

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

дисциплина: «Компьютерный практикум по статистическому
анализу данных»

Работу выполнил:

Снимщиков Иван Игоревич

Группа: НПИбд-02-21

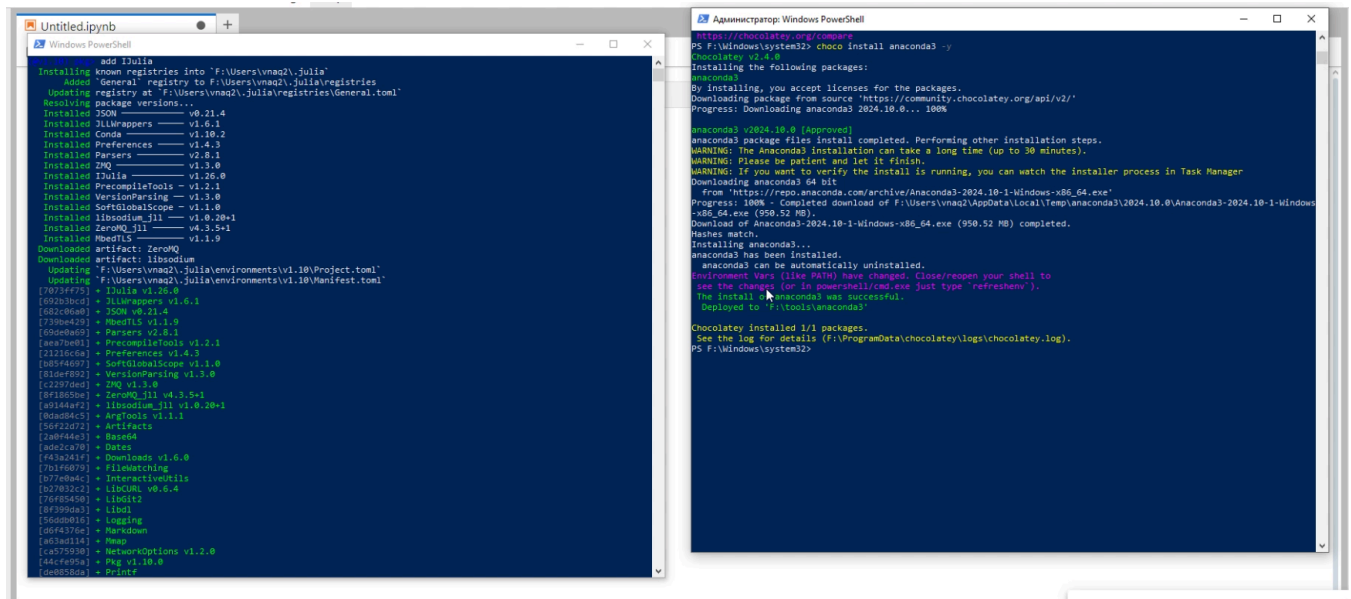
МОСКВА

2024 г.

Цели работы: Основная цель работы — подготовить рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомиться с основами синтаксиса Julia.

Ход работы:

Первым делом я установил все нужные компоненты



```
add !Julia
Installing known registries into "F:\Users\vmag2\Julia"
Added "General" registry to F:\Users\vmag2\Julia\registries
Updating registry at F:\Users\vmag2\Julia\registries\General.toml
Resolving package versions...
Installed JSON v0.21.4
Installed LLVMC v1.4.1
Installed Conda v1.10.2
Installed Preferences v1.4.3
Installed Parsers v2.8.1
Installed ZMQ v1.3.0
Installed IJulia v1.26.0
Installed PrecompileTools v1.2.1
Installed VersionParsing v1.3.0
Installed SoftGlobalScope v1.1.0
Installed Libsodium_jll v1.0.20+1
Installed ZeroMQ_jll v4.3.5+1
Installed RteDLS v1.1.9
Downloaded artifact: Libsodium
Updating F:\Users\vmag2\Julia\environments\v1.10\Project.toml
Updating F:\Users\vmag2\Julia\environments\v1.10\Manifest.toml
[7073f775] + IJulia v1.26.0
[502b3bc0] + LLVMC v1.4.1
[682c06a0] + JSON v0.21.4
[739be429] + RteDLS v1.1.9
[50de0a0d] + Parsers v2.8.1
[aea7be01] + PrecompileTools v1.2.1
[21216c6a] + Preferences v1.4.3
[ba5f4697] + SoftGlobalScope v1.1.0
[81def892] + VersionParsing v1.3.0
[c2297d6d] + ZMQ v1.3.0
[af18520a] + ZeroMQ_jll v4.3.5+1
[a9144af2] + Libsodium_jll v1.0.20+1
[6a0d04c5] + ArgTools v1.1.1
[56f22d72] + Artifacts
[2a0f44e3] + Base64
[8a2c2e79] + Dates
[f43a2d1f] + Downloads v1.6.0
[7b1f6079] + FileWatching
[977e0e4c] + InteractiveUtils
[b2783c22] + LibCURL v0.6.4
[76f85450] + LibGit2
[8f999e9d] + Libs
[56ddb016] + Logging
[d6f4376e] + Markdown
[a63a0114] + Mmap
[ca575930] + NetworkOptions v1.2.0
[44c4995a] + Pkg v1.10.0
[ca08350a] + Printf
```

```
PS F:\Windows\system32> choco install anaconda3 --y
Chocolatey v1.4.4
Installing the following packages:
anaconda3
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading anaconda3 2024.10.0... 100%
anaconda3 v2024.10.0 [Approved]
anaconda3 package files install completed. Performing other installation steps.
WARNING: The Anaconda3 installation can take a long time (up to 30 minutes).
WARNING: Please be patient and let it finish.
WARNING: If you want to verify the install is running, you can watch the installer process in Task Manager
Downloading anaconda3 64 bit
from 'https://repo.anaconda.com/archive/Anaconda3-2024.10-1-Windows-x86_64.exe'
Progress: 100% - Completed download of F:\Users\vmag2\AppData\Local\Temp\Anaconda3\2024.10.0\Anaconda3-2024.10-1-Windows-x86_64.exe (950.52 MB).
Download of Anaconda3-2024.10-1-Windows-x86_64.exe (950.52 MB) completed.
Hashes match.
Installing anaconda3...
anaconda3 has been installed.
anaconda3 can be automatically uninstalled.
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type 'refreshenv').
The install of anaconda3 was successful.
Deployed to 'F:\tools\anaconda3'
Chocolatey installed 1/1 packages.
See the log for details (F:\ProgramData\chocolatey\logs\chocolatey.log).
PS F:\Windows\system32>
```

Далее я протестировал синтаксис Julia как в примерах для лабораторной работы

```
Untitled.ipynb
[11]: 2+3
[11]: 5
[12]: 4+5
[12]: 9
[13]: 1+99
[13]: 100

[14]: ?println
search: println printstyled print sprint isprint

[14]: println([io::IO], xs...)
Print (using print) xs to io followed by a newline. If io is not supplied, prints to the default output stream stdout.

See also printstyled to add colors etc.

Examples

julia> println("Hello, world")
Hello, world

julia> io = IOBuffer();

julia> println(io, "Hello", ',', " world.")

julia> String(take!(io))
"Hello, world.\n"

[ ]:

[15]: ;whoami
desktop-10071a4\vnaq2

[ ]:
```

```
[16]: for T in
      [Int8,Int16,Int32,Int64,Int128,UInt8,UInt16,UInt32,UInt64,UInt128]
      println("${lpad(T,7)}: [$(typemin(T)),$(typemax(T))]" )
      end

      Int8: [-128,127]
      Int16: [-32768,32767]
      Int32: [-2147483648,2147483647]
      Int64: [-9223372036854775808,9223372036854775807]
      Int128: [-170141183460469231731687303715884105728,170141183460469231731687303715884105727]
      UInt8: [0,255]
      UInt16: [0,65535]
      UInt32: [0,4294967295]
      UInt64: [0,18446744073709551615]
      UInt128: [0,340282366920938463374607431768211455]
```

[]: |

```
[18]: typeof("1")
```

```
[18]: String
```

```
[19]: typeof(3)
```

```
[19]: Int64
```

```
[20]: typeof(false)
```

```
[20]: Bool
```

[]: |

```
[21]: function f(x)
      x^2
      end
      f(4)
```

```
[21]: 16
```

```
[22]: g(x)=x^2
```

```
[22]: g (generic function with 1 method)
```

```
[23]: a = [4 7 6]
      b = [1, 2, 3]
      a[2], b[2]
```

```
[23]: (7, 2)
```

```
[ ]:
```

Далее я приступил к пункту 1.3.4 для самостоятельной работы:

Read()

```
: file = open("1.txt", "r")
  content = read(file, String)
  close(file)
  println("Содержимое: $content")
```

Readline()

```
file = open("1.txt", "r")
line = readline(file)
close(file)
println("Первая строка: $line")
```

Readlines()

```
[ ]: lines = readlines("1.txt")
      println("Массив: ", lines)
```

Readdlm()

```
[ ]: data = readdlm("data.csv", ',')
```

Print() и Println()

```
[27]: print("Привет")  
      println(", Ваня")
```

Привет, Ваня

```
[ ]:
```

Write()

```
[ ]: write("1_out.txt", "privet")
```

Пример работы с parse()

```
: num = parse(Int, "123")  
  println("Число: $num")
```

Число: 123

Математические действия

```
[29]: x = 10  
      y = 20  
      println(x*y)
```

200

Работа с матрицами и скалярное произведение:

```
[31]: A = [1 2; 3 4]
      B = [4 3; 2 1]
      println("Сложение матриц: ", A + B)
```

Сложение матриц: [5 5; 5 5]

```
[32]: A = [1 2; 3 4]
      println("Транспонирование: ", A')
```

Транспонирование: [1 3; 2 4]

```
[33]: v1 = [1, 2, 3]
      v2 = [4, 5, 6]
      println("Скалярное произведение: ", dot(v1, v2))
```

```
[35]: A = [1 2; 3 4]
      B = [5 6; 7 8]
      println("Умножение: ", A * B)
```

Умножение: [19 22; 43 50]

Вывод: Я подготовил рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомился с основами синтаксиса Julia.