Содержание

[ОБЩАЯ ИНФОРМАЦИЯ 2](#_heading=h.gjdgxs)

[Тема занятия](#_heading=h.30j0zll) 2

[Цели и задачи 2](#_heading=h.1fob9te)

[Ожидаемый результат 2](#_heading=h.3znysh7)

[Структура занятия 3](#_heading=h.tyjcwt)

[ОПИСАНИЕ ЗАНЯТИЯ 4](#_heading=h.3dy6vkm)

# ОБЩАЯ ИНФОРМАЦИЯ

### Тема занятия

### Условия ветвления.

**Цели и задачи:**

* Познакомиться с логическими выражениями
* Познакомиться с условиями ветвления
* Научиться использовать в С++

**По результатам занятия слушатель будет знать:**

* Что такое логическое выражение
* Что такое условие ветвления

**По результатам занятия слушатель будет уметь:**

* Создавать элементарные ветвления
* Применять на практике условия ветвления в алгоритмах

### Структура занятия

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тайминг занятия** | |  |  |  | Таблица 1 |
| № | Этапы | Что делает преподаватель | Что делает Слушатель | Время | Общее время |
| 1 | Приветственное слово преподавателя |  |  | 5 мин. | 5 мин. |
| 2 | Повторение пройденного |  |  | 5 мин. | 40 мин. |
| 3 | Теоретическая часть |  |  | 30 мин. |
| 4 | Вопросы по теоретической части |  |  | 5 мин. |
|  | *Перерыв* |  |  | 15 мин. | 15 мин. |
| 5 | Практическая часть |  |  | 40 мин. | 45 мин. |
| 6 | Рефлексия и вопросы |  |  | 5 мин. |

# ОПИСАНИЕ ЗАНЯТИЯ

**Теоретическая часть**

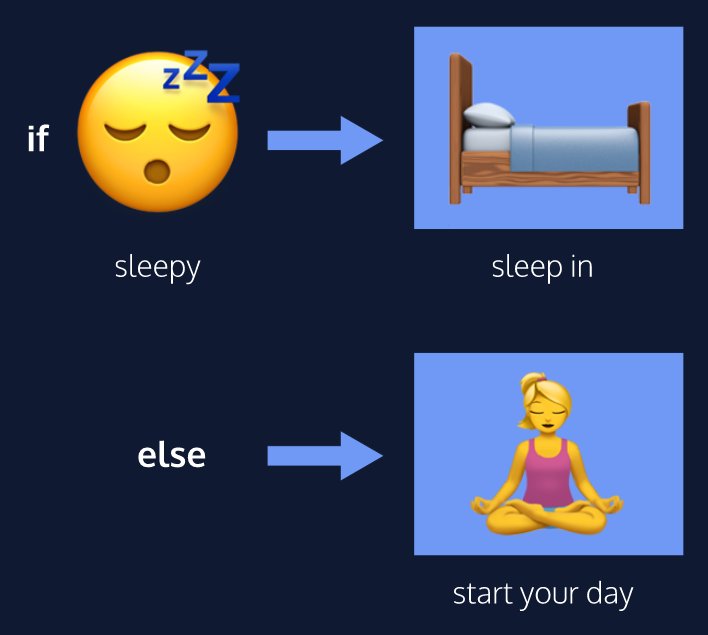
**Преамбула**

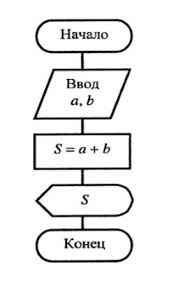
Любая программа, написанная на языке программирования – это реализация алгоритма.

Алгори́тм — конечная совокупность точно заданных правил решения произвольного класса задач или набор инструкций, описывающих порядок действий исполнителя для решения некоторой задачи.

Другими словами, алгоритм – это последовательность команд.

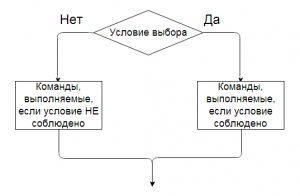
Алгоритм можно реализовать на разных языках программирования, но суть алгоритма остаётся неизменной.





До сегодняшнего для программы, которые мы писали выполнялись построчно (строчка за строчкой), но очень часто бывает необходимо

выполнять определённые действия в зависимости от конкретных условий.



**Выполнение решения**

Выражение if используется для выполнения некоторого кода при соответствии условиям.

Условие определяет, какое выражение будет выполнено. Если условие истинно, то выражение в фигурных скобках будет выполнено.

Если условие ложно, то выражение просто игнорируется, и программа продолжает выполнение после тела оператора if.

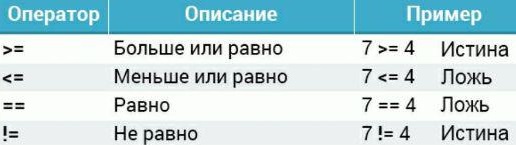
**Оператор if**

Используйте операторы отношения для работы с условиями.

Условие, указанное в операторе if, не требует точки с запятой.

**Операторы отношения**

Дополнительные операторы отношения:

****

**Оператор else**

Оператор if может использоваться вместе с оператором else, который выполняется, когда условие ложно.

Синтаксис:

if (condition)

{  
 //statements  
}  
else

{  
 //statements  
}

**Вложенный оператор if else**

C++ обеспечивает опцией неограниченного использования вложенных операторов if/else.

(пример на слайде 14)

**Конструкция else if**

Помимо проверок условий if и else имеется возможность добавления «промежуточных проверок» else if.

**Практическая часть**

**Задание 1**

«Угадай число»

Напишите программу, которая будет запрашивать положительное целое число (тип данных – int). Затем ваша программа должна вывести на экран:

* Если пользователь угадал число, то вы должны отправить поздравительное предложение в стандартный вывод
* В противном случае – подсказать пользователю «hot» (близко ли он подобрался к загаданному вами числу) или “cold” (либо далеко от него)

Например, загаданное число 50, если пользователь вводит 70.

Программа выдаст «hot», если пользователь введёт 20 -> “cold”

**Задание 2**

Ввести с клавиатуры (стандартный входной поток) номер дня недели, в зависимости от номера вывести название «понедельник», «вторник», «среда», «четверг», «пятница», «суббота», «воскресенье». Если номер больше 7 или меньше 1 – вывести «такого дня недели не существует».

(Файл с кодом лежит в папке code/)