05. Работа с файлами в Ruby

Практические задания.

№ 05.0

Удалите каталог **~/CodeClub-IoT/**. Загрузите свежую версию учебных материалов командой:

```
git clone https://github.com/mike-shock/CodeClub-IoT.git
```

№ 05.1

1. Запустите редактор **Geany** из раздела «Программирование» в главном меню.
2. Создайте в редакторе Geany новый файл **read_file-1.rb** и введите в него с клавиатуры текст программы:

```
#!/usr/bin/ruby

file_name = './thermo_log.txt'

File.open(file_name, 'r') do |f| # открыть файл на чтение (Read)
  while (line = f.gets) do # читать строки в line до конца файла
    puts line              # выводить значение line в STDOUT
  end
end
```

3. Сохраните программу **read_file-1.rb** в рабочем каталоге **~/projects/**.
4. Запустите программу на выполнение из раздела меню «Сборка», пункт «Execute» (выполнить).
5. Понаблюдайте результаты её работы в открывшемся терминальном окне. Закройте терминальное окно.

№ 05.2

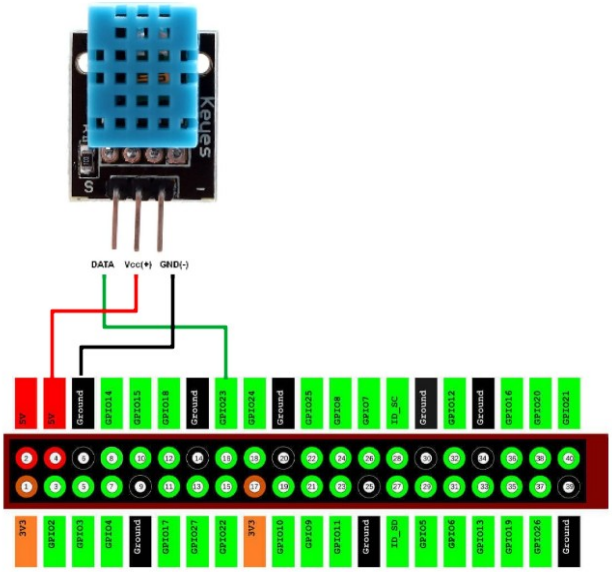
1. Измените программу **read_file-1.rb** так, чтобы разделить считанную строку на 3 части по разделителю «,» и присвоить в разные переменные значения даты, времени и температуры, например, так:
   ```
   date, time, temperature = line.split(',')
   ```
2. Добавьте в программу форматированный вывод значений даты, времени и температуры с помощью метода **printf**.
   ```
   printf("%s %s %f\n", date, time, temperature.to_f)
   ```
   Для форматирования данных разных типов используются такие указания форматирования:
   - **%s** — строка
   - **%12s** — строка в поле вывода шириной 12 символов (с выравниванием вправо)
   - **%-12s** — строка в поле вывода шириной 12 символов (с выравниванием влево)
   - **%d** — целое число (в десятичной системе счисления)
   - **%5d** — целое число в поле вывода шириной 12 символов (с заполнением пробелами)
   - **%05d** — целое число в поле вывода шириной 12 символов (с ведущими нулями)
   - **%f** — дробное число (в десятичной системе счисления)
   - **%7.3f** — дробное число в поле вывода шириной 7 символов с 2-мя цифрами после точки

   Проверьте, как будет выводить данные формат **"'%12s' '%-10s' %07.3f\n"**.
3. Сохраните программу под именем **read_file-2.rb** в рабочем каталоге **~/projects/**.

4. Запустите программу на выполнение и понаблюдайте результаты её работы в терминальном окне.

№ 05.3

1. Выключите Raspberry Pi. Смонтируйте на макетной плате подключение датчика температуры и влажности **DHT11** по этой схеме подключения:



2. Программа **~/CodeClub-IoT/samples/dht11.rb** на **Ruby** опрашивает датчик температуры и влажности **DHT11** и выводит показания датчика.

```
#!/usr/bin/ruby

require "dht_sensor"                # подключить библиотеку
sensor = DHT11.new(23)              # создать объект sensor класса DHT11

12.times do                                  # выполнить в цикле 12 раз
  sensor.read_data                           # считать показания датчика
  print "Температура: "                              # вывести температуру
  printf "%7.4f°C ",  sensor.celsius                 # в градусах Цельсия
  printf "%7.4f°Ré ", sensor.reaumur                    # ... Реомюра
  printf "%7.4f°F ",  sensor.fahrenheit                 # ... Фаренгейта
  printf "%7.4f K ",  sensor.temperature(:kelvin) # ... Кельвина
  printf "Влажность: %f %%\n", sensor.humidity # влажность в %
  sleep 1                                    # подождать 1 секунду
end
```

3. Измените программу так, чтобы она выводила строки с датой (ГГГГ-ММ-ДД), временем (ЧЧ:ММ:СС) и показаниями температуры (ТТ.ТТТТ) и ылажности (ВВ.ВВВ) в формате CSV:
   ГГГГ-ММ-ДД,ЧЧ:ММ:СС,ТТ.ТТТТ,ВВ.ВВ
4. Выполните программу из редактора **Geany** и понаблюдайте результаты её работы в терминальном окне.
5. Выполните программу в терминальном окне с перенаправлением вывода в файл **~/projects/dht11-log.txt**.

№ 05.4

1. Переделайте программу **read_file-2.rb** так, чтобы она правильно читала файл **dht11-log.txt**.
2. Сохраните программу под именем **read_dht_log.rb** в рабочем каталоге **~/projects/**.
3. Запустите программу на выполнение и проверьте правильно ли она работает.