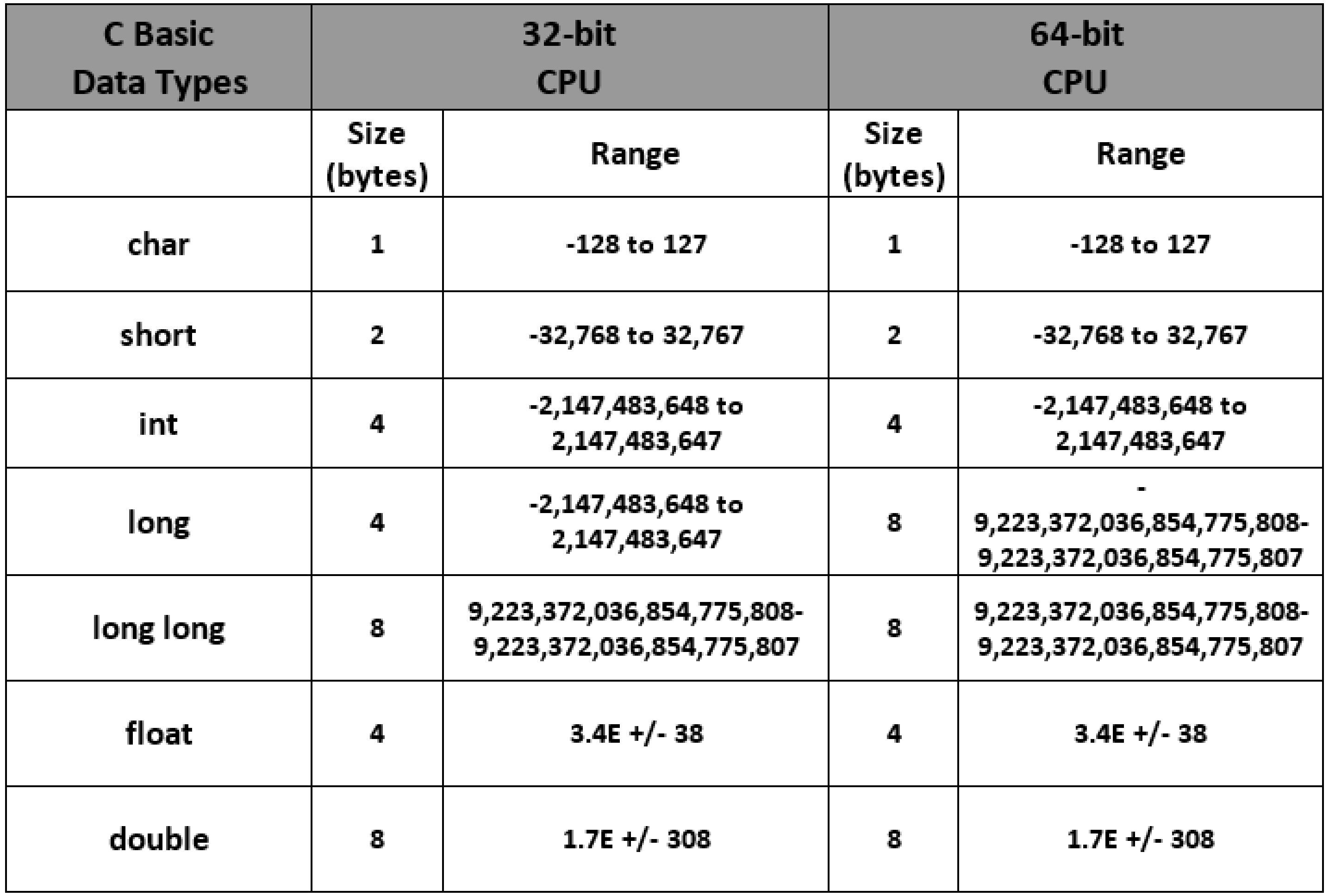
- \n - New line - Moves cursor to the beginning of next line.
- \t – Horizontal tab - Moves the cursor horizontally 8 characters ahead.
- \r - Carriage return - Moves the cursor to the beginning of current line.
- \b - Backspace - Moves cursor one position to the left of its current position.
- \f - Form Feed - Moves cursor to the beginning of next page (Used in context of printer).
- \a -Alert - Alerts by Generating beep.

- \’ - Single quote mark - prints ’
- \” - Double quote mark - prints”
- \\- Backslash Character - Prints\

Type Modifiers :

- **char**
  - signed char (-128 to 127)
  - unsigned char (0 to 255)
- **int/ long (32-bit)**
  - signed int(-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647)
  - unsigned int(0 to 65,535 or 0 to 4,294,967,295)
- **short int**
  - signed short (-32,768 to 32,767)
  - unsigned short(0 to 65,535)
- **long long/ long (64-bit)**
  - signed long (-9223372036854775808 to 9223372036854775807)
  - unsinged long(0 to 18446744073709551615)
- float: ±3.4E +/-38
- double: ±1.7E +/-308

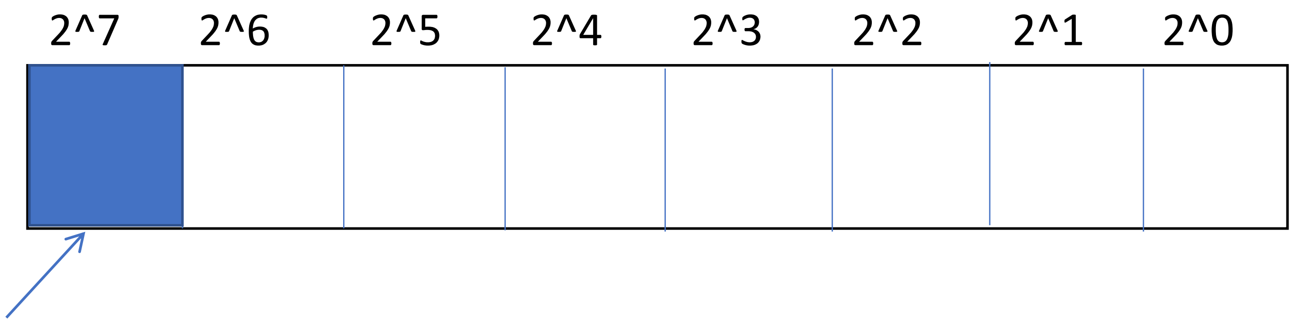
Type Modifiers :



C Basic Data Types	32-bit CPU		64-bit CPU	
	Size (bytes)	Range	Size (bytes)	Range
char	1	-128 to 127	1	-128 to 127
short	2	-32,768 to 32,767	2	-32,768 to 32,767
int	4	-2,147,483,648 to 2,147,483,647	4	-2,147,483,648 to 2,147,483,647
long	4	-2,147,483,648 to 2,147,483,647	8	- 9,223,372,036,854,775,808- 9,223,372,036,854,775,807
long long	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807
float	4	3.4E +/- 38	4	3.4E +/- 38
double	8	1.7E +/- 308	8	1.7E +/- 308

Signed / unsigned char :

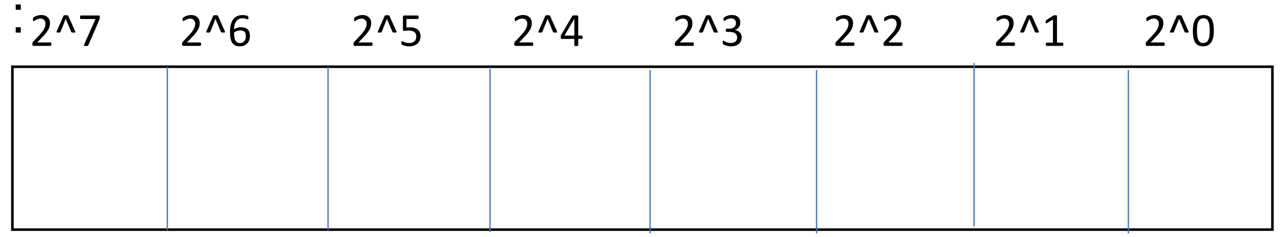
- Signed char :



MSB
Most Significant Bit – sign bit.
Used for storing sign in case of signed datatype

Remaining 7 bits are used for data.
2^7

- Unsigned char :



MSB
Most Significant Bit – Not a sign bit

All 8 bits are used for data. 2^8

Constants :

- **Numeric constants**
  - **Integer constants**
    - Decimal
    - Octal
    - Hexadecimal
  - **Real constants**
    - floating point form
    - exponential form

- **Character constants**
  - Single character constants
  - String constants

Constants :

- **Integer constant :**
  - If integer constant starts with 0 it is assumed to be in octal number system.
  - If integer constant starts with 0x or 0X then it is assumed to be in hexadecimal number system 0x23 or 0X23 is valid which means 23 in hexadecimal number system is equivalent to 19 in decimal.
  - If integer constant has suffix character l/L then it is assumed as Long , if it is terminated with u or U then it is assumed as unsigned int. 23l is a long integer , 23u is a unsigned integer .
- **Floating point constants :**
  - Must have at least one digit, it can be either positive or negative by default it is positive.
  - It should have decimal point.
  - Floating point by default is considered to be double eg.: 23.45 is double.
  - Floating point if suffixed with f/ F is considered as float eg.: 23.45f is a float.

Typecasting :

- Implicit typecasting :
    - Eg : int num = 12.34;
        - Double type value is implicitly converted to integer type. (downcasting).

- Explicit typecasting :
    - Eg : float result = (float) 5/3;
        - Integer calculation is explicitly converted to float.

scanf :

- Scanf is a standard library function which stands for scan formatted string.
- It is used to receive the input from the user.
- It requires the stdio header file.
- While scanning the input, scanf needs to store that input data somewhere. To store this input data, scanf needs to known the memory location of a variable. And here comes the ampersand (&)/address of op to rescue.
- It should have one format specifier for each address passed and types should match.
- Format : scanf(“format string”,&value);
- Do not use any character other than format specifiers in format string.
- Example : int num;

  scanf(“%d”,&num);
- To skip a character from input, use %*c.

Types of Errors :

- Compile time Error :
  - Errors that occur when you violate the syntax writing rules are compile time errors. These errors are detected by the compiler, hence known as compile time errors.
- Run time Error :
  - Errors that occur during program execution and after successful compilation are called runtime errors.
  - These types of error are hard to find as the compiler doesn't point to the line at which the error occurs. E.g. Divide by 0.
- Linker Error :
  - Linker errors occur when the linker is trying to put all the pieces of a program together to create an executable, and one or more pieces are missing. Typically, this can happen when an object file or libraries can't be found by the linker.
- Logical Error :

  Logical error occurs when the mentioned logic does not provide expected output. The program can run but does not do what it is expected to do.

Thank You