Programming Languages 1
Lesson 7 - Guide

struct – the record data structure

Sometimes a programmer must use a heterogeneous complex data structure, a record. It can contain different type data fields. In the C programming language it is defined by the `struct` keyword. For example:

```
struct foo{
  int f1, f2;
  float f3;
  char x;
  char nickname[10];
};
```

Here the name of the record is `foo`, and it has 5 fields called `f1, f2, f3, x, nickname`. The fields can have different types. Record variables can be declared as well. In the following example `var1` is a `struct foo` type record variable:

```
struct foo var1;
```

A fields of a record variable can be referred by the `.` (dot) operator. For example:

```
var1.f1 = 123;
var1.x = 'A';
scanf("%f",&var1.f3);
printf("%s\n",var1.nickname);
```

Built-in string functions

Several library functions are available in the `string.h` header. If they are used, the mention header have to be included: `#include<string.h>`

Do not forget that string in C are handled by a character sequence terminated by the special '`\0`' character. Char pointers and char arrays can also refer to a string, but array always has allocated memory field for characters, however a pointer can store only the starting address of a string.

Let's suppose we have the following declared variables:

```
  int N;
  char x;
  char *P;
  char str[100]="Kings drink gin.";
  char Ar[];
```

In the rest of the section some (but not all) of the functions and procedures are introduced by examples.

```
  N=strlen(str);
```

It determines the length (number of characters) in the parameter string. Termination '`\0`' is not counted.

```
  strcpy(Ar,"Winter is coming.");
```

Copies the characters of constant string into the character array specified in the first parameter. It is also used to copy the content of a string variable into another one because assignment is does not work with strings.

```
  strcmp(Ar,str);
```

It compares the two strings given in the parameters in alphabetical order. If the parameters contains the same character sequence the function returns with 0. Otherwise, if the first parameter string is before the second one in alphabetical order it returns a negative value, else a positive integer. The ==

operator cannot be used to compare strings.

```
  strcat(Ar, str);
```

Concatenate/append the second string to the end of the first one. The `Ar` array will contain the "`Winter is coming. Kings drink gin.`" text.

```
  P=strchr(Ar,'i');
```

It searches the first occurrence of a single character (second parameter) in a string (first parameter). It returns by the address of the first occurrence of the character. (Now, it is stored in a pointer.) If the string does not contain the character at all, it returns by NULL.

```
  P=strstr(Ar, "in");
```

It returns with the address of the first occurrence of a string (second parameter) within another string (first parameter). If the string does not contain the sample string at all, it returns by NULL.

The switch instuction

The switch statement allows programmers to execute one code block among many alternatives. (Contrary to the `if-else` instruction, where there are only 2 branches.) The syntax of the instruction is the follow:

```
switch (expression){
   case Constant_1:
      Instructions_1
   case Constant_2:
      Instructions_2
   case Constant_3:
      Instructions_3
   default:
      Instructions_4
}
```

Its semantics: first the expression is evaluated, so its value will be determined. (`Instuctions_i` can be a single instruction or more instructions or a block.) If the value of the `expression` is equal to `Constant_i`, then the execution starts with the `Instructions_i` and all the remainder instructions will be executed sequentially, thus the instructions before the `Instructions_i` will be skipped. If there is no `case` label for the value of the `expression` the instructions after the `default` label (`Instructions_4`) will be executed. The `expression` just determines where to start the execution of the instruction sequence within switch.

The `break` instruction can be used within the `switch` statement. If the execution reaches the `break` instruction, then the remaining part of the `switch` will be skipped, so the next instruction after the `switch` statement will be executed. In this way, programmer can handle distinct (not overlapping) branches of instructions.

Further readings:

- Brian W. Kernighan, Dennis M. Ritchie: *The C programming language*, Prentece Hall (2012)
- https://www.tutorialspoint.com/cprogramming/c_structures.htm
- https://www.tutorialspoint.com/c_standard_library/string_h.htm
- https://www.tutorialspoint.com/cprogramming/switch_statement_in_c.htm