ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ
ДВОИЧНОЕ КОДИРОВАНИЕ. SWITCH CASE

На прошлых занятиях мы научились описывать систему, находящуюся в одном из четырех состояний. Так, наш робот либо находился в состоянии покоя, либо двигался вперед, либо поворачивал направо, либо – налево. Для того чтобы привести робота в одно из этих состояний, мы задействовали всего две кнопки пульта управления.

Но что делать, если состояний много больше?

Например, какая-либо переменная может принимать только буквенные значения. Для русского языка это 33, для английского – 26. Начнем с английского. К 26 буквам добавим пробел, точку, запятую, двоеточие, восклицательный и вопросительные знаки. В итоге получится 32 символа. Все эти символы следует пронумеровать от единицы до 32. Нам уже ясно, что, для того чтобы с алфавитом мог работать компьютер, необходимо свести всю работу с ним к вопросам с парным выбором – `true/false` (да/нет). Или, говоря другими словами, если мы задумали какой-то символ, то компьютер сумеет его отгадать за определенное количество вопросов с ответом да/нет.

Какое это будет количество вопросов?

Давайте посчитаем. Предположим, мы задумали число два. Самым эффективным способом отгадывания будет метод половинного деления. То есть в случае 32 исходных символов первый вопрос будет: «Ваше число меньше шестнадцати?» На этот вопрос только два ответа. Следующий вопрос будет связан с половиной от 16 и т.д. Каждый вопрос вполовину уменьшает количество возможных правильных ответов. Весь процесс угадывания будет устроен следующим образом:

Вопрос	Ответ	Оставшиеся варианты
X<16	Да	0<X<16
X<8	Да	0<X<8
X<4	Да	0<X<4
X<2	Нет	2<X<4
X<3	Да	X=2

Подобным образом можно отгадать любое число, затратив на это всего 5 вопросов. Так же рассуждал Эмиль Бодо, который в 1870 году для своего телеграфа создал клавиатуру, в которой было ровно 5 клавиш. Но эта же технология используется и в современных компьютерах.

Из прошлых занятий мы знаем, что для описания двух состояний достаточно одного вопроса, для четырех – два.

Количество вопросов с парными ответами связано с количеством состояний следующим образом:

Количество состояний	Количество вопросов
2	1
4	2
32	5

То есть 2 в степени количества вопросов (***в***) должно быть больше количества состояний (***С***). При этом максимальное количество вопросов для 32 состояний (английского алфавита) не будет превышать 5.

$$2^B \geq C$$

В этой связи сложная ситуация сложилась с русским языком: в нем, как известно, 33 буквы. Поэтому чтобы не переделывать существующие телеграфные аппараты, пришлось объединить несколько букв (е-ё, и-й, ь-ъ) и ввести вместо знаков препинания буквенные обозначения: ЗПТ, ТЧК и т.д. Таким образом, получили $2^5=32$.

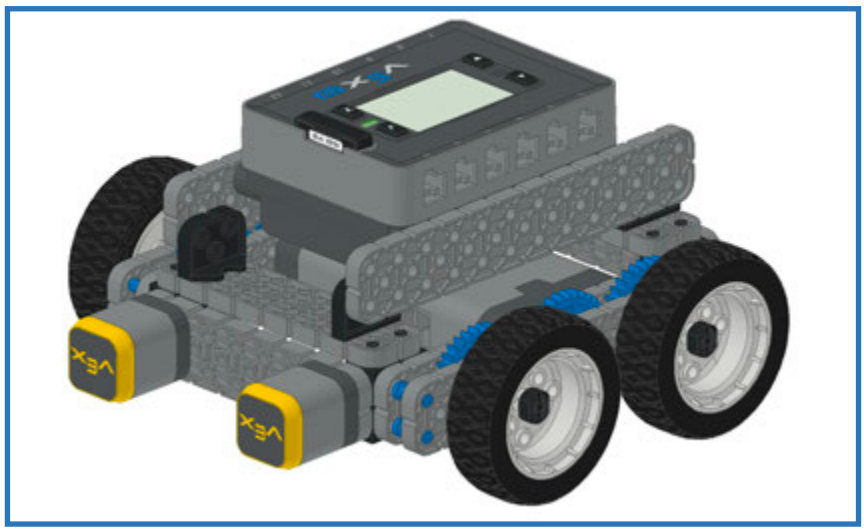
Вообще единицей информации является **бит**. Это то количество информации, которое содержится в вопросе, на который можно однозначно ответить да-нет при условии равной вероятности ответов. Восемь бит – это **байт**. Как мы теперь понимаем, восемь бит, или восемь вопросов, дают возможность описать 28^2=256 состояний. В эти состояния входят 26+26=52 (строчные+заглавные) буквы английского алфавита 33+33=66 – русского, а также цифры, знаки препинания, псевдографика.

Обратимся к нашему роботу. Например, если необходимо реализовать при помощи кнопок такие состояния двигателя, как вращение в обе стороны и остановка, понадобится 2 кнопки, так как состояния 3, а ближайший квадрат числа больше трех - это 4. В свою очередь 4 - это 2 во второй степени.

Как реализовать множественный выбор с использованием if-else? Не получатся ли огромные нечитаемые вложенные ветвления?

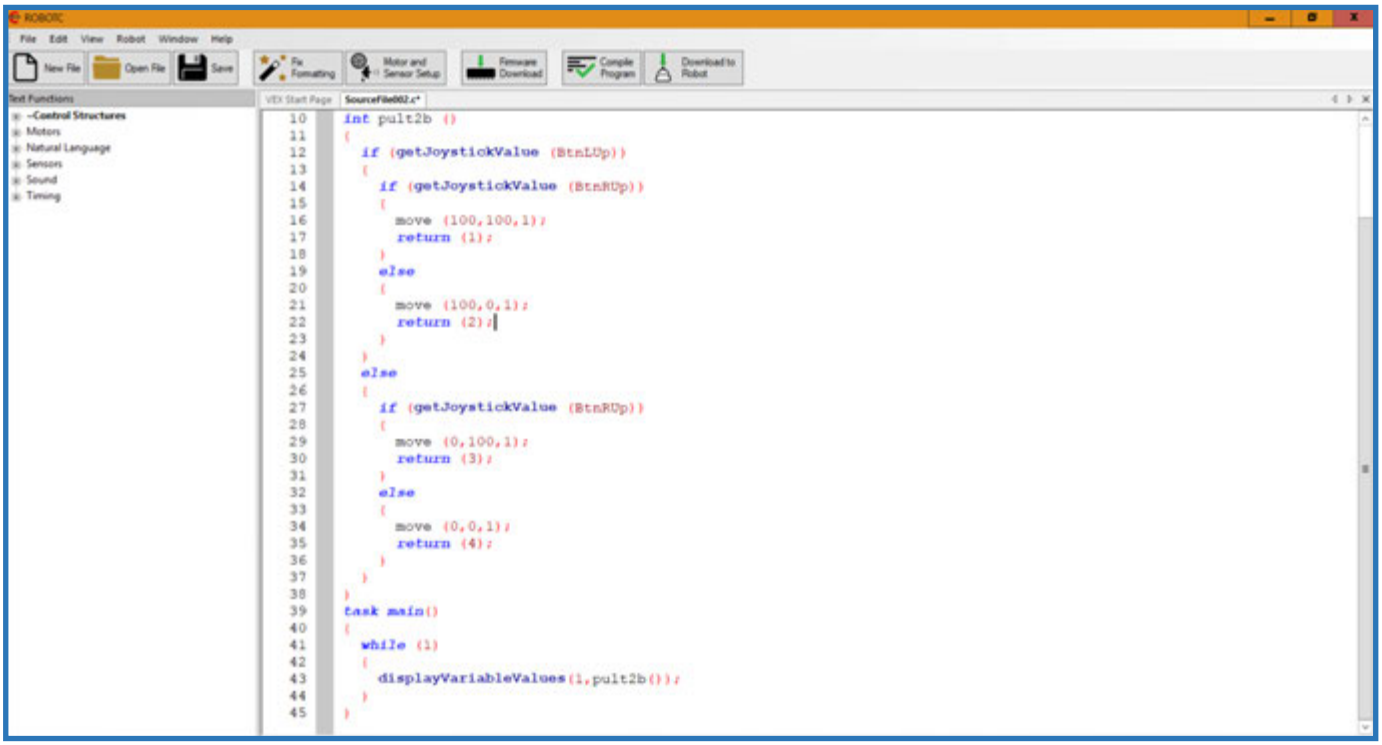
Для этого в языке С есть специальная конструкция, которая называется **`select case`**. Проиллюстрируем ее работу вот на каком примере. Добавим в нашего робота два датчика касания с LED подсветкой, которые умеют менять свой цвет.

Расположим их сзади робота слева и справа. Их предназначение - предупреждать роботов, двигающихся позади, о маневрах идущего впереди робота. Если наш робот начнет останавливаться, оба датчика должны сменить цвет на красный; при повороте направо желтым цветом должен загореться правый датчик, налево – левый. При



движении вперед датчики должны быть погашены. Подобным образом осуществляется и безопасность на автомобильных дорогах общего пользования.

Произведем модернизацию кода с прошлого занятия.



```
10  int pult2b ()
11  {
12    if (getJoystickValue (BtnLUp))
13    {
14      if (getJoystickValue (BtnRUp))
15      {
16        move (100,100,1);
17        return (1);
18      }
19      else
20      {
21        move (100,0,1);
22        return (2);
23      }
24    }
25    else
26    {
27      if (getJoystickValue (BtnRUp))
28      {
29        move (0,100,1);
30        return (3);
31      }
32      else
33      {
34        move (0,0,1);
35        return (4);
36      }
37    }
38  }
39  task main()
40  {
41    while (1)
42    {
43      displayVariableValues(1,pult2b());
44    }
45  }
```

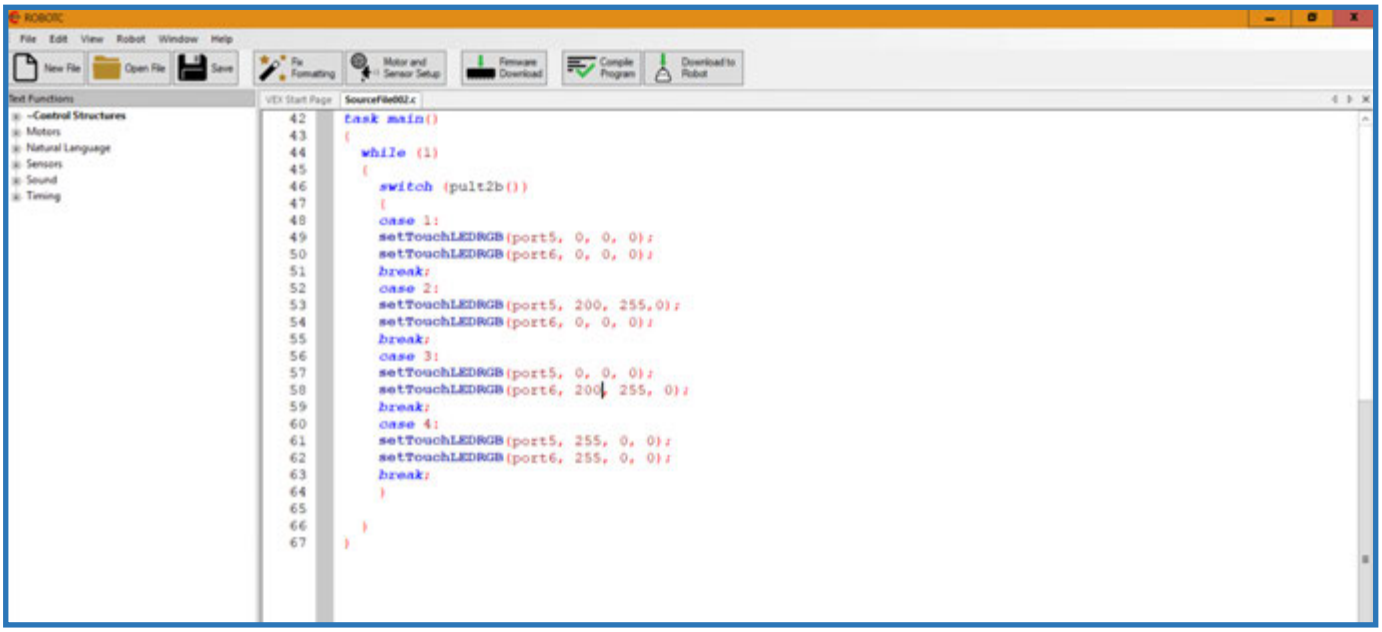
Во-первых, все тело цикла превратим в функцию **int pult2b ()**. Во-вторых, сделаем так, что эта функция будет возвращать номер состояния, в котором находится робот. Всего этих состояний 4:

- робот едет вперед – в 17 строке **return (1)** ;
- робот поворачивает направо – в 22 строке **return (2)**;
- робот поворачивает налево – в 30 строке **return (3)**;
- робот стоит на месте – в 35 строке **return (4)**.

В теле же основной функции находится бесконечный цикл, в теле которого на экран выводится значение функции **pult2b ()**. Данные значения - это числа 1, 2, 3 или 4. Но функция работает таким образом, что при ее вызове происходит управление движением робота.

Осталось научиться подавать сигналы датчикам касания с LED подсветкой. У нас есть 4 состояния, и можно было бы воспользоваться вложенными ветвлениями, но сегодня наша задача познакомиться с новой программной конструкцией **switch case**. Это удобная замена длинной if-else конструкции, которая сравнивает переменную с несколькими константными значениями.

В 46 строке вызываем функцию **switch ()**. Ее аргумент - это переменная, все возможные значения которой нам известны. В нашем случае их 4. Затем в 48, 52, 55 и 60 строках находим ключевое слово **case** и возможное значение. Например, для движения вперед в 48 строке указываем значение 1, светодиоды выключаем (строки 49-50); для поворота направо в 52 строке указываем значение 2, в этом случае необходимо левые светодиоды выключить (54 строка), а правые включить так, чтобы они показывали желтый цвет (53 строка), и т.д.



Обратите внимание, что в строках 51, 55, 59,63 находится кодовое слово **break**. Оно необходимо для того, чтобы робот перестал перебирать варианты, если нужное значение уже найдено.

Итак, конструкция **switch case** - это конструкция множественного выбора. Она позволяет задать условие выбора оператором **switch**, и описать варианты оператором **case**. При этом оператор **switch** сравнивает значение одной переменной, которая указана в условии, с несколькими константами, которые следуют за ключевым словом **case**. Оператор **break** используется для того, чтобы прерывать ход программы в операторе **switch** и передавать управление следующему оператору.