



04. Программирование на **Ruby**

Практические задания.

№ 04.1

1. В главном меню выберите раздел «Программирование» и запустите из него редактор **Geany**.
2. Создайте в редакторе Geany новый файл **cycles.rb** и введите в него с клавиатуры текст программы:

```
#!/usr/bin/ruby

minimum = 100          # сначала будет такое минимальное число
maximum = 0            # сначала будет такое максимальное число

100.times do |i|       # повторить 100 раз, помещая число в i
  r = rand(100)         # поместить в r случайное число от 0 до 99
  print i, ' ', r, "\n" # напечатать i и r
  if r < minimum        # сравнить, меньше ли r значения в minimum
    minimum = r        # если да, то поместить r в minimum
  end
  # из 2-х чисел [в, списке] выбрать максимум и поместить в maximum
  maximum = [maximum, r].max
  # minimum = [minimum, r].min # минимум можно вычислять так же
end

print "Минимальное случайное число = ", minimum, "\n"
print "Максимальное случайное число = ", maximum, "\n"
```

3. Сохраните программу **cycles.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, запустив несколько раз. Не забывайте каждый раз закрывать терминальное окно.

№ 04.2

1. Напишите на **Ruby** программу **happy.rb**, которая в цикле находит все «счастливые» билеты в рулоне с номерами билетов от 000001 до 999999. Билет считается счастливым, если сумма первых трёх цифр равна сумме последних трёх цифр.
2. Выводите все номера счастливых билетов на экран командой **print**.
3. Чтобы сохранить этот вывод в файл, запустите эту программу из терминального окна с перенаправлением стандартного вывода с экрана в файл:
~/projects/happy.rb > happy.txt
4. Подсчитайте общее количество найденных счастливых билетов и выведите его на экран.
5. Можно также сосчитать количество счастливых билетов в каждой тысяче и вывести на экран их количество и процент на каждую тысячу.

№ 04.3

1. Программа **~/CodeClub-IoT/samples/thermo.rb** на **Ruby** опрашивает встроенный датчик температуры Raspberry Pi и выводит текущую дату, время и показания датчика.

```
require "thermal_sensor"

sensor = RaspberryPi::ThermalSensor.new

(1..12).each do |n|
  sensor.read_data
  printf("Дата: %s. Время: %s. Температура: %7.4f°C \n",
    Time.now.strftime("%d.%m.%Y"),
    Time.now.strftime("%H:%M:%S"),
    sensor.celsius)
  sleep 1
end
```

2. Выполните программу из редактора **Geany** и наблюдайте результаты её работы в терминальном окне.
3. Сохраните программу в рабочем каталоге `~/projects/` под именем `thermo_csv.rb`.
4. Измените программу так, чтобы она выводила только одну запись с датой (ГГГГ-ММ-ДД), временем (ЧЧ:ММ:СС) и показанием температуры (ТТ.ТТТТ) в формате CSV:
ГГГГ-ММ-ДД, ЧЧ:ММ:СС, ТТ.ТТТТ
5. Сделайте программу исполняемой и запустите её в терминальном окне.
6. Для запуска программы `thermo_csv.rb` сделайте исполняемый командный скрипт `thermo_csv.sh`:
#!/bin/bash
cd ~/projects
./thermo_csv.rb >> thermo_log.txt
7. Добавьте скрипт `thermo_csv.sh` в расписание демона `cron` для запуска *каждые 5 минут*, для чего запустите редактор расписания:
crontab -e
и введите нужное правило:
*/5 * * * * /home/pi/projects/thermo_csv.sh
8. Проверьте, что всё работает правильно: строки с данными о температуре в требуемом формате добавляются в файл протокола `thermo_log.txt` каждые 5 минут.