Паламарчук Андрей ИУ7-23Б

# **Представление в памяти строк и массивов строк**

## **Представление строки в языке Си**

Программа:

#include <stdio.h>  
  
int main(void)  
{  
    char str[] = "I like to study at BMSTU";  
    return 0;  
}

Дамп памяти содержащий эту строку:

$ gcc -std=c99 -g3 str.c

$ gdb ./a.out  
GNU gdb (GDB) Fedora Linux 13.1-3.fc38

...  
Reading symbols from ./a.out...  
(gdb) break 6  
Breakpoint 1 at 0x401142: file str.c, line 6.  
(gdb) run  
Starting program: /home/Natalia/practic\_PTP/Task\_3.3/a.out  
Breakpoint 1, main () at str.c:6  
6           return 0;

(gdb) x /25tb str  
0x7fffffffdc50: 01001001        00100000        01101100 01101001        01101011        01100101        00100000  
01110100  
0x7fffffffdc58: 01101111        00100000        01110011  
01110100        01110101        01100100        01111001  
00100000  
0x7fffffffdc60: 01100001        01110100        00100000  
01000010        01001101        01010011        01010100  
01010101  
0x7fffffffdc68: 00000000

Первые 24 байта содержат последовательность символов нашей строки, кроме того 25-й байт хранит нулевой символ "\0", который обозначает конец строки.

## **Хранение строк в языке Си в виде двумерного массива**

Программа:

#include <stdio.h>  
#define Nmax 7  
  
  
int main(void)  
{  
    char arr[][Nmax + 1] = {"Jeep", "Volvo", "Bentley", "Toyota", "BMW"};  
    return 0;  
}

Дамп памяти:

$ gcc -std=c99 -g3 double\_arr\_str.c  
$ gdb ./a.out  
GNU gdb (GDB) Fedora Linux 13.1-3.fc38

...  
Reading symbols from ./a.out...  
(gdb) break 8  
Breakpoint 1 at 0x401144: file double\_arr\_str.c, line 8.  
(gdb) run  
Starting program: /home/Natalia/practic\_PTP/Task\_3.3/a.out  
Breakpoint 1, main () at double\_arr\_str.c:8  
8           return 0;

(gdb) x /40tb arr  
0x7fffffffdc40: 01001010        01100101        01100101  
01110000        00000000        00000000        00000000  
00000000  
0x7fffffffdc48: 01010110        01101111        01101100  
01110110        01101111        00000000        00000000  
00000000  
0x7fffffffdc50: 01000010        01100101        01101110  
01110100        01101100        01100101        01111001  
00000000  
0x7fffffffdc58: 01010100        01101111        01111001  
01101111        01110100        01100001        00000000  
00000000  
0x7fffffffdc60: 01000010        01001101        01010111  
00000000        00000000        00000000        00000000  
00000000

Наш массив хранит в себе массивы фиксированной длины, каждый из которых содержит в себе последовательность символов одной из наших строк и символ окончания строки. У массивов со строками должна быть длина Nmax + 1, где Nmax – максимальная длина строки, которую необходимо хранить. А также дополнительный байт для хранения нулевого символа "\0".

* Общий объем (объем двумерного массива):

5 \* (7 + 1) = **40 байт**,

где 5 – количество строк, а 7 + 1 – длина каждой строки.

* Объем «полезных» данных:

(4 + 1) + (5 + 1) + (7 + 1) + (6 + 1) + (3 + 1) = **30 байт**,

где слагаемые – это длины наших строк и символы их окончания.

* Объем «вспомогательных» данных:

5 \* (7 + 1) - ((4 + 1) + (5 + 1) + (7 + 1) + (6 + 1) + (3 + 1)) = **10 байт**,

где 5 \* (7 + 1) – общий объем, а (4 + 1) + (5 + 1) + (7 + 1) + (6 + 1) + (3 + 1) – это длины наших строк и символы их окончания.

## **Хранение строк в языке Си в виде массива указателей**

Программа:

#include <stdio.h>  
  
int main(void)  
{  
    char \*arr[] = {"Jeep", "Volvo", "Bentley", "Toyota", "BMW"};  
    return 0;  
}

Дамп памяти:

$ gcc -std=c99 -g3 arr\_pointer\_str.c  
$ gdb ./a.out  
GNU gdb (GDB) Fedora Linux 13.1-3.fc38

...  
Reading symbols from ./a.out...  
(gdb) break 6  
Breakpoint 1 at 0x401132: file arr\_pointer\_str.c, line 6.  
(gdb) run  
Starting program: /home/Natalia/practic\_PTP/Task\_3.3/a.out  
Breakpoint 1, main () at arr\_pointer\_str.c:6  
6           return 0;

(gdb) print sizeof(arr[0])  
$1 = 8  
  
(gdb) x /40tb arr  
0x7fffffffdc40: 00010000        00100000        01000000  
00000000        00000000        00000000        00000000  
00000000  
0x7fffffffdc48: 00010101        00100000        01000000  
00000000        00000000        00000000        00000000  
00000000  
0x7fffffffdc50: 00011011        00100000        01000000  
00000000        00000000        00000000        00000000  
00000000  
0x7fffffffdc58: 00100011        00100000        01000000  
00000000        00000000        00000000        00000000  
00000000  
0x7fffffffdc60: 00101010        00100000        01000000  
00000000        00000000        00000000        00000000  
00000000  
  
(gdb) x /30tb \*arr  
0x402010:       01001010        01100101        01100101  
01110000        00000000        01010110        01101111  
01101100  
0x402018:       01110110        01101111        00000000  
01000010        01100101        01101110        01110100  
01101100  
0x402020:       01100101        01111001        00000000  
01010100        01101111        01111001        01101111  
01110100  
0x402028:       01100001        00000000        01000010  
01001101        01010111        00000000

Наш массив хранит в себе указатели на массивы, каждый массив содержит в себе последовательность символов одной из наших строк и символ окончания строки "\0".

* Общий объем:

5 \* 8 + (4 + 1) + (5 + 1) + (7 + 1) + (6 + 1) + (3 + 1) = **70 байт**,

где 5 – количество строк, а 8 – размер указателя (sizeof(arr[0]) = 8), а остальные слагаемые – это длины наших строк и символы их окончания.

* Объем «полезных» данных:

(4 + 1) + (5 + 1) + (7 + 1) + (6 + 1) + (3 + 1) = **30 байт**,

где слагаемые – это длины наших строк и символы их окончания.

* Объем «вспомогательных» данных:

5 \* 8 = **40 байт**, где 5 – количество строк(количество массивов, каждый из которых требует указатель), а 8 – размер указателя.