Задание №4 в рамках вычислительного практикума. Представление в памяти многомерного статического массива

1.Описание трехмерного массива

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #define BLOCK\_SIZE 2  #define ROW\_SIZE 3  #define COLUMN\_SIZE 4  int main(void)  {  int array\_3d[BLOCK\_SIZE][ROW\_SIZE][COLUMN\_SIZE] =  { {{10,20,30,40},{40,50,60,70},{70,80,90,100}},//elements of block 1  {{11,22,33,44},{44,55,66,77},{77,88,99,111}} //elements of block 2  };    **return** EXIT\_SUCCESS;  } |

2.Вывод дампа памяти, содержащего массив полностью.

|  |
| --- |
| (gdb) x /24xw array\_3d  0xffffffffeef8: 0x0000000a 0x00000014 0x0000001e 0x00000028  0xffffffffef08: 0x00000028 0x00000032 0x0000003c 0x00000046  0xffffffffef18: 0x00000046 0x00000050 0x0000005a 0x00000064  0xffffffffef28: 0x0000000b 0x00000016 0x00000021 0x0000002c  0xffffffffef38: 0x0000002c 0x00000037 0x00000042 0x0000004d  0xffffffffef48: 0x0000004d 0x00000058 0x00000063 0x0000006f |

3. Вывод компонент массива

|  |
| --- |
| (gdb) x /12xw array\_3d[0]  0xffffffffeef8: 0x0000000a 0x00000014 0x0000001e 0x00000028  0xffffffffef08: 0x00000028 0x00000032 0x0000003c 0x00000046  0xffffffffef18: 0x00000046 0x00000050 0x0000005a 0x00000064  (gdb) x /12xw array\_3d[1]  0xffffffffef28: 0x0000000b 0x00000016 0x00000021 0x0000002c  0xffffffffef38: 0x0000002c 0x00000037 0x00000042 0x0000004d  0xffffffffef48: 0x0000004d 0x00000058 0x00000063 0x0000006f  (gdb) x /4xw array\_3d[0][0]  0xffffffffeef8: 0x0000000a 0x00000014 0x0000001e 0x00000028  (gdb) x /4xw array\_3d[0][1]  0xffffffffef08: 0x00000028 0x00000032 0x0000003c 0x00000046  (gdb) x /4xw array\_3d[0][2]  0xffffffffef18: 0x00000046 0x00000050 0x0000005a 0x00000064  (gdb) x /4xw array\_3d[1][0]  0xffffffffef28: 0x0000000b 0x00000016 0x00000021 0x0000002c  (gdb) x /4xw array\_3d[1][1]  0xffffffffef38: 0x0000002c 0x00000037 0x00000042 0x0000004d  (gdb) x /4xw array\_3d[1][2]  0xffffffffef48: 0x0000004d 0x00000058 0x00000063 0x0000006f  (gdb) x /xw &array\_3d[0][0][0]  0xffffffffeef8: 0x0000000a  (gdb) x /xw &array\_3d[0][0][1]  0xffffffffeefc: 0x00000014  (gdb) x /xw &array\_3d[0][0][2]  0xffffffffef00: 0x0000001e  (gdb) x /xw &array\_3d[0][0][3]  0xffffffffef04: 0x00000028  (gdb) x /xw &array\_3d[0][1][0]  0xffffffffef08: 0x00000028  ///////////////  (gdb) x /xw &array\_3d[1][2][3]  0xffffffffef54: 0x0000006f |

4. Указатели для работы с компонентами.

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #define BLOCK\_SIZE 2  #define ROW\_SIZE 3  #define COLUMN\_SIZE 4  #define START 0  int main(void)  {  int array\_3d[BLOCK\_SIZE][ROW\_SIZE][COLUMN\_SIZE] =  { {{10,20,30,40},  {40,50,60,70},  {70,80,90,100}}, //elements of block 1  {{11,22,33,44},  {44,55,66,77},  {77,88,99,111}} //elements of block 2  };  int (\*a)[ROW\_SIZE][COLUMN\_SIZE] = array\_3d;  int (\*b)[COLUMN\_SIZE] = array\_3d[START];  int (\*c) = array\_3d[START][START];  int \*s = &array\_3d[START][START][START];  return EXIT\_SUCCESS;  } |

Размер указателя равен 8 байт .

Выведем через gdb размер указателей и компонент:

|  |
| --- |
| (gdb) p sizeof(c)  $1 = 8  (gdb) p sizeof(b)  $2 = 8  (gdb) p sizeof(a)  $3 = 8  (gdb) p sizeof(s)  $4 = 8  (gdb) p sizeof(\*c)  $5 = 4  (gdb) p sizeof(\*b)  $6 = 16  (gdb) p sizeof(\*a)  $7 = 48  (gdb) p sizeof(\*s)  $8 = 4 |

Указатель \*a весит 48 байт, т.к ссылается на array\_3d[3][4], где 12 элементов. Также array\_3d[0][0] занимает 16 байт.

Указатели \*b и \*c ссылаются на элемент массива типа “int”, поэтому их размер \*b составляет 16 байт, а \*c составляет 4 байта.

Массив имеет тип “int”, поэтому элемент массива весит 4 байта.

5. Заголовок функции для обработки компонент соответствующего уровня.

|  |
| --- |
| void level0(int array\_3d[][ROW\_SIZE][COLUMN\_SIZE], size\_t count\_row, size\_t count\_cols); |

|  |
| --- |
| void level1(int array\_3d[][COLUMN\_SIZE], size\_t count\_row, size\_t count\_cols); |

|  |
| --- |
| void level2(int array\_3d[COLUMN\_SIZE], size\_t count\_cols); |

|  |
| --- |
| void level3(int s); |