

# Программирование в среде RobotC

Занятие 3: Оператор вывода на  
дисплей

# Взаимодействие с внешним миром

---

## *Задачи:*

- *научиться выводить на экран значения, сохранённые в переменных, чтобы посмотреть результат работы программ;*
- *научиться сохранять в переменные данные, которые поступили из внешнего мира.*

# Взаимодействие с внешним миром

---

## *Данные можно получить:*

- через датчики,
- ввести, используя кнопки на контроллере,
- из других программ,
- различных устройств, например, с web-камеры или микрофона, и др.

*Но мы не предполагаем использования робота, поэтому ввод данных с внешних устройств рассматривать не будем!*

## *Данные можно вывести:*

- на экран,
- в другой файл.

# Операторы вывода

---

*Существует более десяти операторов вывода данных на дисплей NXT. Сейчас мы рассмотрим наиболее частые*

- `nxtDisplayTextLine`
- `nxtDisplayBigTextLine`
- `nxtDisplayCenteredTextLine`
- `nxtDisplayBigCenteredTextLine`

Мы в основном будем иметь дело с первой командой.

Остальные команды отличаются только:

- 1) тем, что позволяют выводить данные шрифтом большего размера (Big);
- 2) тем, что выводимое располагается по центру (Centered).

# Оператор `nxtDisplayTextLine`

*Давайте теперь подробнее разберемся, как устроена и как работает эта команда.*

- Выводить информацию мы будем на дисплей NXT. Он имеет 8 строк, которые занумерованы от 0 до 7.
- Команда позволяет выводить любой текст и тогда она выглядит так:

`nxtDisplayTextLine` (n, "string");

n – обязательный параметр, указывающий номер строки

- Команда позволяет выводить форматные данные:

`nxtDisplayTextLine` (n, "%d", a);

n – номер строки,

"%d" – формат-строка,

a – данные для вывода.

# Оператор `nxtDisplayTextLine`

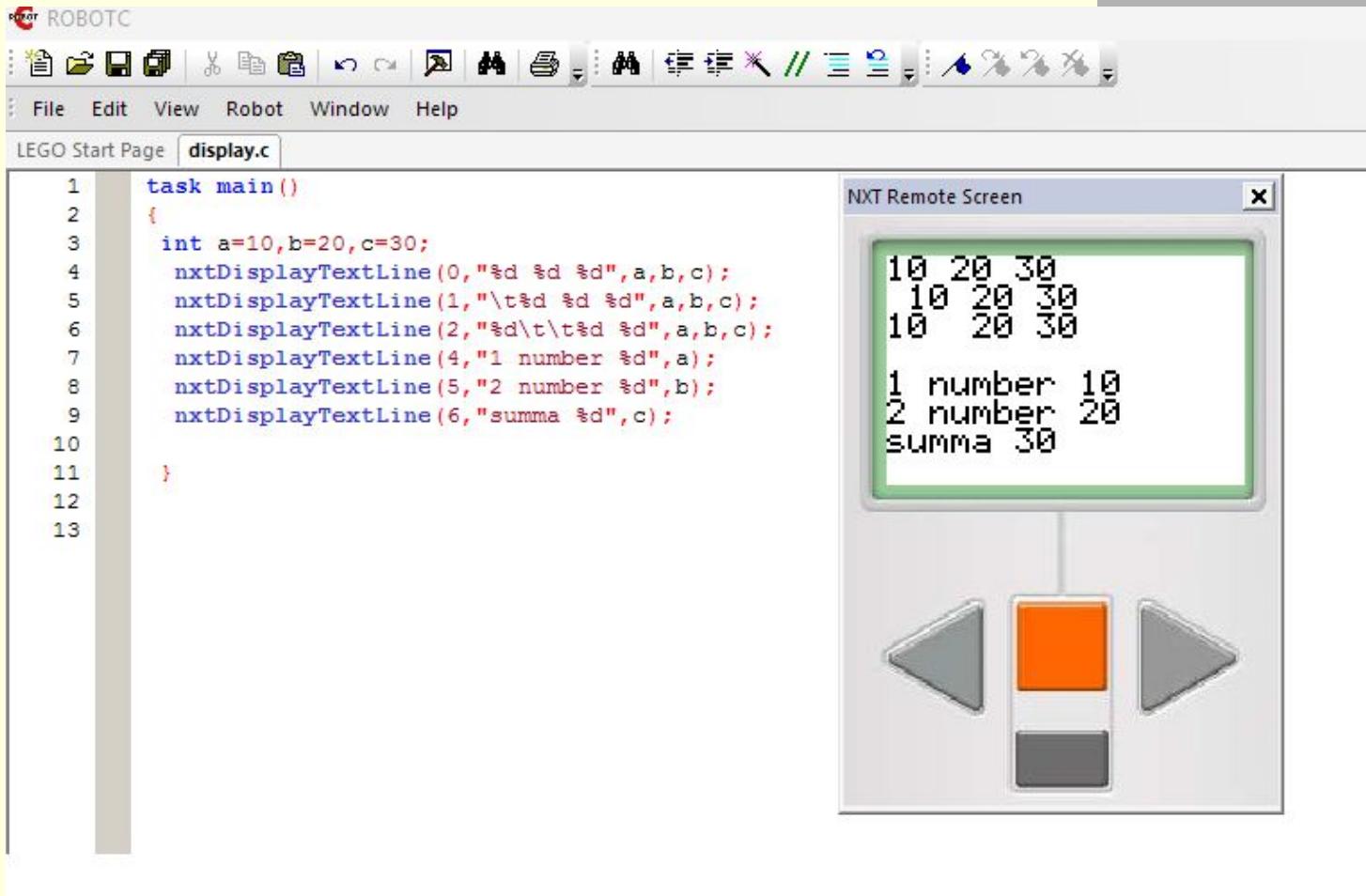
Любой символ в формат-строке относится к одной следующих групп:

- символы, которые выводятся на экран без изменений. Например, "Hello!", "a+b=c", " " (пробел). Русский шрифт выводится не будет.
- escape-последовательности. Начинаются с backslash и выглядят так:
  - `\t` – сдвигает выводимые данные вправо на одну позицию.
  - `\n` – вывод символа "
- спецификаторы формата. Спецификаторы формата всегда начинаются с символа %, и предназначены для вывода на экран значений переменных и выражений. Для каждого типа данных есть свой спецификатор формата.

# Оператор `nxtDisplayTextLine`

- Основные спецификаторы формата:
  - `%d, %i` - целые числа
  - `%f, %g` - вещественные числа
  - `%c` - символы
- Есть и другие спецификаторы формата. Сами спецификаторы формата на экран не выводятся. Вместо них выводятся данные, которые передаются в функцию `nxtDisplayTextLine` после строки форматирования.
- Функция `nxtDisplayTextLine` работает следующим образом. Все символы, заключенные в двойные кавычки, кроме управляющих последовательностей и спецификаторов формата, выводятся на экран. Спецификаторы формата во время вывода заменяются на значения, указанные после формат-строки. Причем, если используется несколько спецификаторов формата, то первый спецификатор заменяется на первое значение, расположенное после формат строки, второй – на второе, и т.д.
- Посмотрим на примерах.

# Примеры



The image shows a screenshot of the ROBOTC IDE. The main window displays a C program named `display.c` with the following code:

```
1  task main()  
2  {  
3    int a=10,b=20,c=30;  
4    nxtDisplayTextLine(0,"%d %d %d",a,b,c);  
5    nxtDisplayTextLine(1,"\t%d %d %d",a,b,c);  
6    nxtDisplayTextLine(2,"%d\t\t%d %d",a,b,c);  
7    nxtDisplayTextLine(4,"1 number %d",a);  
8    nxtDisplayTextLine(5,"2 number %d",b);  
9    nxtDisplayTextLine(6,"summa %d",c);  
10  
11 }  
12  
13
```

To the right of the code editor is a window titled "NXT Remote Screen" which displays the output of the program on a virtual LCD screen. The output is as follows:

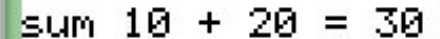
```
10 20 30  
 10 20 30  
10  20 30  
  
1 number 10  
2 number 20  
summa 30
```

The virtual screen also shows a graphical representation of the NXT controller's navigation buttons: a left arrow, a central orange square, a right arrow, and a bottom button.



# Оператор `nxtDisplayTextLine`

```
1 task main()  
2 {  
3   int a=10,b=20,c=30;  
4   char plus='+';  
5   nxtDisplayTextLine(1,"sum %d %c %d %d =\t%d", a, plus, b, c);  
6  
7 }  
8  
9
```



sum 10 + 20 = 30

Итак, в формат-строке (между кавычками) расположены 1) выводимые символы (sum, =, пробелы); 2) спецификаторы формата (%d %c %d %d); 3) escape-последовательности (\t).

По сути, формат строки задаёт некоторый трафарет (шаблон), в который подставляются данные для вывода, в том порядке, в котором они указаны.

*Два основных правила, которые нужно соблюдать при работе с функцией `nxtDisplayTextLine`:*

- количество спецификаторов формата должно совпадать с количеством данных для вывода
- спецификаторы формата должны точно соответствовать типам выводимых данных

# Примеры неправильного использования функции `nxtDisplayTextLine`

---

```
task main()  
{  
    int z = 4; float b = 5.4;  
    nxtDisplayTextLine(1, "%f", z); // нарушено 2 правило  
    // переменная z целого типа, а команда форматирования %f  
    // предназначена для // вывода переменных типа float  
    nxtDisplayTextLine(2, "%d", z, b); //нарушено 1 правило нет  
    // команды форматирования для переменной b.  
}
```

# Модификаторы формата

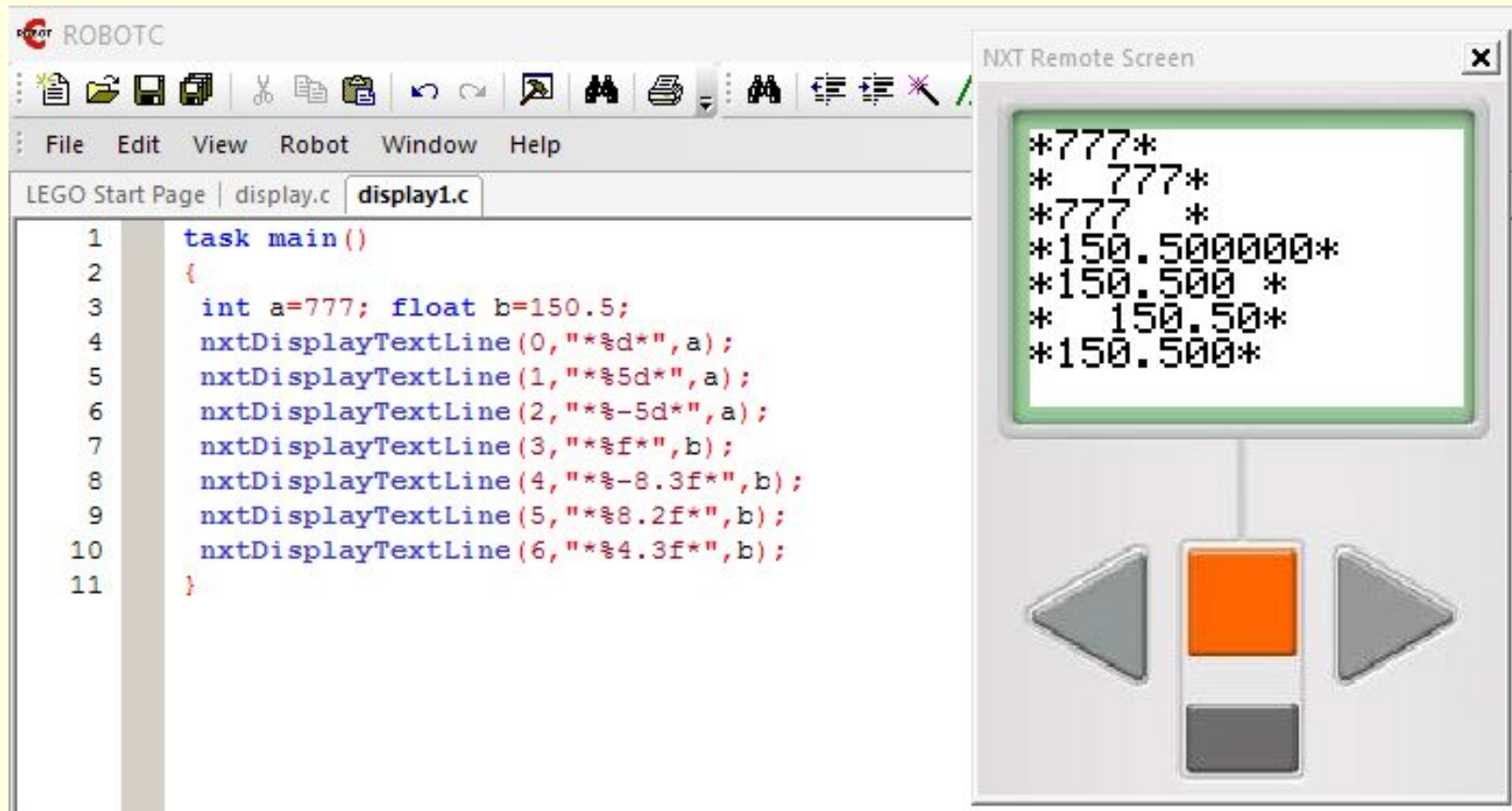
- При выводе вещественных чисел через спецификатор %f на экране появятся числа с шестью знаками после запятой. Обычно такая точность не нужна. К счастью, этим можно управлять. Для этого предназначены модификаторы формата.
- Модификаторы формата записываются между символом % и буквой используемого типа: "%8.3f "
- Первое число обозначает ширину поля, выделяемого для записи числа. Второе число обозначает точность, с которой мы хотим вывести данное вещественное число. В примере под вещественное число мы выделяем 8 символов и хотим видеть 3 знака после запятой.
- Если указанного в ширине количества позиций нам не хватает для вывода числа, то ширина поля увеличивается автоматически, до минимально-возможного количества позиций.
- У первого числа может спереди еще стоять знак минус, например %-8.3f. Этот минус говорит о том, что необходимо выровнять число по левому краю используемой области.
- Для иллюстрации описанных возможностей модификаторов формата, напишем небольшую программу.

# Пример: модификаторы формата

```
task main(){  int a=777; float b=150.5;
nxtDisplayTextLine(0, "%d*", a);
// никаких модификаторов нет, вывод использует минимальную ширину поля
nxtDisplayTextLine(1, "%5d*", a);
// ширина 5 позиций, выравнивание по правому краю
nxtDisplayTextLine(2, "%-5d*", a);
// ширина 5 позиций, выравнивание по левому краю
nxtDisplayTextLine(3, "%f*", b);
// никаких модификаторов нет, выведет используя минимальную ширину поля и
// стандартную точность 6 знаков
nxtDisplayTextLine(4, "%-8.3f*", b);
// ширина 8 позиций, выравнивание по левому краю, 3 знака после запятой
nxtDisplayTextLine(5, "%8.2f*", b);
// ширина 8 позиций, 2 знака после запятой, выравнивание по правому краю
nxtDisplayTextLine(6, "%4.3f*", b);
// число позиций 4, точность 3 знака после запятой, но этого мало поэтому ширина поля
// увеличивается до минимально-возможного значения
}
```

Знаки \* стоят специально вокруг каждого числа, чтобы можно было увидеть, что означает ширина поля для вывода и как работает выравнивание по левому краю.

# Пример: модификаторы формата



The image shows the ROBOTC IDE interface with a C program named `display1.c` and a window titled "NXT Remote Screen" displaying the output of the program.

```
1  task main()  
2  {  
3      int a=777; float b=150.5;  
4      nxtDisplayTextLine(0, "%d", a);  
5      nxtDisplayTextLine(1, "%5d", a);  
6      nxtDisplayTextLine(2, "%-5d", a);  
7      nxtDisplayTextLine(3, "%f", b);  
8      nxtDisplayTextLine(4, "%-8.3f", b);  
9      nxtDisplayTextLine(5, "%8.2f", b);  
10     nxtDisplayTextLine(6, "%4.3f", b);  
11 }
```

The output displayed on the NXT Remote Screen is:

```
*777*  
*  777*  
*777 *  
*150.500000*  
*150.500 *  
* 150.50*  
*150.500*
```

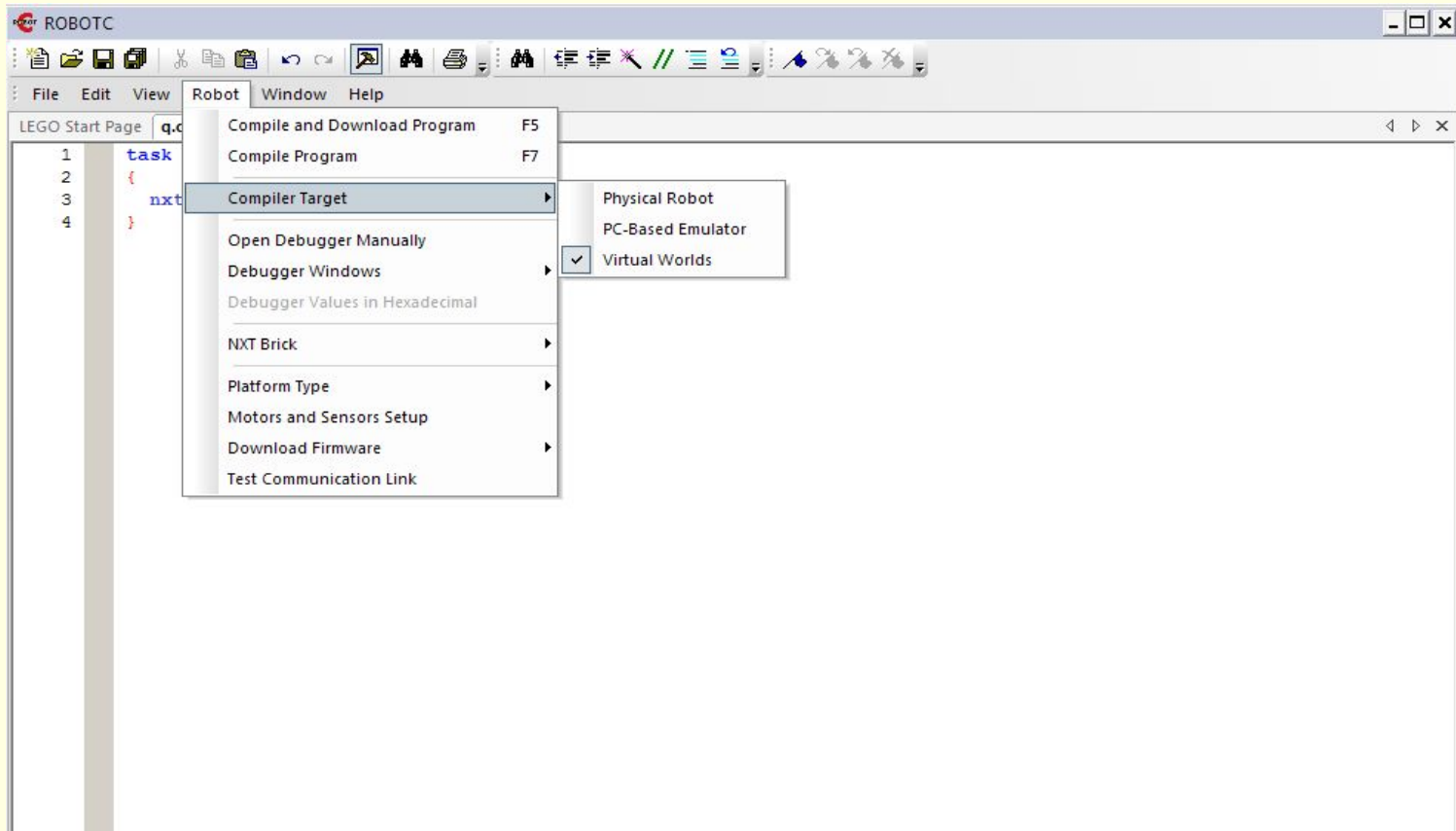
# Первая программа

---

- Теперь мы напишем первую программу на RobotC. Обычно, программу загружают на контроллер, но мы сейчас учимся программировать без использования робота. Поэтому мы будем эмулировать работу контроллера на компьютере.
- Запускаем RobotC.
- Заходим на вкладку File->New->New file. Перед нами чистый экран.
- Не забываем перейти на английскую раскладку. Набираем программу из первого разобранного примера.
- Когда всё набрали. Нажимаем F7 – компиляция программы. Среда просит придумать название программы и папку, куда будет сохраняться она.
- После исправления ошибок, наконец программа написана правильно!

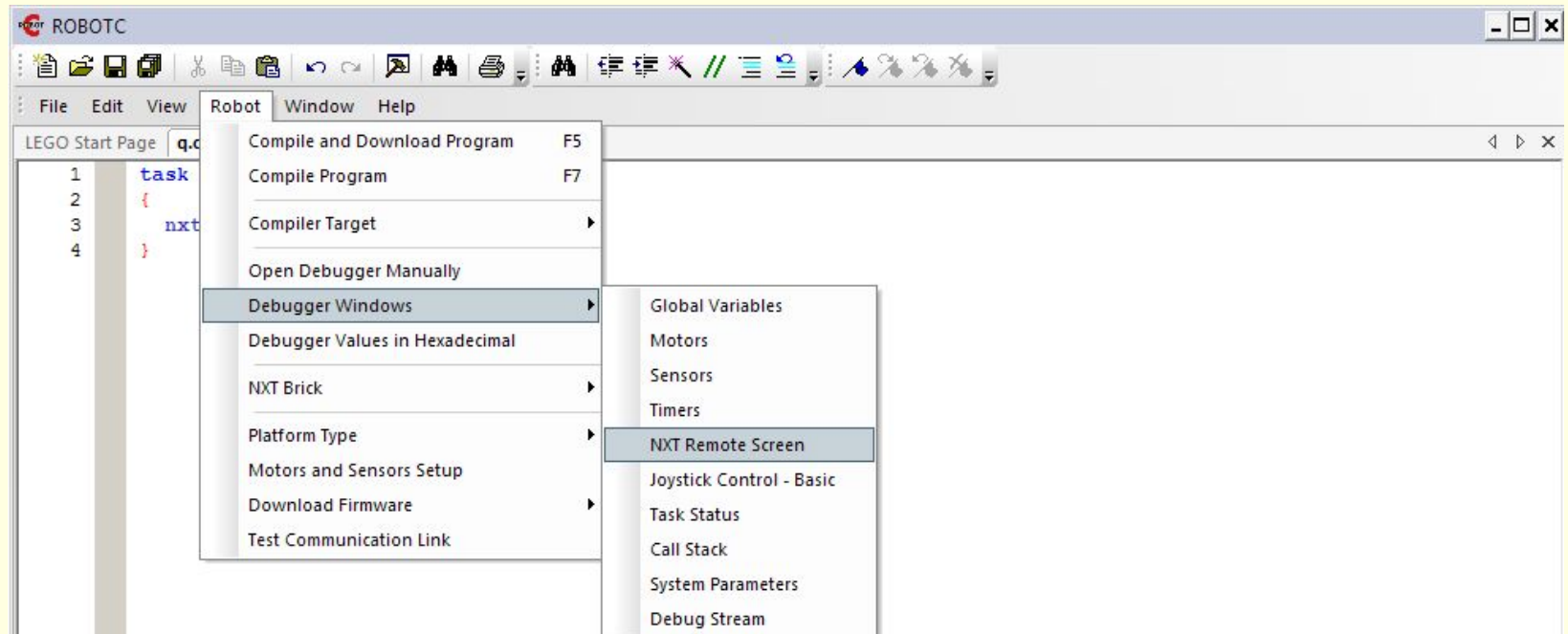
# Первая программа

- Выбираем Robot->Compiler Target-> ставим галочку Virtual Worlds



# Первая программа

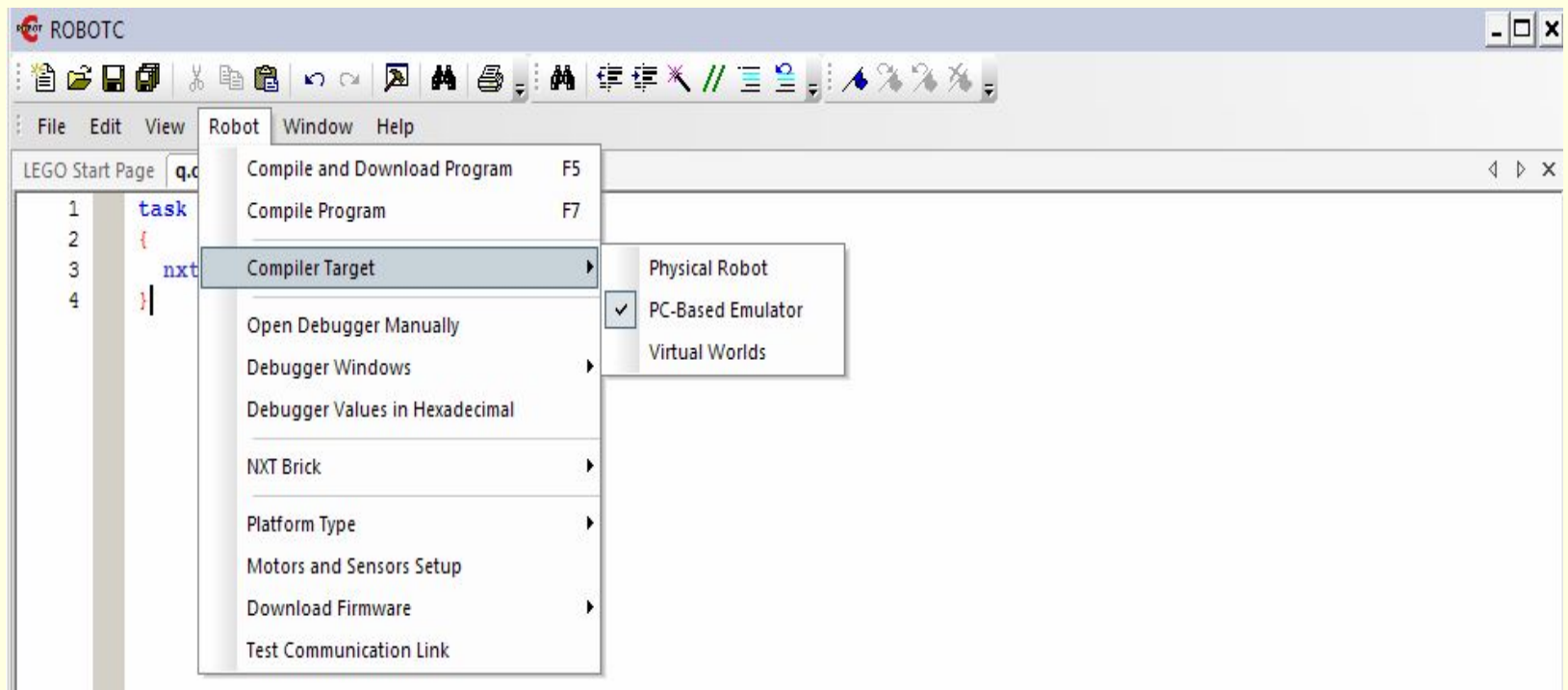
- После этого нажимаем F5 – запуск программы. Сейчас система, возможно, выдаст сообщения об ошибке. Игнорируем его.
- Robot->Debugger Windows->NXT Remote Screen





# Первая программа

- Снова переключаемся на эмулятор, но теперь после запуска программы у нас будет появляться экран NXT



# Первая программа

---

- Теперь можно запускать программу такую, как на слайде 8. Нажимаем F5. Затем на появившемся окне Start. В результате должно появиться то, что на слайде 8.
- В качестве задания надо набрать все программы, которые были на сегодняшнем уроке. Запустить их и убедиться, что на экране появляется тоже, что и в примерах.
- Можно поэкспериментировать и попробовать выводить различные символы и числа.
- Посмотрите как работают операторы `nxtDisplayBigTextLine`, `nxtDisplayCenteredTextLine`, `nxtDisplayBigCenteredTextLine`

# Завершение занятия 3

---

На этом уроке мы познакомились с операторами вывода. Теперь вы можете проверить работу своих программ в среде RobotC, даже не имея робота. В следующий раз будут разные задачи по пройденному материалу.

На этом занятии 3 завершено.