Константи. Представяне на текстова и числова информация в C/C++

Константата е програмен елемент който непроменя стойността си по време на изпълнението на програмата. C++ поддържа всички типове константи, дефинирани от K&R и поддържа следните типове константи:

- константи с плаваща запетая
- цели константи
- символни константи
- стрингови константи

Константи с плаваща запетая

Константите с плаваща запетая са десетични числа, представени във вид реално число с десетична точка или експонента. Има следният формат:

[цифри].[цифри] [E|e [+|-] цифри] .

В C++ всички константи с плаваща запетая по дефиниция са от тип **double**. Въпреки това, потребителят има възможност да обяви явно константа с плаваща запетая от тип **float**, като се добави наставка **F** към константата. Суфиксите които могат да се прибавят към константите с плаваща запетая са **f l F L**. За въвеждане на експонента се използва символът **E** или **e**. Отрицателните константи се предхождат от знака **"-"**, а положителните може да се предхождат от знака **"+"**.

Пример:

```
15.75       // = 15.75
+15.75      // = 15.75
-15.75      // = -15.75
1.575E1     // = 15.75
1575E-2     // = 15.75
-2.5E-2     // = -0.0025
25E-4       // = 0.0025
2500F       // = 2500  тип float
2500D       // = 2500  тип double
2500L       // = 2500  тип long
```

Цели константи

Разрешени са константи в интервала от 0 до 4294967295 (десетично). Отрицателните константи са просто константи без знак с оператор унарен минус. Разрешени са осмични, десетични и шестнадесетични константи. Компилаторите за микроконтролери поддържат и двоични константи.

Наставката **L** (или **l**) към константата предизвиква

представянето й от тип **long**. Аналогично наставката **U** (или **u**) показва, че ако константата е от тип **unsigned long**, независимо от използваната бройна система. Позволено е използването на двете наставки за една константа.

Цели константи на C++ (без L или U)

Десетични константи	
0 ÷ 32767	int (short int)
32767 ÷ 2147483647	long
2147483648 ÷ 4294967295	unsigned int
> 4294967295	препълване без предупреждение; получената константа ще има младшите битове на действителната стойност.

Осмични константи		
00	**÷ 077777**	int (short int)
0100000	**÷ 0177777**	unsigned int
01000000	**÷ 017777777777**	long
0100000000000	**÷ 0377777777777**	unsigned long
> 037777777777		препълване, както е обяснено по-горе

Шестнадесетични константи		
0x0000	**÷ 0x7fff**	int
0x8000	**÷ 0xffff**	unsigned int
0x10000	**÷ 0x7fffffff**	long
0x80000000	**÷ 0xffffffff**	unsigned long
>0xFFFFFFFF		препълване,както е обяснено по-горе

Както се вижда по-горе осмичните константи респективно числа се предхождат от **0**, шестнадесетичните от **0x** или **0X**, а двоичните от **0b** или **0B**.

Примери:

Десетични константи	Осмични константи	Шестнадесетични константи	Двоични константи
10	012	0xA	0b000010010
123	0204	0x84	0b100000100
32179	076663	0x7db3 или 0X7DB3	0b0111110110110011 0B0111110110110011

Константи с двойна дължина (long)

Десетични константи	Осмични константи	Шестнадесетични константи
10L	012L	0xAL
123L	0204L	0x84L
32179L	076663L	0x7db3L или 0X7DB3L

Десетични константа без знак:45463U

Десетични константа без знак- unsigned long: 45463UL, 45463LU

Символни константи

Символни константи представляват единични символи затворени в апострофи (единични кавички), например - 'A', 'я', '1'. Повечето ASCII символи могат да се представят като символни константи. Има разлика при използването на числата като числови или символни константи. Числовите константи имат стойността на числото, с което е записана, а символната константа има стойност, равна на кода на това число, взето като символ в ASCII таблицата. Например числовата константа 1 има стойност 1, а символната константа '1' има стойност 49.

Новите версии на C++ поддържат двусимволни константи (Unicode).

Например `'An'`, `'\n\t'` и `'\007\007'`. Те се представят чрез 16 битова цяла стойност (**short int**), с първи символ в младшия байт и втория - в старшия байт. Забележете, че тези константи не осигуряват преносимост на програмата.

Едносимволните константи, например `'A'`, `'\t'` и `'\007'`, също се представят в 16 битова цяла стойност. В случая младшият байт е знаково разширен в старшия по следния начин: ако стойността е над 127 (десетично), старшият байт става -1 (това е 0xFF). Този механизъм може да се отмени като се декларира типът **char** като **unsigned**. Така старшият байт винаги е 0, независимо от стойността на младшия.

C++ позволява шестнадесетично представяне на кодове на символи, например `'\x1F'`, `'\x82'`. Поставя се `"X"` или `"x"` и една до три цифри. C++ поддържа и списък на позволените ESC-поредици. ESC-поредиците са стойности, вмъкнати в константи от тип символ и символен низ, предшествани от обратна наклонена черта `(\)`. В следващата таблица са показани всички позволени поредици.

ESC-поредици използвани в C++

Поредица	Стойност	Символ	Действие
\a	0x07	BEL	Звуков сигнал (Bell)
\b	0x08	BS	Позиция назад (BackSpace)
\f	0x0C	FF	Нова страница (FormFeed)
\n	0x0A	LF	Нов ред (LineFeed)
\r	0x0D	CR	Връщане в началото на реда (Carriage Return)
\t	0x09	HT	Хоризонтален табулатор (Horizontal Tab)
\v	0x0B	VT	Вертикален табулатор (Vertical Tab)
\\	0x5C	\	Обратна наклонена черта
\'	0x2C	'	Апостроф
\"	0x22	"	Кавички
\?	0x3F	?	Въпросителен знак
\DDD	всеки	DDD -	1 до 3 осмични цифри
\xHHH 0xHHH	всеки	HHH	1 до 3 шестнадесетични цифри

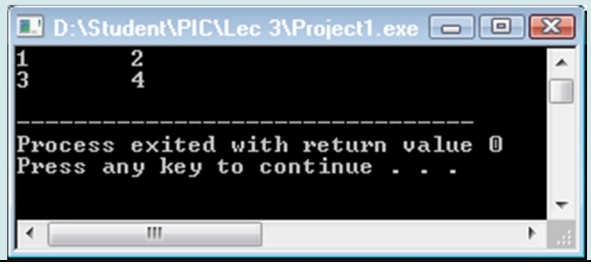
Забележка: Има опасност от двусмислие, ако осмична поредица съдържаща по-малко от три цифри е последвана от цифра (която е позволена за този тип числа), тъй като C++ позволява двусимволни константи. В такива случаи, C++ може да интерпретира следващия символ като част от осмичната поредица.

Пример:

Програмен код

```
#include <iostream>
using namespace std;
int main(int argc, char** argv) {
  cout << "1\t2\n3\t4\n";
  return 0;
}
```

Резултат



Символни низове

Съгласно K&R, константа символен низ е една символна поредица, която се състои от двойни кавички, текст и отново двойни кавички **("текст")**. За да се удължи символна константа на няколко реда трябва да се използва обратна наклонена черта **"\"**.

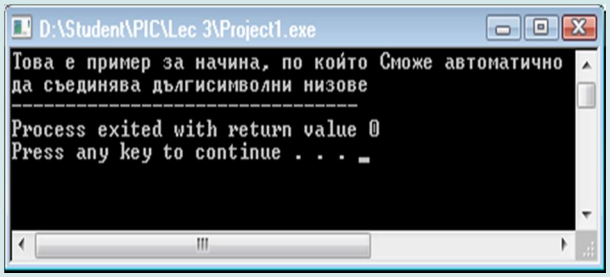
```
"Ето пример за начина, по който C++ \
 разделя дълги символни низове" ;
```

C++ позволява в символна константа да се използват няколко низа, като след това компилаторът ще ги конкатенира. Ето един пример:

Програмен код

```
#include <iostream.h>
using namespace std;
void main() {
  char *p;
  setlocale(LC_ALL,"Bulgarian");
  p = "Това е пример за начина,"
   "по който C може автоматично"
      "\nда съединява дълги "
      "символни низове";
  cout << p;
}
```

Резултат



Резултатът от работата на програмата показва действието с дълги символни низове. Трябва да добавим, че низовата константа, както и низовете в C++ завършват със символа **'\0'** (**ASCII код = 0**).