

Семинар №6

ФАКИ 2017

Бирюков В. А.

October 9, 2017

Пройденные темы

Название типа	Число бит	Макс./мин. значения
char	8	-128..127 или 0..255
short	16	-32768..32767
int	32	$-2 \cdot 10^9 \dots +2 \cdot 10^9$
long	64	$-2^{63} \dots +2^{63} - 1$
long long	64	$-2^{63} \dots +2^{63} - 1$
float	32	$10^{-38} \dots 10^{+38}$
double	64	$10^{-308} \dots 10^{+308}$
Указатель <имя типа>*	64	
например int*, char** ...		

- `if else`
- Циклы
 - `for`
 - `while`
 - `do while`
- `break` и `continue`
- `switch`

Возвращаемый
тип

Имя функции

Параметры
функции

```
int sum ( int a, int b )
```

```
{
```

```
    int c;
```

← Объявление локальной
переменной

```
    c = a + b;
```

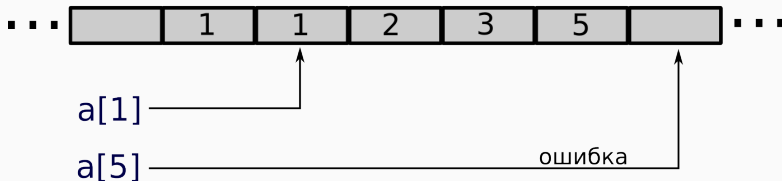
```
    return ( c );
```

```
}
```

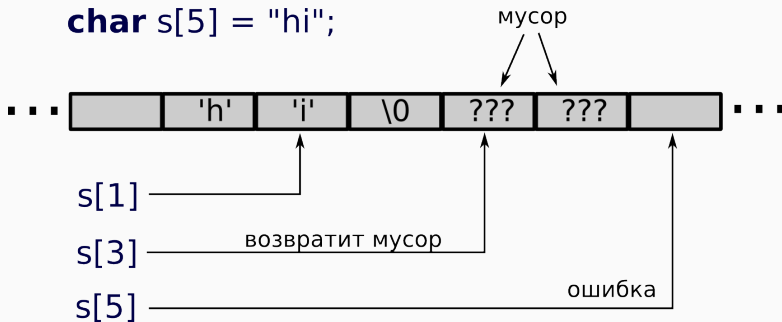
↑
Оператор return

завершает выполнение функции
возвращает значение c

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



```
char s[5] = "hi";
```



Замечания

Для 64-х битных систем:

Название типа	Число бит	Макс. значения
char	8	-128..127 или 0..255
short	16	-32768..32767
int	32	$-2 \cdot 10^9 \dots +2 \cdot 10^9$
long	64	$-2^{63} \dots +2^{63} - 1$
long long	64	$-2^{63} \dots +2^{63} - 1$

Беззнаковые целочисленные типы

Для 64-х битных систем:

Название типа	Число бит	Макс. значения
unsigned char	8	0..255
unsigned short	16	0..65535
unsigned int	32	0 ..+4 · 10 ⁹
unsigned long	64	0 ..+2 ⁶⁴ - 1
unsigned long long	64	0 ..+2 ⁶⁴ - 1

Функции printf() и scanf().

Обозначение	Тип
%d или %i	int
%u	unsigned int
%l	long
%ul	unsigned long
%ll	long long
%ull	unsigned long long
%f	float
%f	double
%c	char
%s	Строка

- Опция -std=c99:

```
for (int i = 0; i < N; i++)
```

```
gcc -std=c99 <имя_файла.c>
```

- Цикл for:

```
for (int i = 0; i * i < 1000; i += 5)
```

```
for (int i = 10; i > 0; i--)
```

- Целочисленное деление: (напечатает 2.0)

```
int a = 20, b = 7;  
printf("%f\n", a / b);
```

- Использование библиотеки <math.h>

Чтобы использовать математические функции `sqrt()`, `log()`, `sin()`, `cos()`, `tan()` и другие нужно подключить библиотеку:

```
#include <math.h>
```

и добавить опцию компилятора `-lm`:

```
gcc -std=c99 -lm <имя_файла.c>
```

- Директива `#define`:

```
#define NUMBER_OF_ELEMENTS 100
```

Заменяет в тексте программы

`NUMBER_OF_ELEMENTS` на 100

- `const`:

```
const int number_of_elements = 100;
```

```
number_of_elements = 200; // Не будет работать
```

Замечания: двумерные массивы

Сумма двумерных массивов

```
int A[100][50], B[100][50], C[100][50];  
// ...  
for (int i = 0; i < 100; i++)  
    for (int j = 0; j < 50; j++)  
    {  
        C[i][j] = A[i][j] + B[i][j];  
    }
```

Передача аргументов в функцию

Передача по значению

```
int min(int a, int b)
{
    if (a < b)
        b = a;
    return b;
}

int main()
{
    int a = 10, b = 40;
    min(a, b);
    printf("%d\n", b);
}
```


Передача аргументов в функцию

Передача по значению

```
int min(int a, int b)
{
    if (a < b)
        b = a;
    return b;
}

int main()
{
    int x = 10, y = 40;
    min(x, y);
    printf("%d\n", y);
}
```

Передача аргументов в функцию

Передача по значению

```
int min(int a, int b)
{
    if (a < b)
        b = a;
    return b;
}

int main()
{
    min(10, 40);
}
```

Передача аргументов в функцию

Передача с помощью указателей

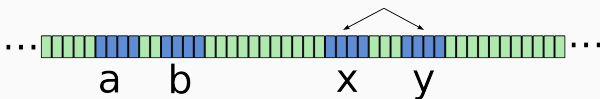
```
void normalize(float* a, float* b)
{
    float sum = *a + *b;
    *a = *a / sum;
    *b = *b / sum;
}

int main()
{
    float x = 10.0, y = 40.0;
    normalize(&x, &y);
    printf("%f\n", y);
}
```

Указатели и аргументы функций

Передача по значению

```
void swap(int x, int y) /* НЕПРАВИЛЬНО! */  
{  
    int temp;  
    temp = x;  
    x = y;  
    y = temp;  
}  
...  
swap(a, b);
```



```
void swap(int * px, int * py)
{
    int temp;
    temp = *px;
    *px = *py;
    *py = temp;
}
...
swap(&a, &b);
```



Передача массивов в функцию

Автоматически передаются с помощью указателей

```
void add_num(int n, int arr[], int x)
{
    for (int i = 0; i < n; ++i)
        arr[i] += x;
}

int main()
{
    int arr[5] = {1, 5, 7, 3, 16};
    add_num(10, arr, 2);
}
```

Передача массивов в функцию

Автоматически передаются с помощью указателей

```
void add_num(int n, int* arr, int x)
{
    for (int i = 0; i < n; ++i)
        arr[i] += x;
}

int main()
{
    int arr[5] = {1, 5, 7, 3, 16};
    add_num(10, arr, 2);
}
```

```
for (int i = 0; i < 10; ++i)
{
    if (i == 6)
        break
    printf("%d ", i);
}
```

Напечатает: 0 1 2 3 4 5


```
for (int i = 0; i < 10; ++i)
{
    if (i == 6)
        continue
    printf("%d ", i);
}
```

Напечатает: 0 1 2 3 4 5 7 8 9

Могли пройти, но не прошли

```
int i = 0;

do
{
    i++;
    printf("%d ", i);
} while (i < 3);
```

Напечатает 1 2 3

```
switch(x) {  
    case 1:  
        printf("It's one!\n");  
        break;  
    case 2:  
        printf("It's two!\n");  
        break;  
    case 3:  
        printf("It's three!\n");  
        break;  
    default:  
        printf("It's something else!\n")  
}
```

Перечисляемый тип enum

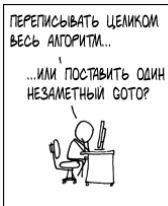
```
enum day{Mon, Tue, Wed, Thur, Fri, Sat, Sun};  
// Mon = 0, Tue = 1, Wed = 2, ...  
int main()  
{  
    enum day x;  
    x = Wed;  
    printf("%d", x);  
}
```

Напечатает 2

- Тернарный оператор: $\langle \text{условие} \rangle ? \langle \#1 \rangle : \langle \#2 \rangle$

```
int b = a > 0 ? a : -a;
```

- Запретный оператор goto



Тренировка перед контрольной работой

<http://style.vdi.mipt.ru/> – тренировка перед контрольной работой