Практикум на ЭВМ
Динамические многомерные массивы.
Динамические структуры.

Баев А.Ж.

Казахстанский филиал МГУ

06 октября 2018

Статические матрицы

Массив `int a[2]` — указатель типа `int *`.
Матрица типа `int [2][3][4]` — указатель типа `int (*)[2][3]`.
Почему именно первая размерность?

Следующие прототипы одинаковы:

```
void f(int array[2][3][4]);
void f(int (*array)[3][4]);
```

Статические матрицы

Обратите внимание на круглые скобки.

```
int (*array)[3][4]
```

```
int *array[3][4]
```

Статические матрицы — плотно упакованные

```
void *memset(void *s, int c, size_t n);
void *memcpy(void *dest, void *src, size_t n);
```

Инициализация всей матрицы.

```
#include <string.h>

const int n = 10, m = 20;

int A[n][m];
memset(A, 0, n * m * sizeof(int));

int B[n][m];
memcpy(B, A, n * m * sizeof(int));

int C[n * m];
memcpy(C, A, n * m * sizeof(int));
```

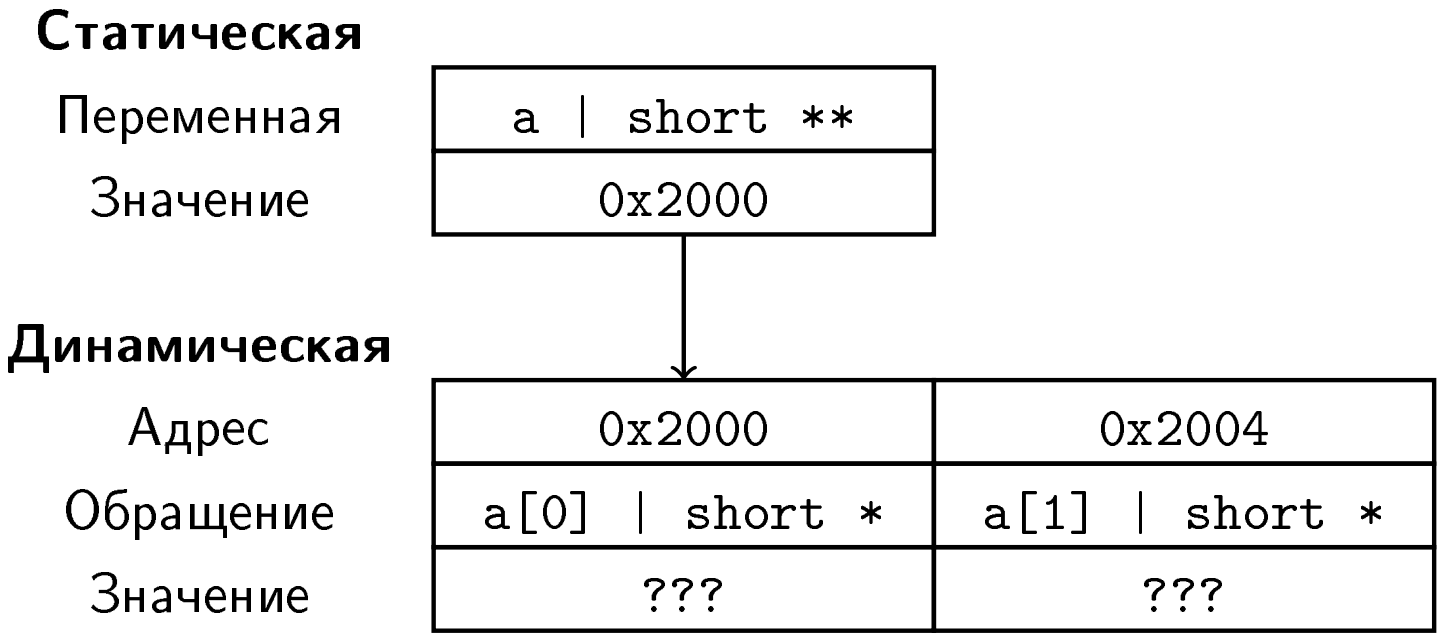
Динамические матрицы

1) уложить всю матрицу построчно в виде одного динамического массива (плотная упаковка);
2) создать дополнительный массив указателей на динамические массивы (неплотная упаковка).
Смотрим второе. Почему?

Динамические матрицы. Матрица 2x3

Каждая строка матрицы — динамический массив. Где хранить все эти указатели? Еще в одном динамическом массиве!

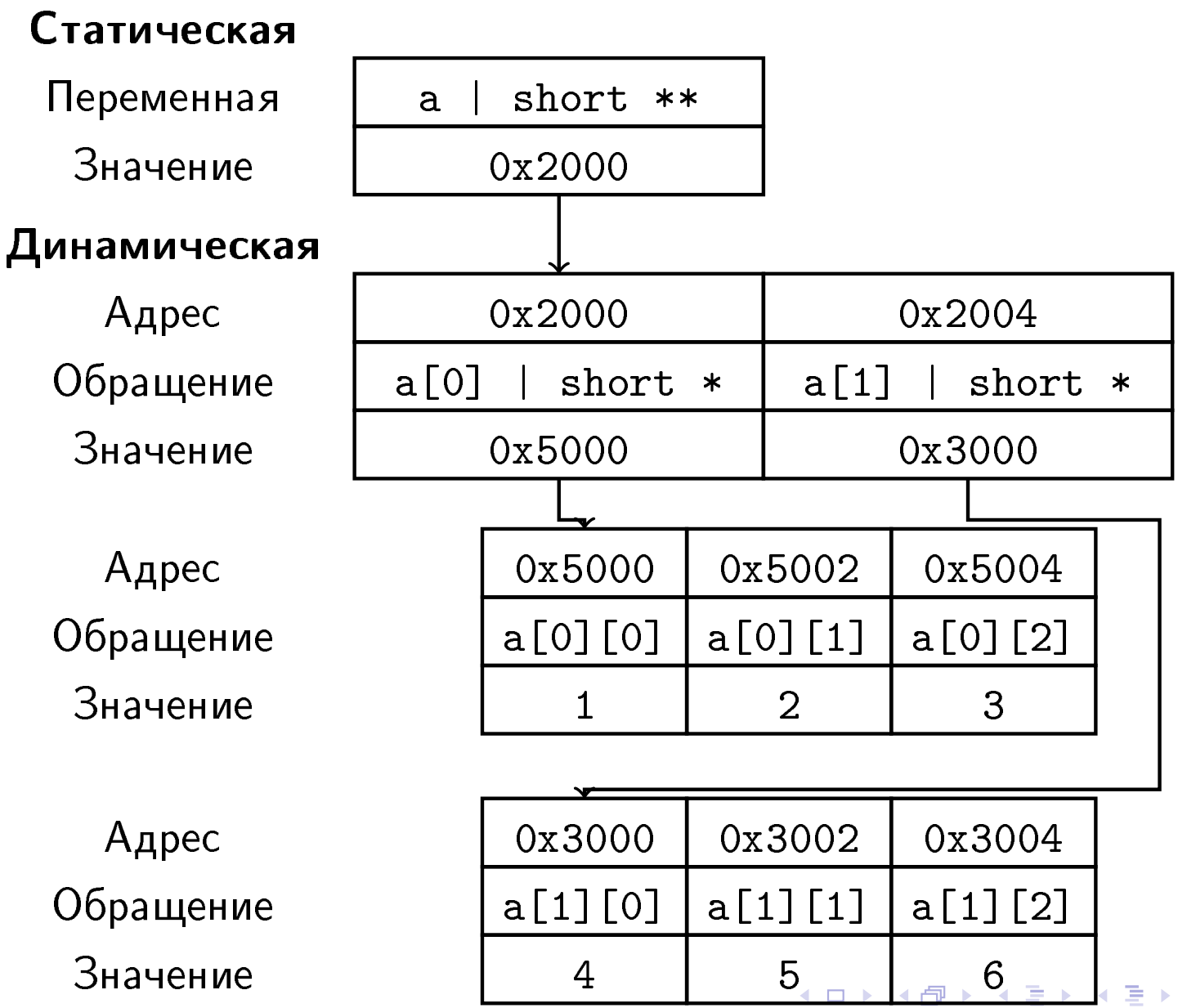
```
int **a = malloc(2 * sizeof(int *));
```



Динамические матрицы. Матрица 2x3

```
    a[0] = malloc(3 * sizeof(int));
    a[1] = malloc(3 * sizeof(int));
```

Динамические матрицы. Матрица 2x3



Динамические матрицы. Матрица 2x3

Очистка

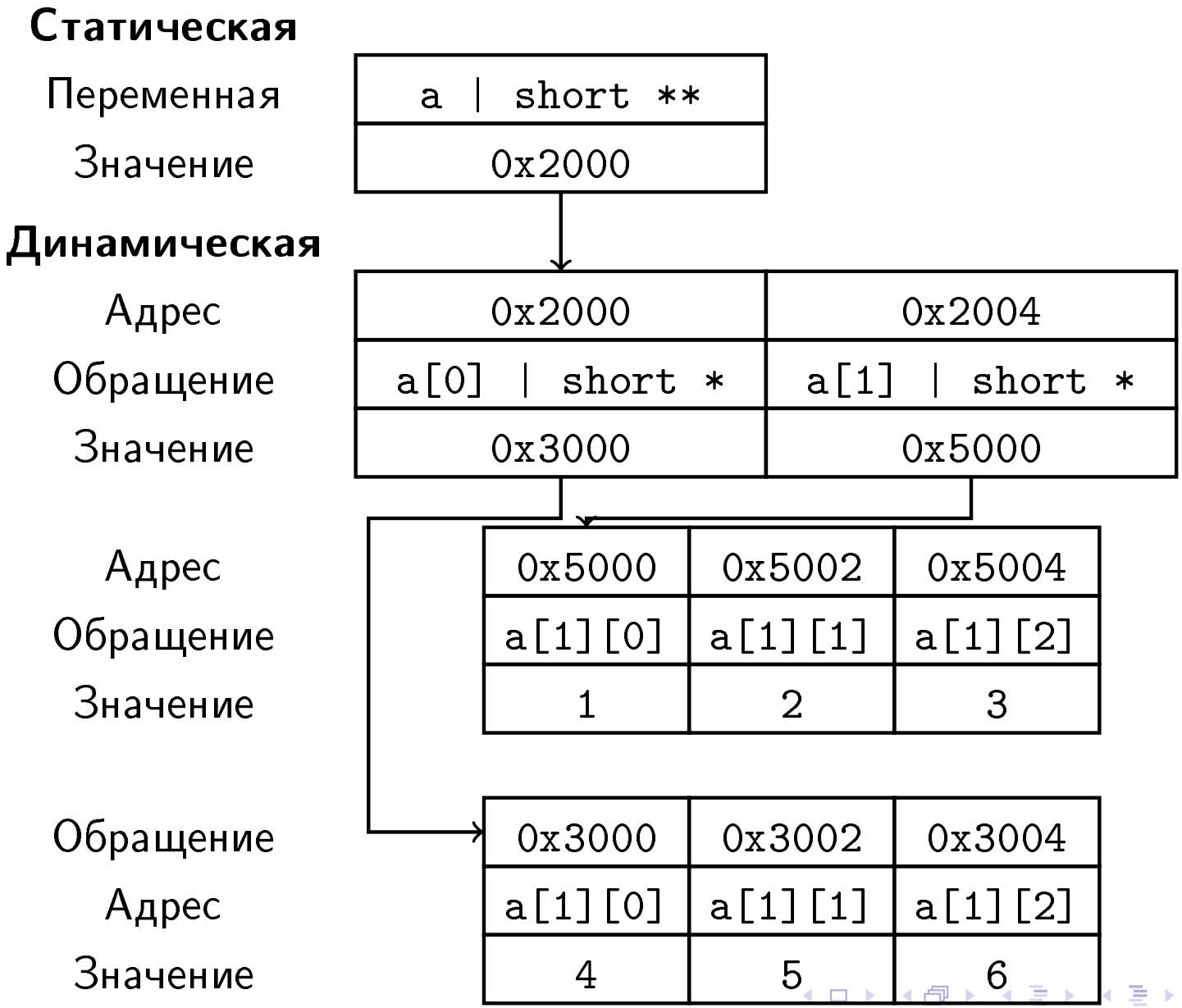
```
free(a[0]);
free(a[1]);
free(a);
```

Динамические матрицы. Любимая плюшка линала

Перестановка строк!

```
short *tmp = a[0];
a[0] = a[1];
a[1] = tmp;
```

Динамические матрицы. Матрица 2x3



Отличия динамической и статической матрицы.

1. матрица размера $n \times m$, где каждый элемент имеет размер s, а указатель — размер p. В случае со статической памятью: $n \cdot m \cdot s$. В случае с динамической памятью: $(1 + n) \cdot p$ для указателей и $n \cdot m \cdot s$ для самой матрицы.
2. динамические матрицы хранятся кусками по строкам, а статические — цельном блоком. Перестановка строк O(1).
3. Размеры разных строк могут быть разными.

Аргументы командной строки.

```
$ gcc prog.c -o prog -Wall
```

Здесь 4 аргумента:

```
int main(int argc, char** argv);
```

При запуске вашей программы, параметр argc будет содержать количество параметров командной строки, включая название программы (5).
Аргумент argv — это массив строк, которые сформированы из аргументов, включая название программы:

```
argv[0] == "gcc"
argv[1] == "prog.c"
argv[2] == "-o"
argv[3] == "prog"
argv[4] == "Wall"
```

Головоломки.

0. Какие присваивания допустимы между указанными переменными (среди всех возможных пар)?

```
int a[2][3];
int (*b)[3];
int *c[3];
int **d;
```

1. Динамическая матрица из 3×4. Строки и столбы — динамические.
2. Динамическая матрица из 3×4. Строки — статические, столбы — динамические.
3. Динамическая матрица из 3×4. Строки — динамические, столбы — статические.
4. Написать программу, которая печатает свой исходный код, при условии, что он лежит рядом и имеет такое же название (prog и prog.c).
5. Написать программу, которая печатает свой бинарный код.