**Смирнов Иван ИУ7-22Б - 2023г.**

**Отчет**

**Задание №3.3**

**Отладка**

*Целью* работы является изучение представления в памяти строки языка Си, дампа памяти, который содержит строку полностью.

В программе задан двумерный массив строк длиной 10 (учитывая символ в кодом ‘\0’).

Первая программа:

(gdb) list

#include <stdio.h>

#define OK 0

#define N 10

int main(void)

{

char string[][N] = {"First", "Second", "Third", "Fourth", "Fifth"};

(gdb) list

return OK;

}

Вторая программа:

(gdb) list

#include <stdio.h>

#define OK 0

#define N 10

int main(void)

{

/\*const\*/ char \*string[] = {"First", "Second", "Third", "Fourth", "Fifth"};

(gdb) list

return OK;

}

С помощью команды отладчика gdb “x /nfu”, узнаем дамп памяти данного массива строк. Но для начала узнаем размер массива в байтах.

Первая программа:

(gdb) print sizeof(string)

$1 = 50

(gdb) x /50xb string

0x7fffffffdef0: 0x46 0x69 0x72 0x73 0x74 0x00 0x00 0x00

0x7fffffffdef8: 0x00 0x00 0x53 0x65 0x63 0x6f 0x6e 0x64

0x7fffffffdf00: 0x00 0x00 0x00 0x00 0x54 0x68 0x69 0x72

0x7fffffffdf08: 0x64 0x00 0x00 0x00 0x00 0x00 0x46 0x6f

0x7fffffffdf10: 0x75 0x72 0x74 0x68 0x00 0x00 0x00 0x00

0x7fffffffdf18: 0x46 0x69 0x66 0x74 0x68 0x00 0x00 0x00

0x7fffffffdf20: 0x00 0x00

Вторая программа:

(gdb) print sizeof(string)

$30 = 40

(gdb) x /40xb \*string

0x555555556004: 0x46 0x69 0x72 0x73 0x74 0x00 0x53 0x65

0x55555555600c: 0x63 0x6f 0x6e 0x64 0x00 0x54 0x68 0x69

0x555555556014: 0x72 0x64 0x00 0x46 0x6f 0x75 0x72 0x74

0x55555555601c: 0x68 0x00 0x46 0x69 0x66 0x74 0x68 0x00

0x555555556024: 0x01 0x1b 0x03 0x3b 0x30 0x00 0x00 0x00

В дампе памяти видно, что строки (массив из char) представлены в виде последовательности 10 байтов (причем если строка имеет длину меньше 10, то оставшиеся байты заполняются символами ‘\0’, в дампе они выглядят ка 0x00). Переменная типа char рассчитана на хранение только одного символа (например, буквы или пробела). В памяти компьютера символы хранятся в виде целых чисел (-128 до 127). Соответствие между символами и их кодами определяется таблицей кодировки, которая зависит от компьютера и операционной системы.

(gdb) print \*\*string

$3 = 70 'F'

Из дампа памяти было видно, что значение байта для прописной буквы F было равно 0x46, что в 10-ой системе как раз равно 70, а 70 – код данного символа в таблице ASCII.

Суммарный расчет размера занятой памяти для первой программы:

(gdb) print sizeof(string)

$4 = 50

(gdb) print sizeof(\*string)

$5 = 10

(gdb) print sizeof(\*\*string)

$6 = 1

”Вспомогательные” данные – символы с кодом “\0”. Расчитаем объем данных.

(gdb) print (\*(\*string+5))

$6 = 0 '\000'

(gdb) print (\*(\*string+6))

$7 = 0 '\000'

(gdb) print sizeof(\*(\*string+5))

$8 = 1

(gdb) print sizeof(\*(\*string+6))

$9 = 1

“Вспомогательные” данные занимают 23 байта.  
Значит “полезные” данные – (50-23)=27 байт.

Аналогичный расчет выполним для второй программы:

(gdb) print sizeof(string)

$30 = 40

(gdb) print sizeof(\*string)

$31 = 8

(gdb) print sizeof(\*\*string)

$32 = 1

(gdb) print \*(\*string+5)

$42 = 0 '\000'

(gdb) print \*(\*string+6)

$43 = 83 'S’

(gdb) print sizeof(\*(\*string+5))

$44 = 1

”Вспомогательные” данные занимают 5 байт.

“Полезные” данные занимают – (40-5)=35 байт.