Programming in C

Tushar B. Kute,
http://tusharkute.com

Strings

- The string can be defined as the one-dimensional array of characters terminated by a null ('\0').
- The character array or the string is used to manipulate text such as word or sentences.
- Each character in the array occupies one byte of memory, and the last character must always be 0.
- The termination character ('\0') is important in a string since it is the only way to identify where the string ends.
- When we define a string as char s[10], the character s[10] is implicitly initialized with the null in the memory.

Strings

- There are two ways to declare a string in c language.
  - By char array
  - By string literal
- Let's see the example of declaring string by char array in C language.
  - char ch[10]={'t', 'u', 's', 'h', 'a', 'r', '\0'};

String Functions

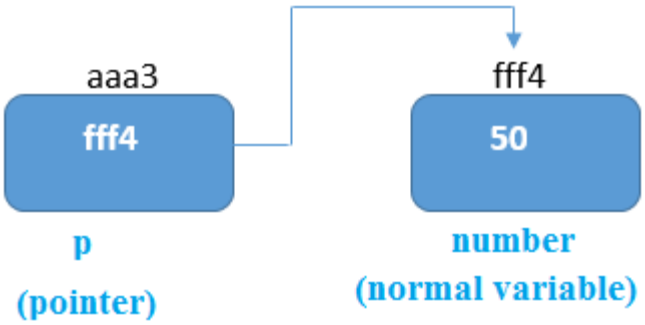
No.	Function	Description
1)	strlen(string_name)	returns the length of string name.
2)	strcpy(destination, source)	copies the contents of source string to destination string.
3)	strcat(first_string, second_string)	concats or joins first string with second string. The result of the string is stored in first string.
4)	strcmp(first_string, second_string)	compares the first string with second string. If both strings are same, it returns 0.
5)	strrev(string)	returns reverse string.
6)	strlwr(string)	returns string characters in lowercase.
7)	strupr(string)	returns string characters in uppercase.

Pointers

- The pointer in C language is a variable which stores the address of another variable.
- This variable can be of type int, char, array, function, or any other pointer.
- The size of the pointer depends on the architecture. However, in 32-bit architecture the size of a pointer is 2 byte.
- Consider the following example to define a pointer which stores the address of an integer.
  - int n = 10;
  - int* p = &n; // Variable p of type pointer is pointing to the address of the variable n of type integer.

Pointers

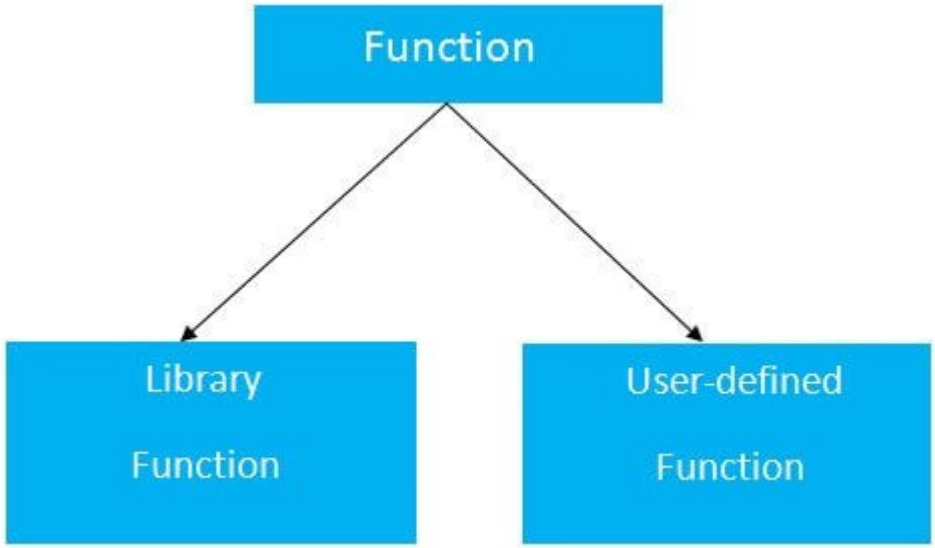
- The pointer in c language can be declared using * (asterisk symbol).
- It is also known as indirection pointer used to dereference a pointer.
  - int *a;//pointer to int
  - char *c;//pointer to char



Functions

- In c, we can divide a large program into the basic building blocks known as function.
- The function contains the set of programming statements enclosed by {}. A function can be called multiple times to provide reusability and modularity to the C program.
- In other words, we can say that the collection of functions creates a program.
- The function is also known as procedureor subroutinein other programming languages.

Functions



Functions

- Function declaration A function must be declared globally in a c program to tell the compiler about the function name, function parameters, and return type.
- Function call Function can be called from anywhere in the program. The parameter list must not differ in function calling and function declaration. We must pass the same number of functions as it is declared in the function declaration.
- Function definition It contains the actual statements which are to be executed. It is the most important aspect to which the control comes when the function is called. Here, we must notice that only one value can be returned from the function.

Functions

- Function declaration
  - return_type function_name (argument list);
- Function call
  - function_name (argument_list)
- Function definition
  - return_type function_name (argument list) {function body;}

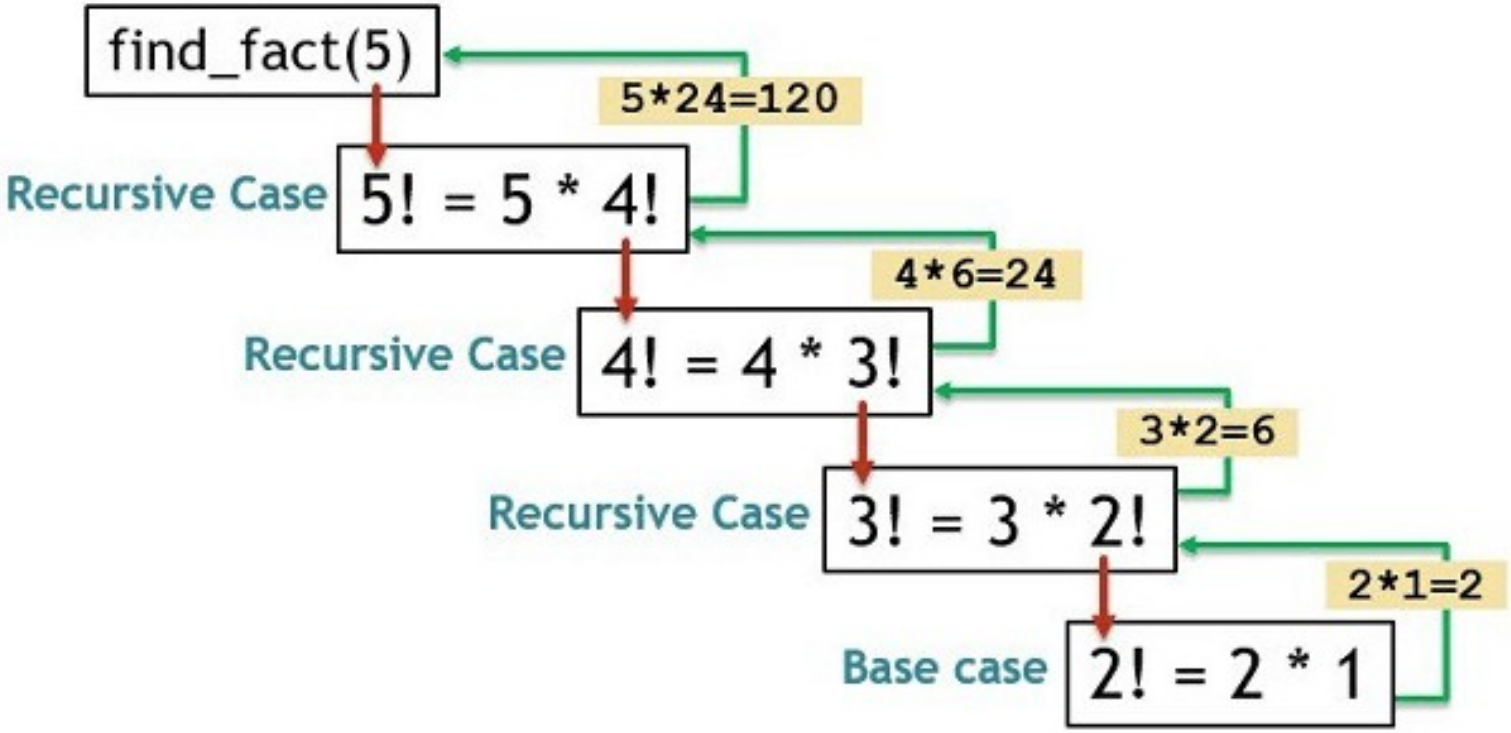
Recursion

- Recursion is the process which comes into existence when a function calls a copy of itself to work on a smaller problem.
- Any function which calls itself is called recursive function, and such function calls are called recursive calls.
- Recursion involves several numbers of recursive calls. However, it is important to impose a termination condition of recursion.
- Recursion code is shorter than iterative code however it is difficult to understand.

Recursion

- Recursion cannot be applied to all the problem, but it is more useful for the tasks that can be defined in terms of similar subtasks.
- For Example, recursion may be applied to sorting, searching, and traversal problems.
- Generally, iterative solutions are more efficient than recursion since function call is always overhead.
- Any problem that can be solved recursively, can also be solved iteratively. However, some problems are best suited to be solved by the recursion, for example, tower of Hanoi, Fibonacci series, factorial finding, etc.

Recursion


find_fact(5)
5*24=120
Recursive Case
5! = 5 * 4!
4*6=24
Recursive Case
4! = 4 * 3!
3*2=6
Recursive Case
3! = 3 * 2!
2*1=2
Base case
2! = 2 * 1

Structure

- In C, there are cases where we need to store multiple attributes of an entity.
- It is not necessary that an entity has all the information of one type only.
- It can have different attributes of different data types. For example, an entity Student may have its name (string), roll number (int), marks (float). To store such type of information regarding an entity student, we have the following approaches:
  - Construct individual arrays for storing names, roll numbers, and marks.
  - Use a special data structure to store the collection of different data types.

Structure

- Structure in c is a user-defined data type that enables us to store the collection of different data types. Each element of a structure is called a member. Structures ca; simulate the use of classes and templates as it can store various information
- The ,struct keyword is used to define the structure. Let's see the syntax to define the structure in c.

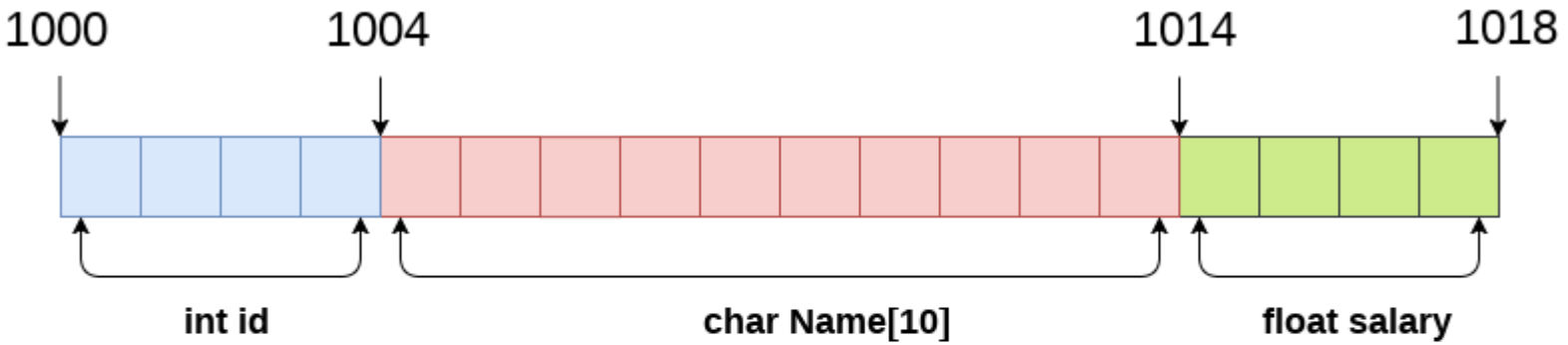
```
struct structure_name
{
   data_type member1;
   data_type member2;
   .
   .
   data_type memeberN;
};
```

Structure

- Let's see the example to define a structure for an entity employee in c.

```
struct employee
{   int id;
    char name[20];
    float salary;
};
```

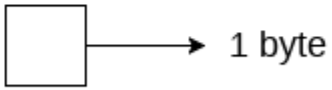
Structure




```
struct Employee
{
    int id;
    char Name[10];
    float salary;
} emp;
```

sizeof (emp) = 4 + 10 + 4 = 18 bytes

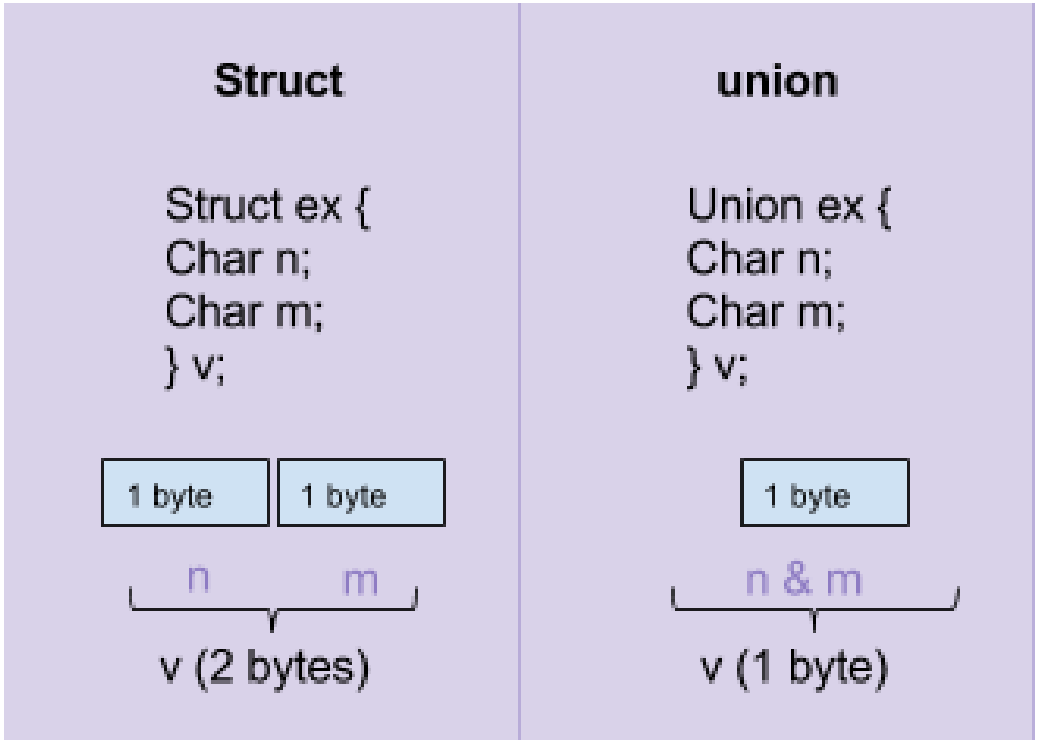
where;
sizeof (int) = 4 byte
sizeof (char) = 1 byte
sizeof (float) = 4 byte



Union

- Union can be defined as a user-defined data type which is a collection of different variables of different data types in the same memory location.
- The union can also be defined as many members, but only one member can contain a value at a particular point in time.
- Union is a user-defined data type, but unlike structures, they share the same memory location.

Union vs. Structure



Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com