* **Introduction**

The origin of C is closely tied to the development of the Unix operating system. The developers were considering rewriting the original PDP-11 version of Unix system using the B language, Thompson's simplified version of BCPL. However, because of some of the B's inability, led to C. The name of C was chosen simply as the next after B.

The development of C started in 1972 on the PDP-11 Unix system and first appeared in Version 2 Unix. In 1972, a large part of Unix was rewritten in C. By 1973, with the addition of struct types, the C language became powerful and most of the Unix kernel was now in C.

|  |  |
| --- | --- |
| **Year** | **C Standard** |
| 1972 | Birth |
| 1978 | K&R C |
| 1989/1990 | ANSI C and ISO C |
| 1999 | C99 |
| 2011 | C11 |
| 2017/2018 | C18 |

Many later languages have borrowed directly or indirectly from C, including C++, C#, Java, etc. These languages have drawn many of their control structures and other basic features from C. Most of them are also very syntactically similar with C and tend to combine the recognizable expression and statement syntax of C.

Reference: https://en.wikipedia.org/wiki/C\_(programming\_language)

* **Data Type**

1. Primitive Data Type of C Language:

void, int, char, float, double

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 32-bit operating system | | 64-bit operating system | |
| **Byte** | **Bit** | **Byte** | **Bit** |
| char | **1** | **4** | **1** | **8** |
| short | **2** | **8** | **2** | **16** |
| int | **4** | **16** | **4** | **32** |
| long | **4** | **16** | **4** | **32** |
| long long | **8** | **32** | **8** | **64** |
| unsigned  int | **4** | **16** | **4** | **32** |



1. char, short, int, long, and long long can be use as keyword.

Because these types of date can be compared with the operation ==, so that they can by use as keyword in switch/case block.

1. The result of the output is 280 as I think.

My Operation System and compiler is 64-bit.

For each struct,

char name[10] -> 10 bytes to store. -> in total 10 bytes

long student\_id -> 4 bytes to store -> in total 16 bytes

char sex -> 1 byte to store -> in total 20 bytes

double score[4] -> 32 bytes to store. -> in total 56 bytes

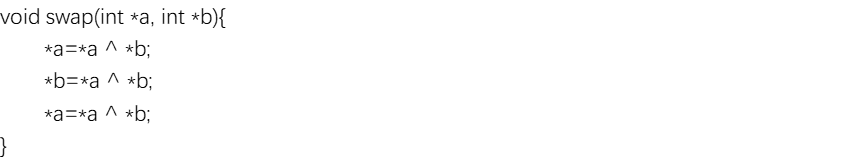
So for the output, the result should be 56\*5 = 280.

* **Operators**

|  |  |
| --- | --- |
| **Priority** | **Operators** |
| **1** | **‘[]’, ‘()’, ‘.’, ‘->’** |
| **2** | **‘++’, ‘--‘** |
| **3** | **‘%’, ‘/’, ‘\*’** |
| **4** | **‘+’, ‘-’** |
| **5** | **‘<<’, ‘>>’** |
| **6** | **‘=’, ‘+=’…** |

1. ++i means that assign i to i+1 first, then do the loop or assignment.

i++ means doing loop or assignment first, then assign i to i+1;

****

* **Arrays and Pointers**

1. char\* a is a pointer, it stores the address of the first element in char array;

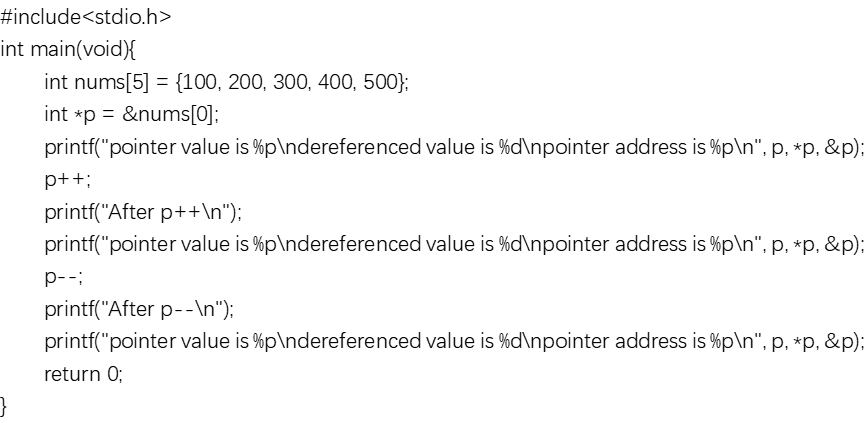
char a[] is an array, it stores the whole char array, and we get elements by using index.

The way to show their difference is using the function sizeof().

For sizeof(char\* a), it show s us the size of the pointer,

while for sizeof(char a[]) , it show s us the size of the whole char.

1. ’++’, ‘--‘, ‘=’, ‘\*’, ‘&’



* **Others**

1. In C99, there are two ways to define the main function:





As for void main(), this function is illegal for C or CPP as the main function.

1. The endian of my system is little endian.



1. strlen can return the length of the string, which end up with ‘\0’.

sizeof can return the space that the array or pointer have in the memory.









1. /\* It’s not my work, just for my review notes. \*/
   1. scanf(char \*format, 存储的地址列表);

功能：根据指定的格式获取指定类型的数据

参数：format格式控制字符串,地址列表(&) 返回：成功获取数据的个数

* 1. printf(“格式控制串”，输出表)

头文：#include

功能：按指定格式向显示器输出数据

返回：输出数据所占的数据宽度

* 1. putchar

功能：输出一个字符；

参数：传递一个整型的表达式；

返回：将成功输出字符所对应ASCI码返回

* 1. getchar

功能：从终端读取一个字符，吃掉垃圾字符；

参数：无参；

返回：将成功读取字符的ASCI码值返回

* 1. int puts(char \*s);//const：可以传递变量的地址、常量的地址

功能：打印字符串

参数：char \*传递地址

返回：输出字符所占字节数

特点：自带换行符，遇到‘\0’字符结束输出

* 1. char \*gets(char \*s); //只能传递变量的地址

功能：从终端读取字符串

参数：char \*传递地址

返回：返回s

特点：只遇到‘\n’才结束输入，相当于scanf("%[^\n]");

* 1. char \* fgets ( char \* str, int num, FILE\* stream );

功能：Reads characters from stream and stores them as a C string into str until (num- characters have been read or either a new line or the End-of-File is reached, whichever comes first. A new line character makes fgets stop reading, but it is considered a valid character and therefore it is included in the string copied to str.A null character is automatically appended in str after the characters read to signal the end of the C string.

参数：StrPointer to an array of chars w here the string read is stored.numMaximum number of characters to be read (including the final null-character). Usually, the length of the array passed as str is used streamPointer to a FILE object that identifies the stream w here characters are read from. To read from the standard input, stdin can be used for this parameter.

返回：On success, the function returns the same str parameter. If the End-of-File is encountered and no characters have been read, the contents of str remain unchanged and a null pointer is returned. If an error occurs, a null pointer is returned.

* 1. int fputs ( const char \* str, FILE\* stream );

功能：Writes the string pointed by str to the stream. The function begins copying from the address specified (str) until it reaches the terminating null character ('\0'). This final null-character is not copied to the stream.

参数：strAn array containing the null-terminated sequence of characters to be written.streamPointer to a FILE object that identifies the stream w here the string is to be written.

返回：On success, a non-negative value is returned. On error, the function returns EOF.

1. **goto** can jump out of the iteration statements.

The most special use is that it doesn’t like break or continue, while can only jump out of the iteration it in. Besides, go to can jump out of a block of codes. So it in deed a kind of method to control the order that the program runs with.

1. **malloc** allocates the specified number of bytes

**free** releases the specified block of memory back to the system

**new** and **delete** are a pair of language constructs that perform dynamic memory allocation, object construction and object destruction.