Семинар 5: МЕМы

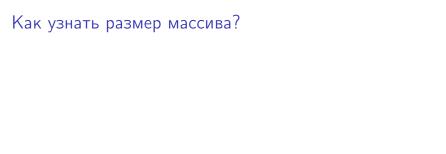
3 Декабря, 2019

Массивы

- Непрерывные куски памяти, элементы располагаются подряд
- Нумерация с нуля
- Массивы размера 0 существуют
- ▶ Обращение к элементам за пределами массива undefined behavior!

Многомерные массивы

- ▶ Многомерные массивы тоже укладываются непрерывно в памяти
- ightharpoonup Одномерные: pos = i
- ightharpoonup Двумерные: $pos = i \times N + j$
- ightharpoonup Трёхмерные: $pos = i \times N \times M + j \times N + m$
- ► К-мерные: $pos = \sum_{i=0}^{K} idx_i \times k_i$



Как узнать размер массива?

```
sizeof (arr) / sizeof (arr [0])
```

Указатели

- Указатель номер байта в адресном пространстве
- ▶ Любой указатель имеет тип (например, int *, long *, ...)
- ▶ Разыменовывание (dereference) обращение к памяти, на которую указывает указатель
- ▶ Разыменовывание NULL-указателя undefined behavior!
- ▶ void* ссылается на что угодно, нельзя разыменовать
- ▶ Приведение указателей (casting): void* v = ...; int* a = v;

Арифметика указателей

- ► Сдвиг вправо на N элементов: ptr + N
- ▶ Сдвиг влево на N элементов: ptr − N
- ▶ Количество элементов между двумя указателями: $ptr_a ptr_b$

Константность указателей

- const char * a;
- ► char * const a;
- const char * const a;

Константность указателей

- ► const char * a; неизменяемые данные
- ▶ char * const a; неизменяемый указатель
- ► const char * const a; «броня не пробита»

Строки в С

ightharpoonup Строки — непрерывная последовательность байт, оканчивающаяся нулевым байтом (\0)

Полезные функции

- size_t strlen (const char* s)
- char* strcpy(char* dest, const char* src)
- ▶ int strcmp(const char* s1, const char* s2)
- char* strstr (const char* haystack, const char* needle)
- char* strchr (const char* s, int c)

Хитрый вопрос...

```
char a [] = "Hi, HSE!";
strlen (a) == ?
```

Хитрый вопрос...

```
char a [] = "Hi, HSE!";
strlen (a) == ?
sizeof (a) == ?
```

...и простой ответ

```
char a [] = "Hi, HSE!";
strlen (a) == 8
sizeof (a) == 9
```

Полезные функции

- void* memset(void* s, int c, size_t n)
- ▶ int memcmp(const void* s1, const void* s2, size t n)
- void* memcpy(void* dest, const void* src, size t num)
- void* memmove(void* dest, const void* src, size_t num)

Полезные функции

- ▶ Старайтесь не использовать str* функции напрямую!
- Если приходится использовать, используйте аналоги с «n» в названии!!
- ▶ Указать длину строчки лучше, чем её не указать!!!

► memset в некоторых случая может быть удалён компилятором
▶ Чтобы такого не происходило, есть memset_s

- memset в некоторых случая может быть удалён компилятором
 - ► Чтобы такого не происходило, есть memset s

▶ А зачем?

memset в некоторых случая может быть удалён компилятором
 Чтобы такого не происходило, есть memset s

А зачем? Криптография!

Динамическое выделение памяти

- void* malloc(size_t size)
- void free (void* ptr)
- void* realloc (void* ptr, size_t size)

Ошибки при работе с динамической памятью

- memory leaks
- ▶ double free
- use-after-free

Understanding malloc



String workshop

Merci!