DIFFERENCE TABLE – 5

Difference between 'Macro' and 'Function'

S. No.	Attributes	MACRO	FUNCTION
1	**Processing**	Macro is Preprocessed	Function is Compiled
2	**Type Checking**	No	Yes
3	**Code length**	Increases	Unaffected
4	**Side effect**	Yes	No
5	**Speed**	Faster	Slower
6	**Usefulness**	When small code is repeated many times	When large code is to be written
7	**Compile-Time Errors**	Does not check	It checks

Difference between 'Object-like-macro' and 'Function-like-Macro'

S. No.	OBJECT-LIKE-MACRO	FUNCTION-LIKE-MACRO
1	An object-like macro is an identifier that will be replaced by a sequence of token (or piece of code) in the program.	C also allows us to define a macro that look like a function call. Therefore, this macro is referred as a function-like macro.
2	As its name implied, the object-like macro is similar to data object in code in term of its usage. We typically use an object-like macro to give a meaningful name to a constant.	The syntax of creating a function-like macro is similar to object-like macro except you have to put the parentheses () right after the macro name.
3	**Syntax:** #define identifier string	**Syntax:** #define identifier (arg1, arg2, … argn) string
4	**Example:** #define MAX_SIZE 1000	**Example:** #define min(a, b) ((a) < (b) ? (a) : (b))

Difference between 'Array' and 'Pointer'

S. No.	Attributes	ARRAY	POINTER
1	**Syntax**	data_type array_name[max_size];	data_type *pointer_var;
2	**Example**	int arr[5];	int *p;
3	**Working**	Stores the value of the variable of homogeneous datatype in adjacent memory locations	Store the address of the another variable of same datatype

4	**Generation**	An array of pointers can be generated.	A pointer to an array can be generated.
5	**Capacity**	An array can store the number of elements, mentioned in the size of array variable.	A pointer variable can store the address of only one variable at a time.