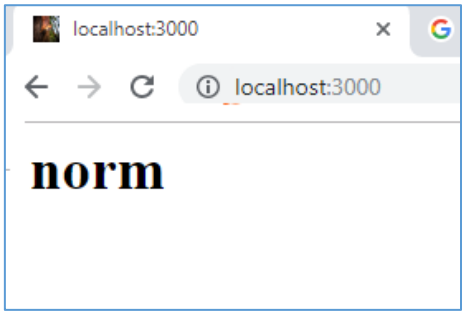
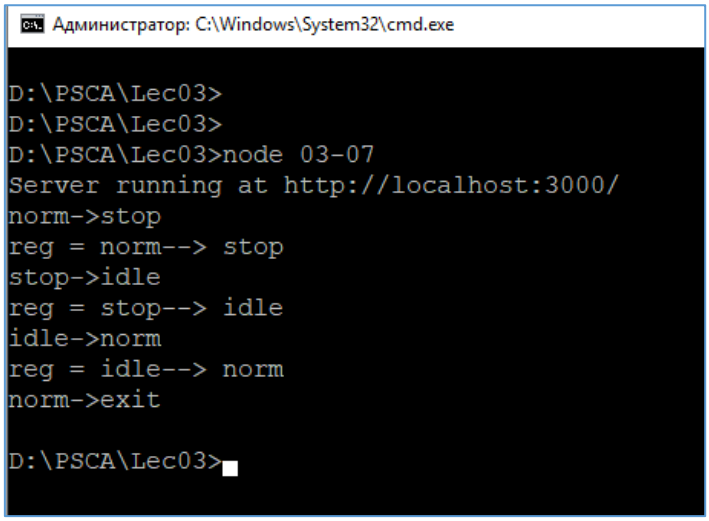
Задание 01

1. Разработайте серверное приложение **03-01**, которое на запрос **http://localhost:5000** возвращает страницу, отражающую состояние приложения (см. рис.).



2. Приложение может находиться в четырех состояниях: **norm**, **stop**, **test**, **idle**.
3. Состояние приложения переключается с помощью стандартного системного ввода, который назначен на консоль. Консоль в приглашении (prompt) указывает текущее состояние приложения.
4. Пользователь может ввести новое состояние (**norm**, **stop**, **test**, **idle**). При корректном вводе состояния осуществляется переключение состояния приложения.
5. При ошибочном вводе режима ошибочно введенная последовательность символов просто отображается, но переключение режима не осуществляется.
6. Допускается ввод состояния exit, которое приводит к завершению приложения (см. рис.)

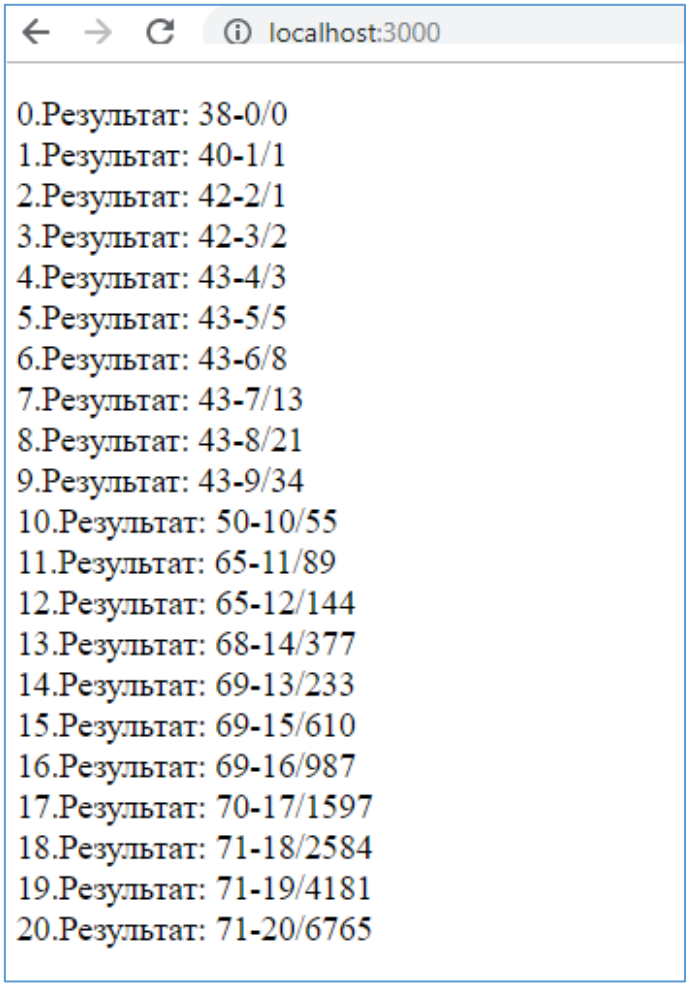


Задание 02

7. Разработайте серверное приложение **03-02**, которое на GET-запрос вида **http://localhost:5000/fact?k=3** возвращает ответ, в теле которого содержится сообщение в json-формате вида **{k:3, fact:6}**, где **k** – полученное в качестве параметра значение, а **fact** – значение факториала.
8. Для расчета факториала используйте рекурсивный алгоритм.
9. Проверьте работоспособность приложения с помощью **POSTMAN**.

Задание 03

10. Доработайте приложение **03-02** таким образом, чтобы на GET-запрос приложение отправляло HTML-страницу, содержимое которой формировалось бы с помощью JS.
11. В цикле x = 1,…,20 с помощью функции fetch сделайте GET-запросы к **http://localhost:5000/fact?k=x** и содержимое ответа выведите в окно браузера, примерно так, как это представлено на следующем рисунке.



12. Результаты вычислений должны иметь следующий вид **t-k/fact**, где **t** – количество миллисекунд, прошедшее с момента начала работы цикла запросов, **k** – параметр, пересылаемый серверу, **fact** факториал k.
13. Запустите приложение и запишите общую продолжительность всего цикла запросов.

Задание 04

14. Разработайте приложение **03-04** на основе приложения **03-02**, но функцию для вычисления факториала реализуйте асинхронной с помощью механизма **process.nextTick**.
15. Выполните аналогичные заданию 3 замеры.

Задание 05

16. Разработайте приложение **03-05** на основе приложения **03-02**, но функцию для вычисления факториала реализуйте асинхронной с помощью механизма **setImmediate**.
17. Выполните аналогичные заданию 3 замеры.

Задание 06. Ответьте на следующие вопросы.

18. Перечислите основные глобальные объекты Node.js и поясните их назначение.
19. Поясните понятие «асинхронная функция».
20. Поясните понятие «стандартные системные потоки».
21. Поясните назначение функций **process.nextTick**, **setImmediate**. Поясните в чем их разница.