Семинар №4
ФАКИ 2015

Бирюков В. А.

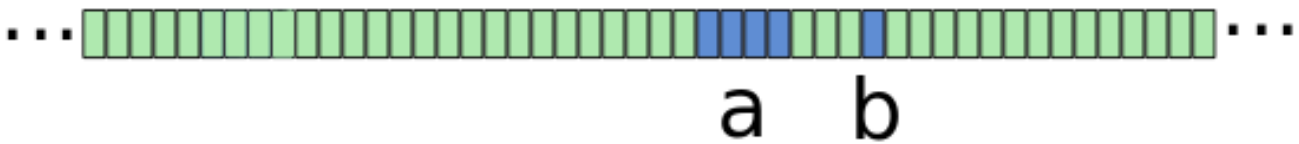
September 23, 2016

Память

Указатели

Указатели в памяти, объявление указателей

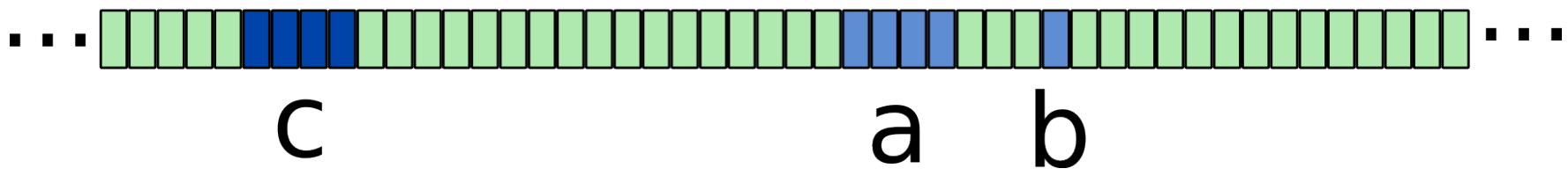
```
int a;
char b;
```



Указатели

Указатели в памяти, объявление указателей

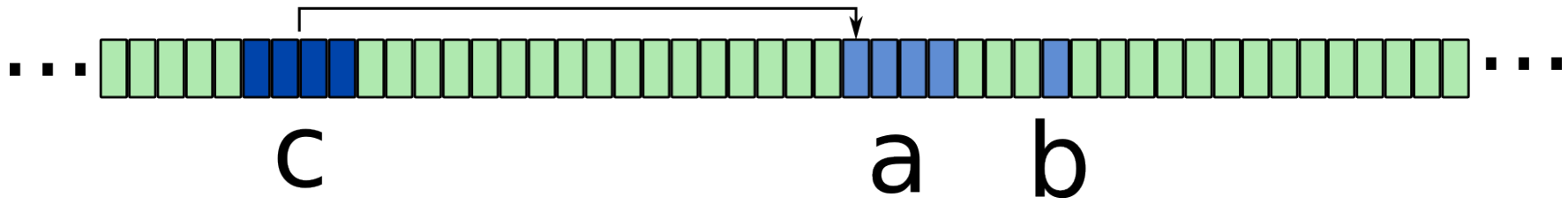
```
int a;
char b;
int * c;
```



Указатели

Адрес переменной

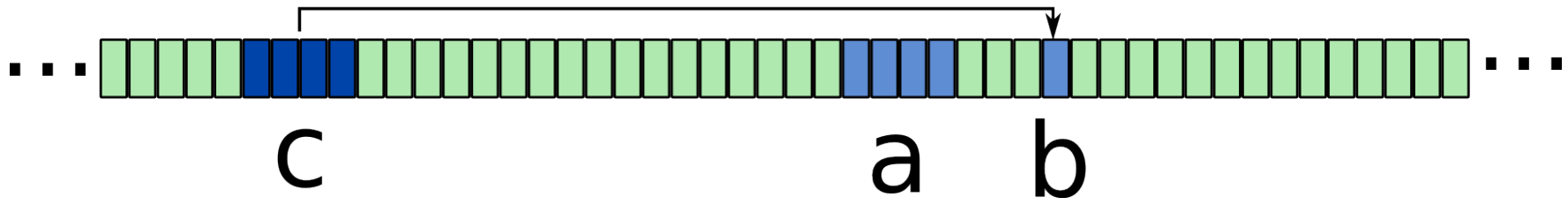
```
int a;
char b;
int * c = &a;
```



Указатели

Адрес переменной

```
int a;
char b;
char * c = &b;
```



Массивы и строки

Массивы

- Массив – набор элементов, расположенных в памяти непосредственно друг за другом, доступ к которым осуществляется по индексу
- Объявление:

```
type arrayName [ arraySize ];
```

- Доступ к элементу (Нумерация в массиве начинается с 0):

```
arrayName [ index ];
```

Массивы

Примеры

Объявление:

```
int array[10];
float average_temperature[12];
```

Доступ к элементу:
Нумерация в массиве начинается с 0

```
printf("%d\n", array[9]);
average_temperature[2] = 5.2;
```

Массивы

Инициализация

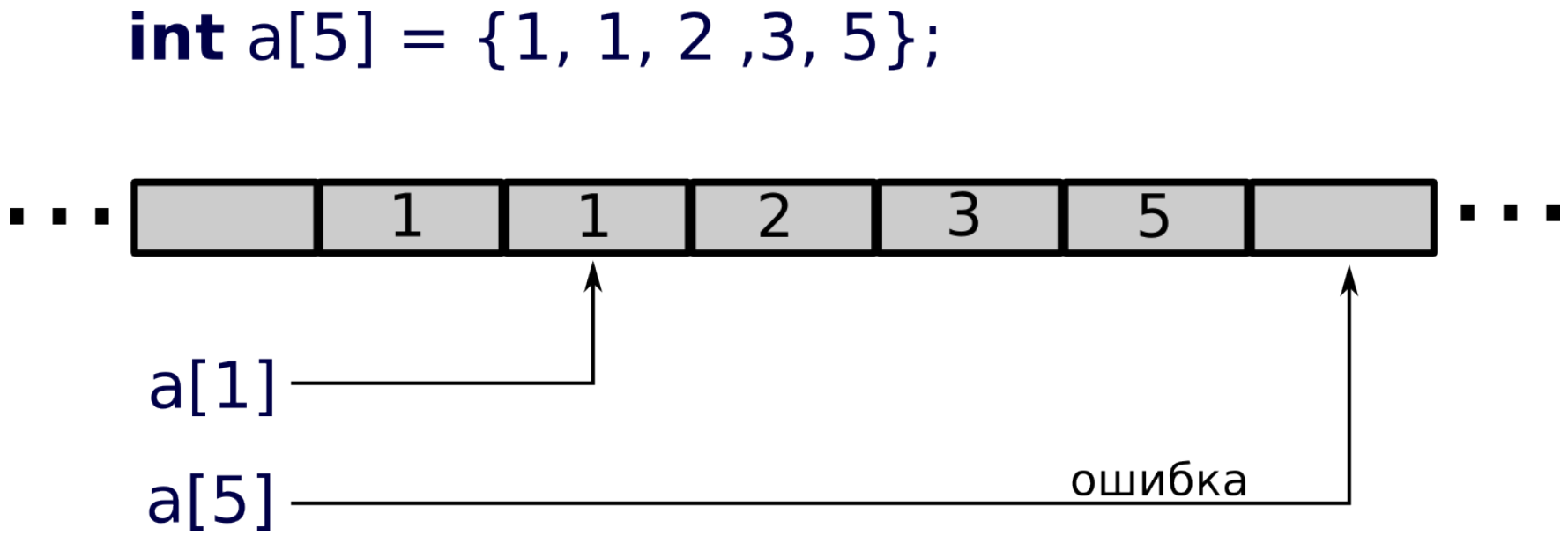
```
int array[10];
for (int i = 0; i < 10; ++i) {
    array[i] = /* something */;
}
```

Или так:

```
int array[5] = {1, 1, 2, 3, 5};
```

Массивы

Массивы в памяти



Примеры кода

Пример 1

Что делает эта программа?

```c
#include <stdio.h>
int main()
{
    int A[10];
    for (int i = 0; i < 10; ++i){
        scanf("%d", &A[i]);
    }
    for (int i = 9; i >= 0; --i) {
        printf("%d ", A[i]);
    }
    printf("\n");
    return 0;
}
```

Строки

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Строки

- Строки в языке C – на самом деле массивы из элементов типа char
- Существенное отличие – в конце строки должен стоять символ '\0'
- Объявление:

```
char string[10];
```

- Доступ к элементу (Нумерация в строке тоже начинается с 0):

```
printf("%c\n", string[0]);
```

Строки

Инициализация

```
int string[10];
for (int i = 0; i < 10; ++i) {
    string[i] = /* something */;
}
```

Или так:

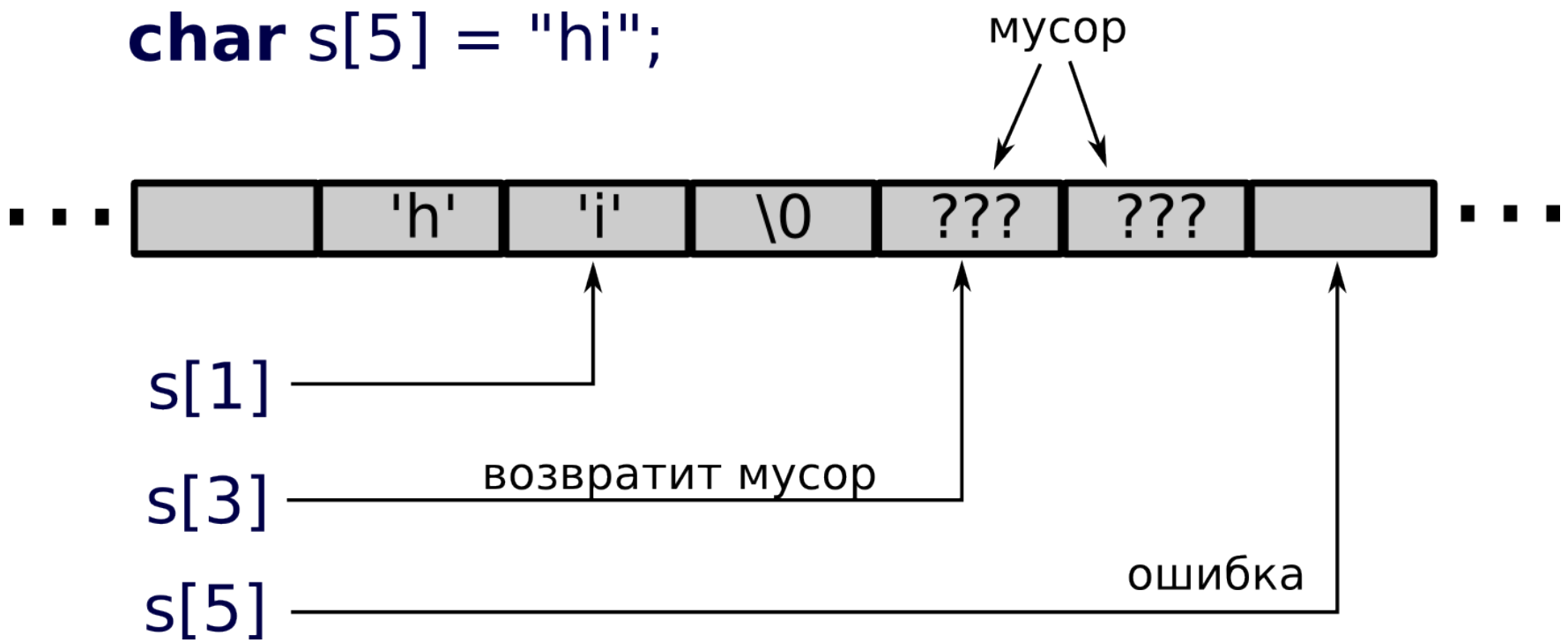
```
int string[10] = {'h', 'e', 'l', 'l',
    'o', '\0', 'a', 'b', 'c', 'd'};
```

Или так:

```
int string[10] = "hello";
```

Строки

Строки в памяти



Функции для работы со строками

Чтение строки:

```
char string[10];
scanf("%10s", string);
```

Длина строки:

```
#include <string.h>
char string[10] = "hello";
int n = strlen(string);
```

Функции для работы со строками

```
#include <string.h>
char s1[10] = "hi";
char s2[10] = "world";
```

Копирование строки s2 в строку s1:

```
strcpy(s1, s2);
```

Конкатенация строки s2 в строку s1:

```
strcat(s1, s2);
```

Функции для работы со строками

```
#include <string.h>
char s1[10] = "hi";
char s2[10] = "hello";
```

Сравнение строк s1 и s2:

```
strcmp(s1, s2);
```

Возвращает указатель на первое вхождение строки s2 в строку s1:

```
strstr(s1, s2);
```

Функции для работы со строками

```
#include <string.h>
char s1[10] = "hi";
char s2[10] = "hello";
```

Считывание из строки:

```
int a, b;
sscanf("42 400", "%d%d", &a, &b);
printf("%s %d", str, i);;
```

Примеры кода

Пример 1

Что делает эта программа?

```
#include <stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    for (int i = 0; i < n; ++i){
        int a;
        scanf("%d", &a);
        printf("%d ", a+1);
    }
    printf("\n");
}
```

Пример 2

Что делает эта программа?

```c
#include <stdio.h>
#include <string.h>
int main()
{
    char s[1001];
    scanf("%1001s", s);
    for (int i = 0; i < strlen(s); ++i){
        if (s[i] == 'a')
            s[i] = 'o';
    }
    printf("%s\n", s);
}
```