

# Онлайн образование

otus.ru



Проверить, идет ли запись

**Меня хорошо видно  
&& слышно?**



Тема вебинара

# Основные конструкции



**Коробков Виктор**

Консультант команды технологического обеспечения ООО «ИТ ИКС5 Технологии»

Telegram: @Korobkov\_Viktor



# Преподаватель



## Виктор Коробков

более 20 лет в IT

специализация: проектирование баз данных (СУБД PostgreSQL, MS SQLServer)

В OTUS веду занятия на курсах: СУБД, PostgreSQL, SQL Server Developer, noSQL, Программист C

# Правила вебинара



Активно  
участвуем



Off-topic обсуждаем  
в Telegram



Задаем вопрос  
в чат или голосом



Вопросы вижу в чате,  
могу ответить не сразу

## Условные обозначения



Индивидуально



Время, необходимое  
на активность



Пишем в чат



Говорим голосом



Документ



Ответьте себе или  
задайте вопрос

# Маршрут вебинара



Переменные

Выражения и операторы

Управляющие конструкции

Указатели

extern, static, const

Рефлексия

# Цели вебинара

После занятия вы сможете

1. Вспомнить основные выражения и операции
2. Управляющие конструкции языка C
3. Понять, что такое указатели и для чего они нужны



# Переменные

тип            идентификатор1 [, идентификатор2, ...] ;

Идентификатор – какая длина ???



# Переменные

тип            идентификатор1 [, идентификатор2, ...] ;

Идентификатор – длина произвольная, НО

## **Стандарт C89:**

внешние имена (external names) – значащими являются первые 6 символов;

внутренние имена (internal names) – значащие 31 первый символ;

## **Стандарт C99:**

внешние имена (external names) – значащими являются первые 31 символ;

внутренние имена (internal names) – значащие 63 символа;

в C++ значащими являются первые 1024 символов.

# Объявление переменных

В стандарте C89 переменные должны быть объявлены в начале функции.

Стандарты C99 и выше допускают объявление переменной в любом месте.

# Области видимости

1. Файл – глобальные переменные, видны везде в файле.
2. Прототип функции – формальные параметры, видны только внутри прототипа функции.
3. Функция – локальные переменные, видны внутри функции.
4. Блок - локальные переменные, видны внутри блока { }.

# Типы памяти

	Автоматическая (auto)	Статическая (static)	Динамическая *
Обнуляется на этапе запуска		+	
Ограничена областью видимости	+	+	
Можно задать значение при инициализации	+	+	
Можно узнать размер (sizeof)	+	+	
Сохраняется между вызовами функций		+	+
Может быть глобальной		+	+
Размер массива можно задать во время выполнения	+		+
Размер массива можно изменять			+

# Инициализация

'1' – символ

"1" – строка

20 – десятичное число

020 – восьмеричное число

0x20 – шестнадцатеричное число

# Специальные символы

1. \b – Сдвиг текущей позиции влево
2. \n – Перевод курсора на новую строку
3. \r – Перевод курсора в начало строки
4. \t – Горизонтальная табуляция
5. \a – Звуковой сигнал
6. \' – Символ одинарной кавычки
7. \" – Символ двойной кавычки
8. \\ – Обратный слэш

# Формат типа

1. %c – символьный тип
2. %d (%i) – целочисленный тип
3. %o – восьмеричное представление целого числа
4. %u – целое беззнаковое
5. %x – шестнадцатеричное представление числа
6. %f – вещественный тип
7. %s – строковый тип
8. %p – указатель

# Арифметические операции

+	-	*	/	%	--	++
+=	-=	*=	/=	%=		

Целое / целое = целое;

В чем разница ???

A ++;

++ A;



# Логические операции

>      >=      <      <=      !=      ==

&&      ||      !

# Битовые операции

&	- побитовое И
	- побитовое ИЛИ
^	- исключающее ИЛИ
~	- побитовое отрицание
>>	- сдвиг вправо
<<	- сдвиг влево

# Условные операторы

1. `if (условие)`  
    `{операторы_1;}`  
    `[else`  
        `{операторы_иначе;} ]`
2. `switch (выражение) {`  
    `case значение1:`  
        `операторы1; break;`  
    `case значение2:`  
        `операторы2; break;`  
    `...`  
    `default:`  
        `операторы_иначе; }`

# Циклы

1.     while (условие)  
              {тело цикла}
2.     for (переменная ; условие ; шаг)  
              {тело цикла}
3.     do  
              {тело цикла}  
          while (условие)

# Операторы безусловного перехода

1. `return` – возврат управления из функции
2. `goto` – переход к метке
3. `break` – преждевременное прерывание цикла или оператора `switch`
4. `continue` – прерывание текущей итерации цикла

`exit()` – функция завершения работы программы и передачи управления операционной системе (описана в библиотеке `stdlib`)

# Устройство Даффа

**Метод Даффа** ([англ. Duff's device](#)) в программировании — это оптимизационная реализация последовательного копирования, использующая технику размотки циклов (loop unrolling).

[Duff's device в Си](#)

[Устройство Даффа - Википедия](#)

# Макросы

Макрос – текстовая подстановка, расширяющая и дополняющая основной текст.

`#define` имя выражение | блок команд

## Правила:

1. Заклучайте в скобки все входные параметры, если нет причин поступать иначе.

2. Избегайте двойных подстановок:

```
#define max(x, y) (x > y ? x : y)
```

```
int a = 5, b = 10, rez;
```

```
rez = max(a, b++);
```

Чему будут равны rez и b ???



# Указатели

Указатель (pointer) – переменная, содержащая адрес другой переменной

тип \*переменная;     - здесь \* указывает, что переменная является указателем

&переменная – возвращает адрес переменной

\*переменная – возвращает значение, находящееся по указанному адресу



# extern / static / const

**extern** (внешняя компоновка) – идентификаторы с одним именем, определенные в разных файлах должны рассматриваться компоновщиком как один идентификатор.

**static** (внутренняя компоновка) – область видимости идентификатора определена местом объявления.

**const** – показывает, что данные, на которые ведет параметр-указатель, не будут изменены внутри функции.

# Рефлексия

# Рефлексия



Что запомнилось/вспомнилось с вебинара?



**Заполните, пожалуйста,  
опрос о занятии  
по ссылке в чате**

Спасибо за внимание!

# Приходите на следующий вебинар

тема: Типы данных

дата: 10.08.2023



Коробков Виктор

