Установка и настройка среды программирования для Lua

**Описание.**

В документе рассматриваются установка и настройка IDE ZeroBrane Studio и Visual Studio Code для Lua под ОС Windows 10 x64 и Ubuntu 22.04, а также установка самого интерпретатора языка Lua.

**Порядок сборки и настройки.**

* + - **Windows 10 x64.**
      * **Установка Lua**

Для загрузки интерпретатора следует перейти на сайт: <https://www.lua.org/home.html>, далее выбрать раздел “Download”. Для установки под windows следует загрузить бинарные файлы, поэтому в подразделе “Tools” следует нажать на ссылку “LuaBinaries”. В разделе “History” нужно выбрать самую последнюю версию (на момент написания документа: Lua 5.4.2 - Release 1), после чего произойдет перенаправление на sourceforge.net. Здесь требуется перейти в “Tools Executables” и следом выбрать “lua-5.4.2\_Win64\_bin.zip”. После загрузки архива его следует распаковать в нужную директорию. Архив содержит исполняемые файлы и файл динамической бибилиотеки. Основной исполняемый файл lua54.exe. К нему целесообразно прописать путь в системной переменной Path.

* + - * **Установка ZeroBrane Studio**

Данная IDE предназначена исключительно для написания программ на языке Lua.

Для загрузки IDE на сайте <https://studio.zerobrane.com/> следует перейти в раздел Download, далее нажать “Take me to the download page this time »” и выбрать “Windows 32bit (exe installer)”.

После чего следует запустить исполняемый файл и выбрать нужную директорию для установки. После установки откроется окно IDE (рис. 1).

В данную IDE уже входит отладчик, а также несколько версий Lua. Выбор конкретной версии осуществляется из меню Project: Project->Lua Interpreter. На момент написания документа последняя версия интерпретатора была 5.3. Добавить новую версию интерпретататора (например, 5.4) не представляется возможным. Это одна из причин, по которым можно обратиться к другим IDE.

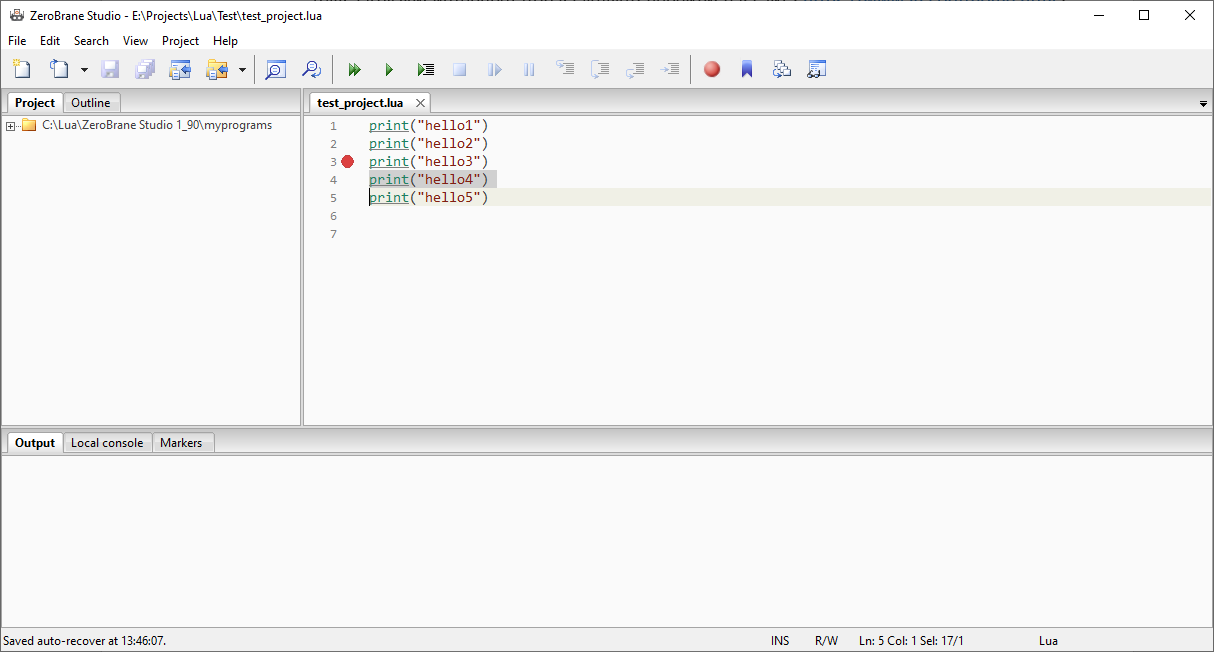


Рис. 1. Главное окно ZeroBrane Studio

* + - * **Установка и настройка Visual Studio Code**

Загрузка IDE осуществляется с сайта: <https://code.visualstudio.com/>

После установки следует создать каталог, в котором будут располагаться скрипты для Lua, а затем добавить этот каталог в VisualStudio Code через меню File: File->Open Folder…

Далее нужно добавить файл скрипта: File->New file… В появившемся окне следует выбрать “Text File” (рис. 2)

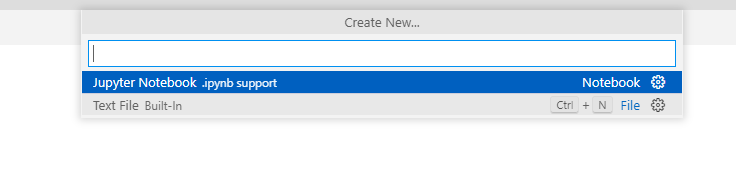


Рис. 2. Создание текстового файла

В созданном файле следует нажать на ссылку “Select a language”, а затем выбрать Lua (рис. 3), затем сохранить файл под нужным именем (File->Save)

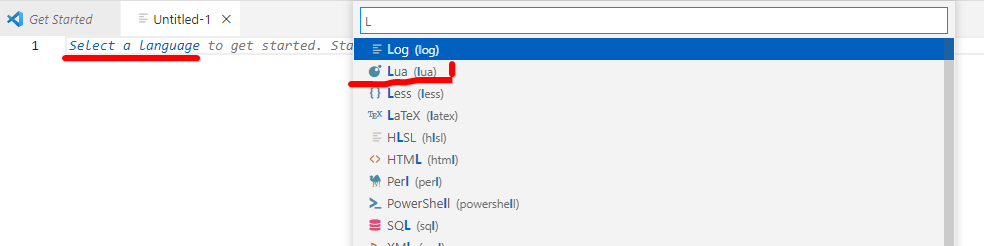


Рис. 4. Выбор языка программирования для поддержки синтаксиса

* *Запуск скрипта внутри PowerShell.*

Для запуска скрипта без отладчика можно воспользоваться терминалом. Для его вызова следует из меню выбрать Terminal: Terminal->New Terminal. Вслед за тем откроется командная строка PowerShell (рис. 5).

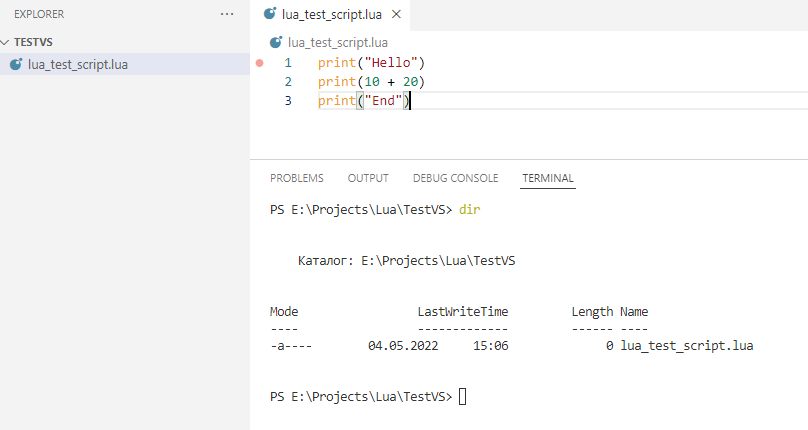


Рис. 5. Окно командной строки

Если путь к интерпретатору прописан в переменной path, то можно запустить скрипт следующей командой: lua54 .\lua\_test\_script.lua

Результат показан на рис. 6.

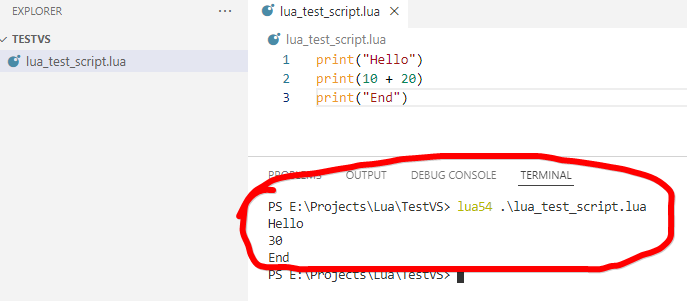


Рис. 6. Демонстрация работы скрипта

* *Запуск скрипта в режиме отладки.*

Для установки поддержки отладчика можно поступить следующим образом: в меню Run выбрать Start Debugging, после чего в открывшемся окне предупреждения выбрать “Find lua extension”. Из открывшегося списка (рис. 7) следует выбрать первый пункт и нажать install.

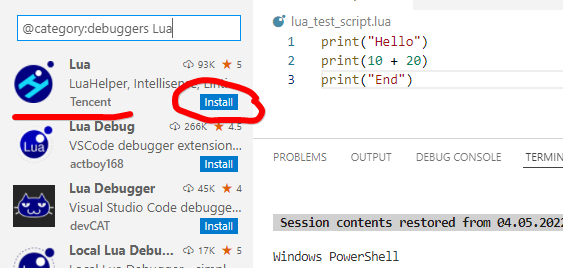


Рис. 7. Установка расширения для поддержки отладчика

После чего нужно в меню Run опять выбрать Start Debugging и в появившемся окне настроек нажать на пункт “LuaHelper: Debug” (рис. 8)

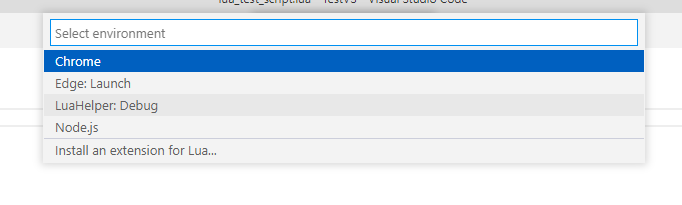


Рис. 8. Выбор среды для отладки

Будет произведен переход в режим отладчика. Однако для создания конфигурации следует дополнительно выбрать Add configuration выпадающем списке (рис. 9).

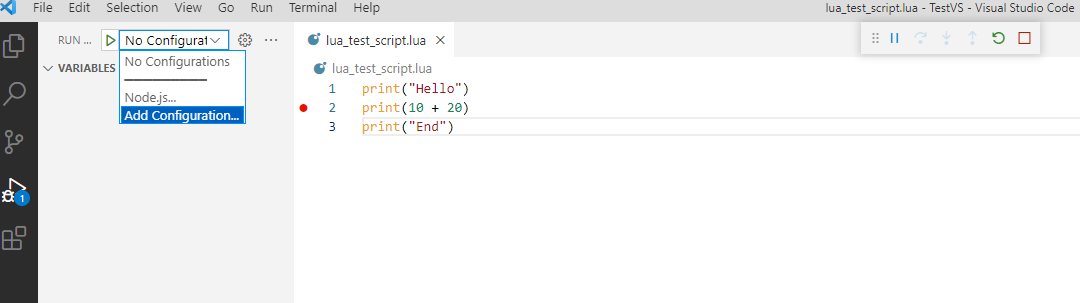


Рис. 9. Выбор конфигурации отладчика

После чего повторно выбрать “LuaHelper: Debug”. После этого в выпадающем списке появится нужный пункт, а в рабочем каталоге будет автоматически создан файл launch.json (рис. 10).

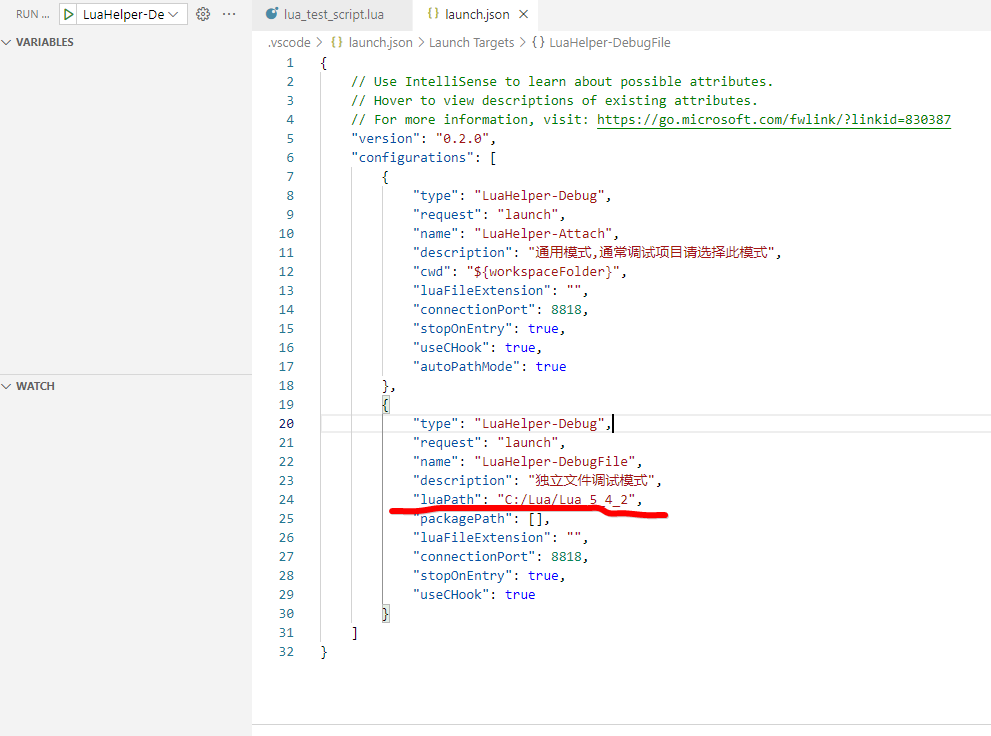


Рис. 10. Демонстрация LuaHelper и файла launch.json

В данном файле следует настроить путь для lua. Для этого следует путь к интерпретатору в поле “luaPath” в элементе с полем “name”: “LuaHelper-DebugFile”. В данном случае этот путь будет “C:/Lua/Lua\_5\_4\_2”. После чего нужно сохранить файл launch.json, перейти на вкладку с lua\_test\_script.lua, установить точку останова напротив нужной строки и выбрать Run->Start Debugging. Результат отладки представлен на рис. 11. В левой панели можно видеть локальные и глобальные переменные, а в окне терминала – вывод скрипта.

* *Запуск скрипта как задачи.*

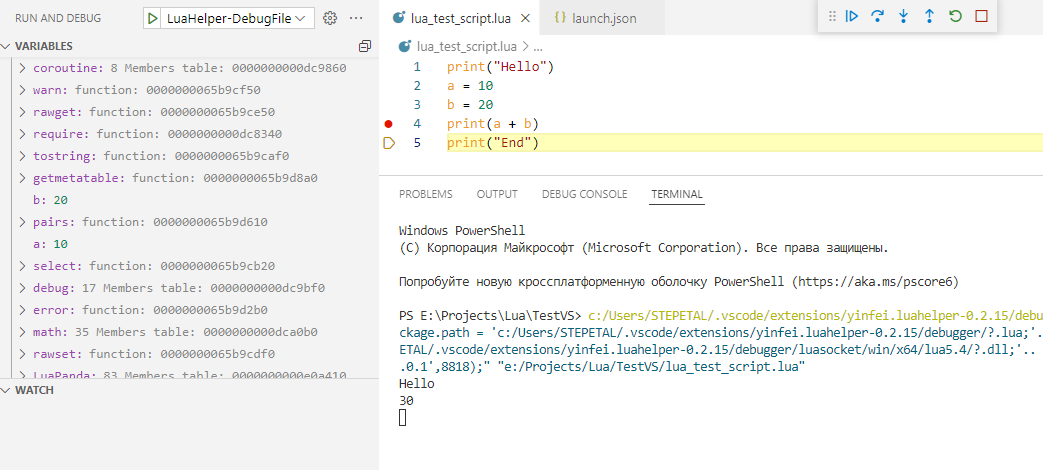
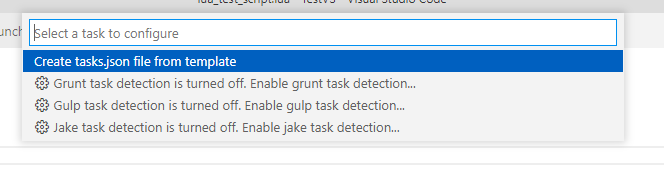


Рис. 11. Режим отладки скрипта

В Visual Studio Code есть возможность сборки решения со скриптом Lua. Для этого следует перейти в меню Terminal: Terminal->Run Build Task… После чего в открывшемся окне нажать на “No build task to run found. Configure build task…”. Далее нажать на “Create tasks.json file from template”, затем на “Others” (рис. 12)



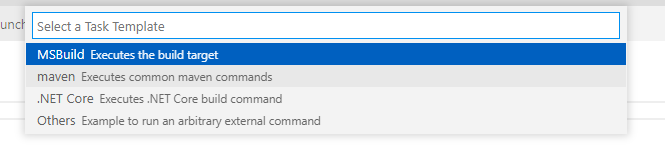


Рис. 12. Выбор создания tasks.json из шаблона

В рабочем каталоге будет создан файл “tasks.json” (рис. 13).

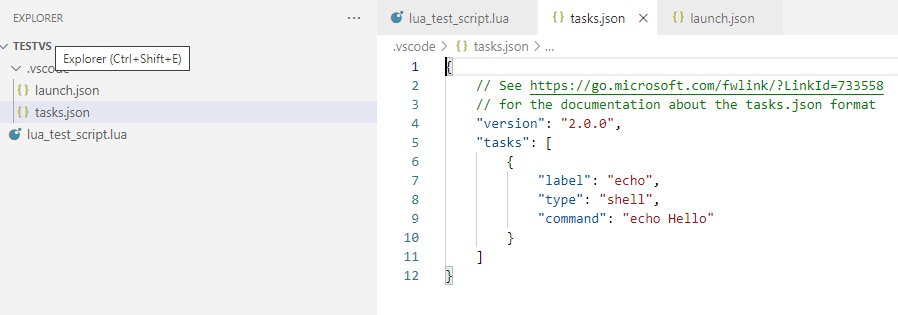


Рис. 11. tasks.json до модификации

Его следует отредактировать (<https://www.youtube.com/watch?v=XCwXWOe6VcU> и https://pikuma.com/files/youtube/tasks.json) следующим образом (рис. 12):

{

"version": "2.0.0",

"tasks": [

{

"label": "Run Lua",

"type": "shell",

"command": "lua54",

"args": [

"${file}"

],

"group": {

"kind": "build",

"isDefault": true

}

}

]

}

Рис. 12. tasks.json после модификации

Для запуска скрипта следует выбрать: Terminal->Run Build Task… При этом будет открыта обычная командная строка, а не терминал PowerShell.

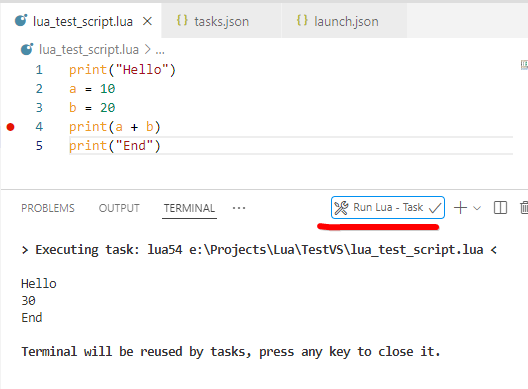


Рис. 13. Демонстрация запуска скрипта lua\_test\_script.lua

* + - **Ubuntu 22.04**
      * **Установка Lua**

Для установки интерпретатора следует в терминале выполнить команду: sudo apt-get install lua5.4 (рис. 1)

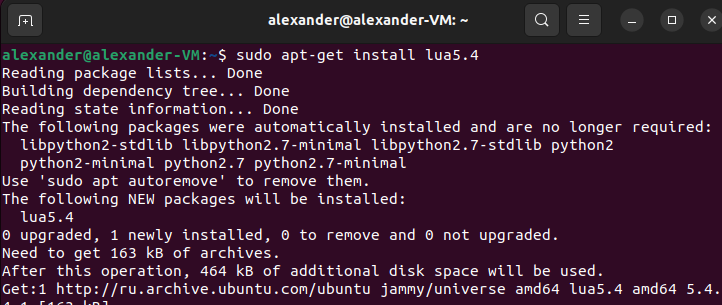


Рис. 13. Демонстрация запуска скрипта lua\_test\_script.lua

* + - * **Установка ZeroBrane Studio**

Для загрузки IDE на сайте <https://studio.zerobrane.com/> следует перейти в раздел Download, далее нажать “Take me to the download page this time »” и выбрать “ Linux 32/64bit (shell archive)”. После чего возможно два варианта: либо будет окно с предложением загрузки скрипта, либо скрипт будет отображен в браузере. В последнем случае его нужно скопировать в файл с расширением \*.sh, либо нажать на “Linux 32/64bit (shell archive)” правой кнопкой мыши и в контекстном меню выбрать “Сохранить ссылку как..”.

После загрузки следует сначала сделать скрипт исполняемым: chmod +x ZeroBraneStudioEduPack-1.90-linux.sh, а затем запустить скрипт на выполнение: ./ ZeroBraneStudioEduPack-1.90-linux.sh (рис. 14)

После чего можно запускать IDE командой zbstudio.

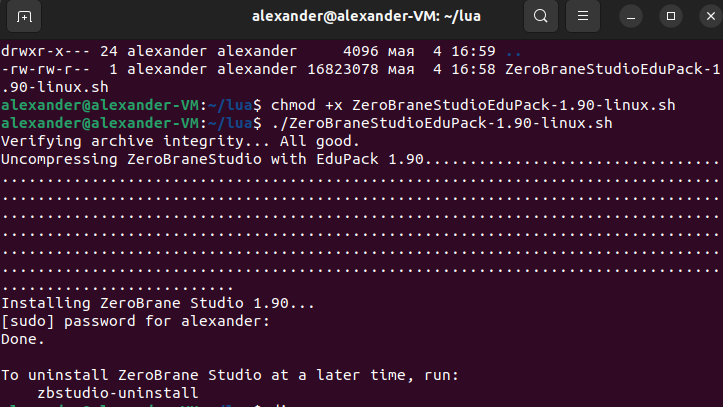


Рис. 14. Запуск скрипта для ZeroBrane Studio

* + - * **Установка и настройка Visual Studio Code**

Полная инструкция по установке Visual Studio Code под различные дистрибутивы Linux находится здесь: <https://code.visualstudio.com/docs/setup/linux>

После загрузки deb - пакета c Visual Studio Code (доступно по ссылке “ .deb package (64-bit)”, находящейся в разделе “Debian and Ubuntu based distributions” сайта, указанного выше) следует выполнить команду (рис. 15):

sudo apt-get install ./code\_1.66.2-1649664567\_amd64.deb

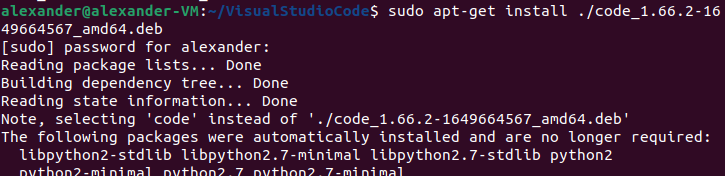


Рис. 15. Запуск пакета с дистрибутивом Visual Studio Code

Для запуска Visual Studio Code следует набрать *code* в терминале или найти приложение во всех доступных.

1. *Запуск внутри терминала.*

Запуск скрипта из командной строки осуществляется аналогично “*Запуск скрипта внутри PowerShell”* в разделе “Установка и настройка Visual Studio Code” для ОС Windows 10. На рис. 16 показан результат запуска скрипта.

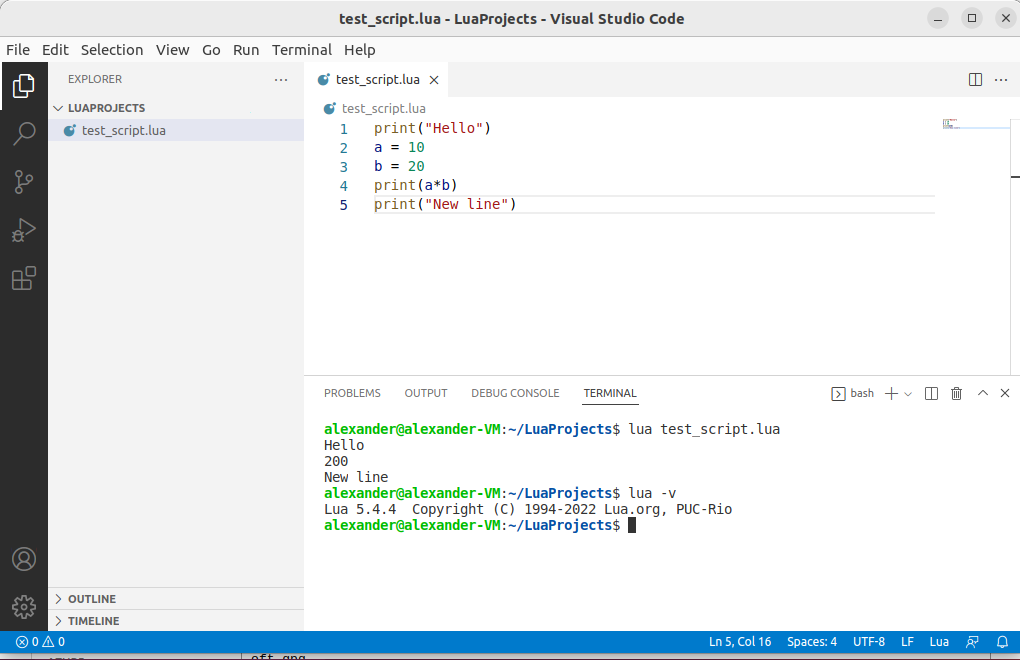


Рис. 16. Запуск скрипта

1. *Запуск в режиме отладки.*

Запуск в режиме отладки осуществляется аналогично “*Запуск скрипта в режиме отладки*” в разделе “Установка и настройка Visual Studio Code” для ОС Windows 10.

Примечание: путь к установленному пакету lua (требуется в файле “launch.json”) можно узнать так: *which lua*

1. *Запуск скрипта как задачи.*

Запуск скрипта как задачи осуществляется аналогично “ *Запуск скрипта как задачи*” в разделе “Установка и настройка Visual Studio Code” для ОС Windows 10.