Haskell – Lab 1

Prepared by Dr. Wooi Ping Cheah

UNNC Staff Portal | Downloads

https://www.haskell.org/downloads/

(1) Go to this website to download Glasgow Haskell Compiler installer

Haskell

Get started | Downloads | Playground | Community | Documentation | Donate

> Psssst!
>
> Looking to get started with Haskell? If so, check out the Get Started page!

Downloads

Recommended installation instructions

for Linux, macOS, FreeBSD, Windows or WSL2

- Use GHCup to install GHC, cabal-install, Stack and haskell-language-server

(2) Click here to follow the link

Find out more about the Haskell toolchain

The Haskell toolchain consists of the following tools:

- GHC: the Glasgow Haskell Compiler
- cabal-install: the Cabal installation tool for managing Haskell software
- Stack: a cross-platform program for developing Haskell projects
- haskell-language-server (optional): A language server for developers to integrate with their editor/IDE

Installation via native OS package manager

Alternatively, many operating systems provide GHC, cabal and Stack through their native package manager

UNNC Staff Portal | GHCup

https://www.haskell.org/ghcup/

GHCup

GHCup

GHCup is the main installer for the general purpose language Haskell.

Installation | First steps | User Guide

(1) Click and drag to select the whole line of text (command)

To install on Windows

run the following in a PowerShell session (as a non-admin user):

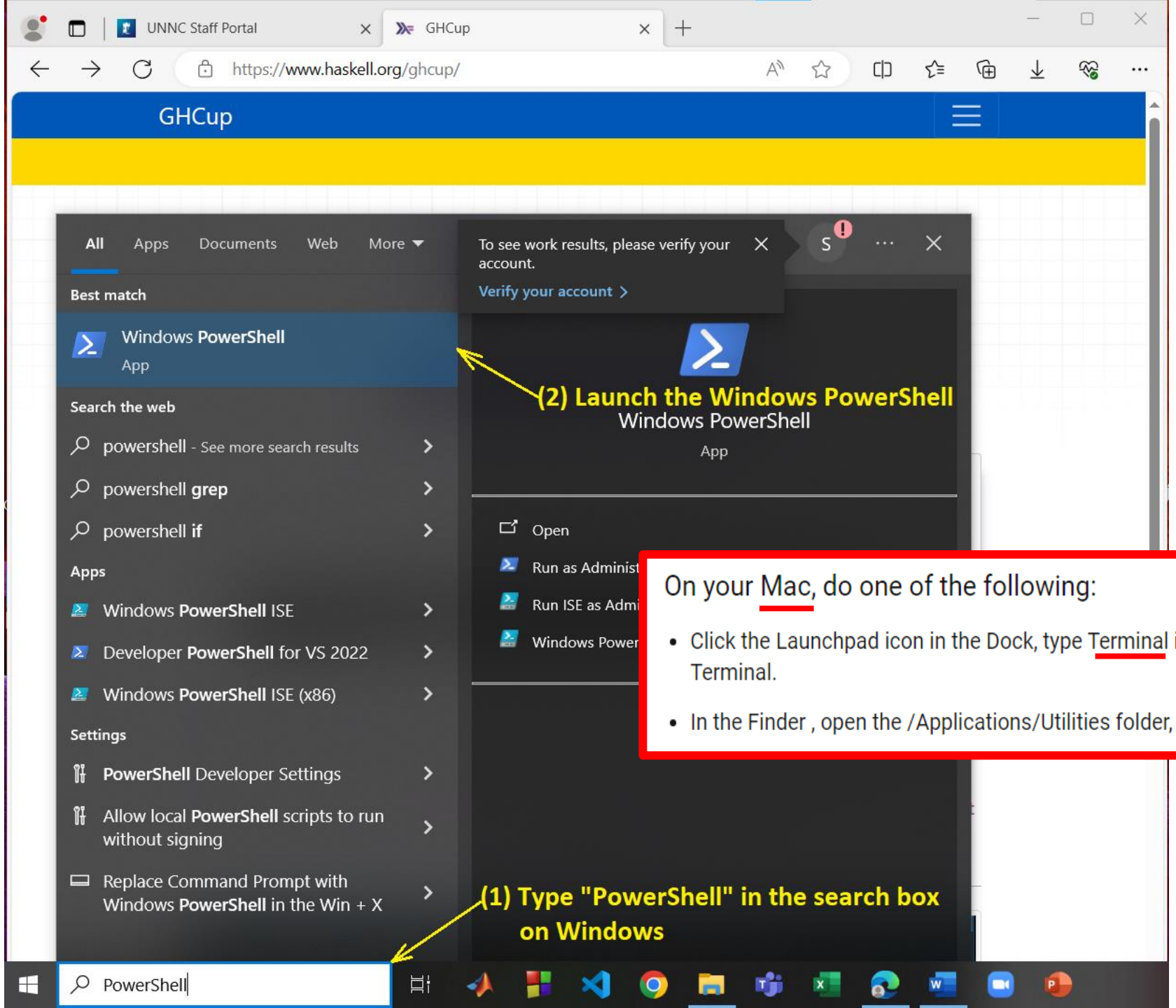
```
$ Set-ExecutionPolicy Bypass -Scope Process -Force;[Syster
```

What does this do? · I don't like curl | sh · Show all platforms

(2) Click here to copy the selected command, which will be pasted to PowerShell later

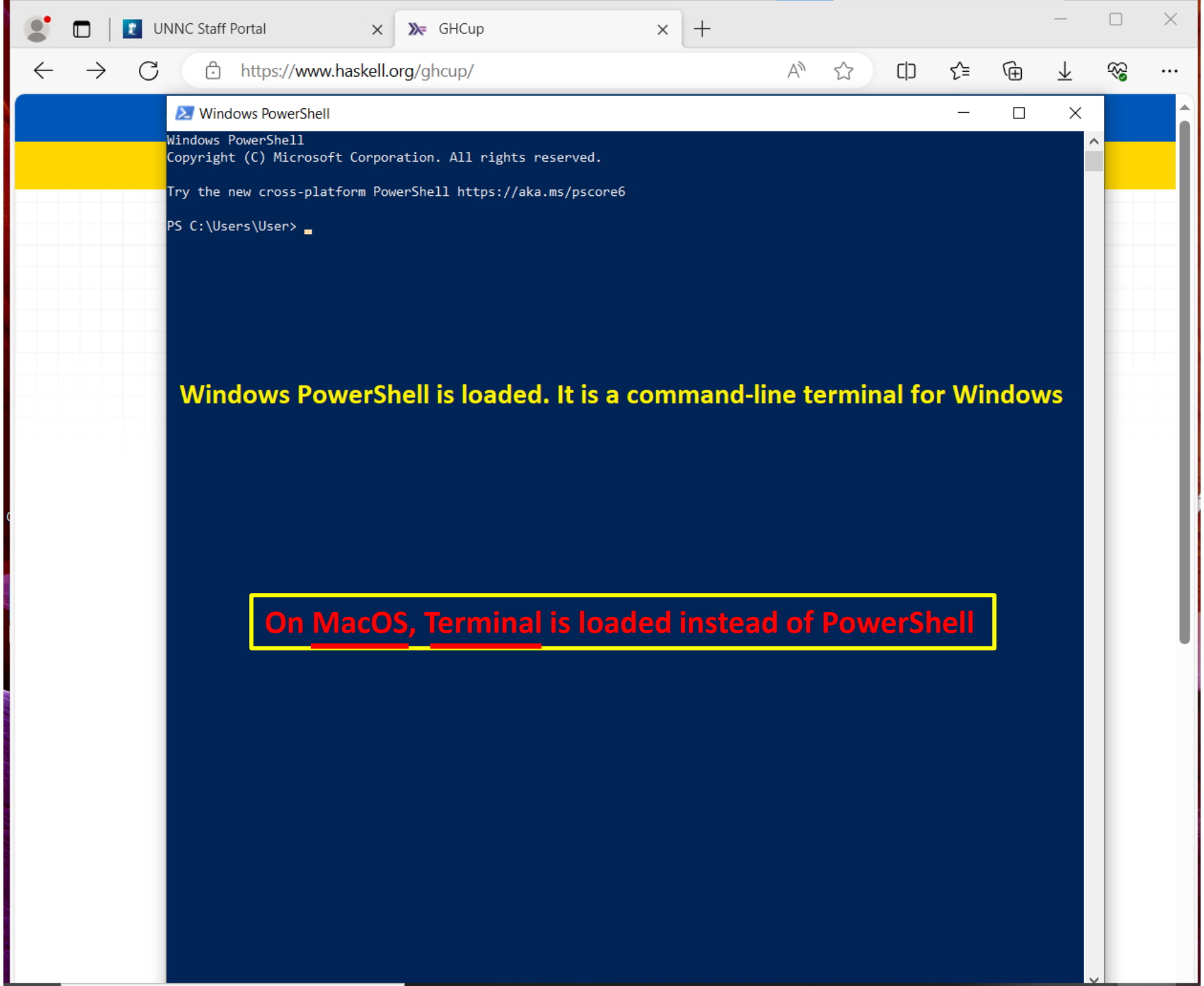
For Linux, macOS, FreeBSD or Windows Subsystem 2 for Linux, run this in a terminal:

```
curl --proto '=https' --tlsv1.2 -sSf https://get-ghcup.haskell.org | sh
```



On your Mac, do one of the following:

- Click the Launchpad icon in the Dock, type Terminal in the search field, then click Terminal.
- In the Finder , open the /Applications/Utilities folder, then double-click Terminal.


UNNC Staff Portal
GHCup
https://www.haskell.org/ghcup/
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\User>
Windows PowerShell is loaded. It is a command-line terminal for Windows
On MacOS, Terminal is loaded instead of PowerShell

Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User>  Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProto
col -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -A
rgumentList $true } catch { Write-Error $_ }
```

Press CTRL V to paste the command for installing Glasgow Haskell Compiler, and press ENTER to start the installation process

On MacOS, paste the following command line:
curl --proto '=https' --tlsv1.2 -sSf https://get-ghcup.haskell.org | sh

Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User>  Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProto
col -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -A
rgumentList $true } catch { Write-Error $_ }
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:
```

Press Enter key to accept the default installation path for GHCup binaries

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:
```

Press Enter key to accept the default installation path for the Cabal build tool

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:


Install HLS
Do you want to install the haskell-language-server (HLS) for development purposes as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y
```

Type Y and press Enter key to install the Haskell Language Server

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:


Install HLS
Do you want to install the haskell-language-server (HLS) for development purposes as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Install stack
Do you want to install stack as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y
```

Type Y and press Enter key to install Stack, a cross-platform development tool

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:


Install HLS
Do you want to install the haskell-language-server (HLS) for development purposes as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Install stack
Do you want to install stack as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Create Desktop shortcuts
Do you want to create convenience desktop shortcuts (e.g. for uninstallation and msys2 shell)?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "Y"):
```

Press Enter key to accept default creation of Desktop shortcuts

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:


Install HLS
Do you want to install the haskell-language-server (HLS) for development purposes as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Install stack
Do you want to install stack as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Create Desktop shortcuts
Do you want to create convenience desktop shortcuts (e.g. for uninstallation and msys2 shell)?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "Y"):
First checking for Msys2...

Install MSys2
Do you want GHCup to install a default MSys2 toolchain (recommended)?
[Y] Yes  [N] No  [?] Help (default is "Y"):
```

Press Enter key to accept default installation of MSys2 toolchain

Windows PowerShell

```
PS C:\Users\User> Set-ExecutionPolicy Bypass -Scope Process -Force;[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; try { Invoke-Command -ScriptBlock ([ScriptBlock]::Create((Invoke-WebRequest https://www.haskell.org/ghcup/sh/bootstrap-haskell.ps1 -UseBasicParsing))) -ArgumentList $true } catch { Write-Error $_ }
Picked C:\ as default Install prefix!
Welcome to Haskell!

This script can download and install the following programs:
  * ghcup - The Haskell toolchain installer
  * ghc   - The Glasgow Haskell Compiler
  * msys2 - A linux-style toolchain environment required for many operations
  * cabal - The Cabal build tool for managing Haskell software
  * stack - (optional) A cross-platform program for developing Haskell projects
  * hls   - (optional) A language server for developers to integrate with their editor/IDE

Please note that ANTIVIRUS may interfere with the installation. If you experience problems, consider
disabling it temporarily.

Where to install to (this should be a short Path, preferably a Drive like 'C:\')?
If you accept this path, binaries will be installed into 'C:\ghcup\bin' and msys2 into 'C:\ghcup\msys64'.
Press enter to accept the default [C:\]:


Setting env variable GHCUP_INSTALL_BASE_PREFIX to 'C:\'
Preparing for GHCup installation...
Specify Cabal directory (this is where haskell packages end up)
Press enter to accept the default [C:\\cabal]:


Install HLS
Do you want to install the haskell-language-server (HLS) for development purposes as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Install stack
Do you want to install stack as well?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "N"): Y

Create Desktop shortcuts
Do you want to create convenience desktop shortcuts (e.g. for uninstallation and msys2 shell)?
[Y] Yes  [N] No  [A] Abort  [?] Help (default is "Y"):
First checking for Msys2...

Install MSys2
Do you want GHCup to install a default MSys2 toolchain (recommended)?
[Y] Yes  [N] No  [?] Help (default is "Y"):
...Msys2 doesn't exist, installing into C:\\ghcup\msys64
Starting installation in 5 seconds, this may take a while...
Downloading Msys2 archive 20221216...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 57.0M  100 57.0M    0     0  17.9M      0  0:00:03  0:00:03 --:--:-- 17.9M
Extracting Msys2 archive...
```

Please wait for the installation process to complete

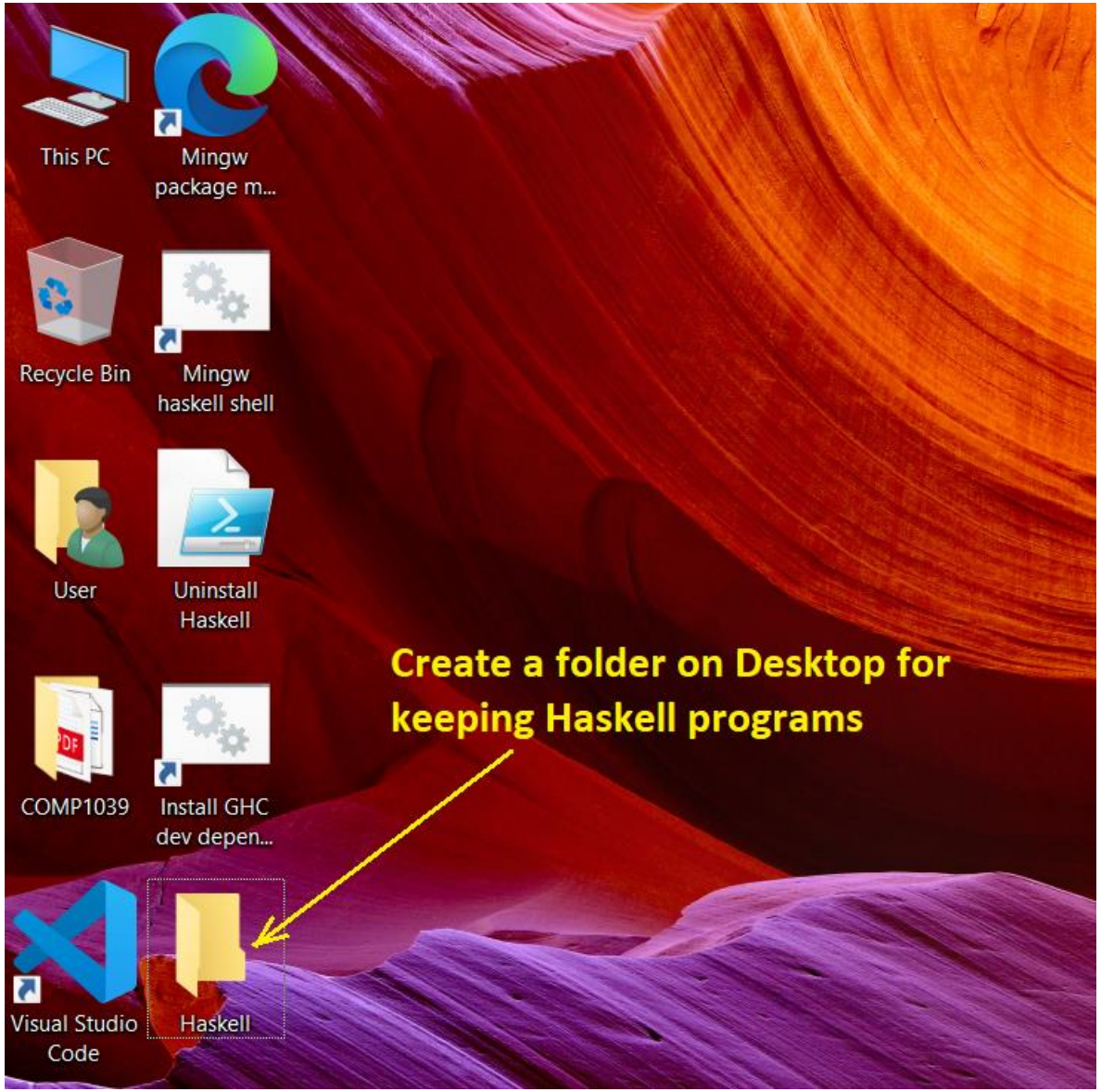
Windows PowerShell

```
Total Download Size:   2.13 MiB
Total Installed Size:  9.01 MiB

:: Proceed with installation? [Y/n]
:: Retrieving packages...
 m4-1.4.19-2-x86_64                          238.1 KiB   277 KiB/s 00:01 [#######################################] 100%
 autoconf2.69-2.69-4-any                     284.6 KiB   294 KiB/s 00:01 [#######################################] 100%
 autoconf2.71-2.71-3-any                     328.5 KiB   273 KiB/s 00:01 [#######################################] 100%
 mingw-w64-x86_64-pkgconf-1~2.1.0-1-any       84.3 KiB   317 KiB/s 00:00 [#######################################] 100%
 autoconf-wrapper-20221207-2-any               5.0 KiB  13.8 KiB/s 00:00 [#######################################] 100%
 diffutils-3.10-1-x86_64                     379.8 KiB   227 KiB/s 00:02 [#######################################] 100%
 autoconf2.13-2.13-6-any                     137.4 KiB   195 KiB/s 00:01 [#######################################] 100%
 autoconf2.72-2.72-1-any                     726.6 KiB   412 KiB/s 00:02 [#######
 Total (8/8)                                   2.1 MiB  1205 KiB/s 00:02 [#######
(8/8) checking keys in keyring                                           [#######
(8/8) checking package integrity                                         [#######
(8/8) loading package files                                              [#######
(8/8) checking for file conflicts                                        [#######
(8/8) checking available disk space                                      [#######
:: Processing package changes...
(1/8) installing m4                                                      [#######
(2/8) installing diffutils                                               [#######
(3/8) installing autoconf2.72                                            [#######
(4/8) installing autoconf2.71                                            [#######
(5/8) installing autoconf2.69                                            [#######
(6/8) installing autoconf2.13                                            [#######
(7/8) installing autoconf-wrapper                                        [#######
(8/8) installing mingw-w64-x86_64-pkgconf                                [#######
:: Running post-transaction hooks...
(1/1) Updating the info directory file...
Updating SSL root certificate authorities...
warning: ca-certificates-20230311-1 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Packages (1) ca-certificates-20230311-1

Total Installed Size:  0.95 MiB
Net Upgrade Size:      0.00 MiB

:: Proceed with installation? [Y/n]
(1/1) checking keys in keyring                                           [#######################################] 100%
(1/1) checking package integrity                                         [#######################################] 100%
(1/1) loading package files                                              [#######################################] 100%
(1/1) checking for file conflicts                                        [#######################################] 100%
(1/1) checking available disk space                                      [#######################################] 100%
:: Processing package changes...
(1/1) reinstalling ca-certificates                                       [#######################################] 100%
Setting default home directory...
Creating shortcuts...
Adding C:\\ghcup\bin to Users Path...
Setting CABAL_DIR to 'C:\cabal'
Starting GHCup installer...
PS C:\Users\User>
```

MinGW x64

```
In order to run ghc and cabal, you need to adjust your PATH variable.
To do so, you may want to run 'source /c/ghcup/env' in your current terminal
session as well as your shell configuration (e.g. ~/.bashrc).

================================================================================

All done!

In a new powershell or cmd.exe session, now you can...

Start a simple repl via:
  ghci

Start a new haskell project in the current directory via:
  cabal init --interactive

To install other GHC versions and tools, run:
  ghcup tui

To install system libraries and update msys2/mingw64,
open the "Mingw haskell shell"
and the "Mingw package management docs"
desktop shortcuts.

If you are new to Haskell, check out https://www.haskell.org/ghcup/steps/
Press any key to exit
```

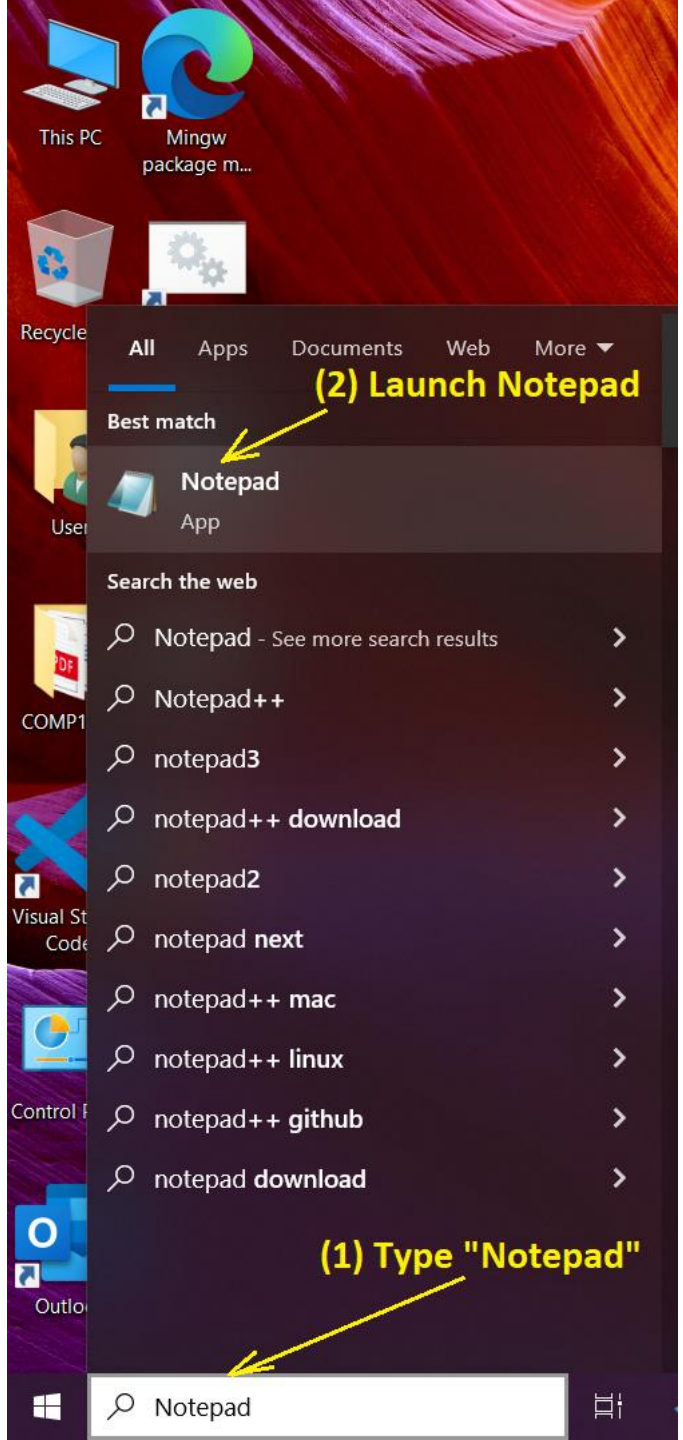
Installation process is completed when you see the PowerShell command prompt and the "Press any key to exit" message on MinGW


This PC
Mingw package m...
Recycle Bin
Mingw haskell shell
User
Uninstall Haskell
COMP1039
Install GHC dev depen...
Visual Studio Code
Haskell
Create a folder on Desktop for keeping Haskell programs

On MacOS, you may use TextEdit instead of Notepad

You can find TextEdit on your Mac by doing one of the following

- Press the F4 key to open Spotlight Search, then type TextEdit and hit enter.
- Use Launch Pad by pinching with your thumb and 3 fingers to open it, then find the TextEdit App icon.
- Use the Finder App to navigate to the Applications folder, then double-click TextEdit to open the app.
- Invoke Spotlight by pressing cmd + space, then search and open TextEdit.
- Starting macOS Catalina, TextEdit is located in /System/Applications/TextEdit.app/Contents/MacOS/TextEdit



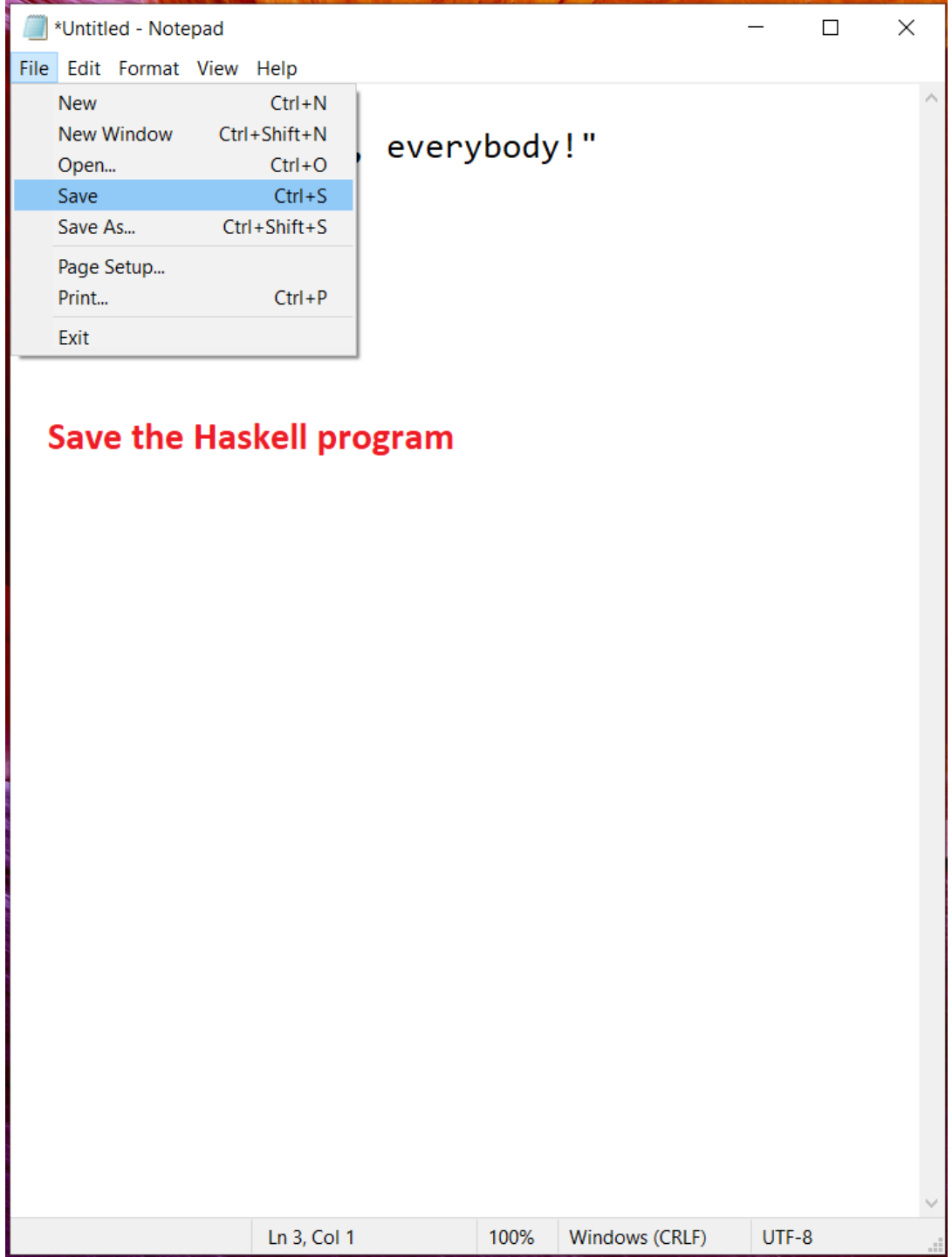
*Untitled - Notepad

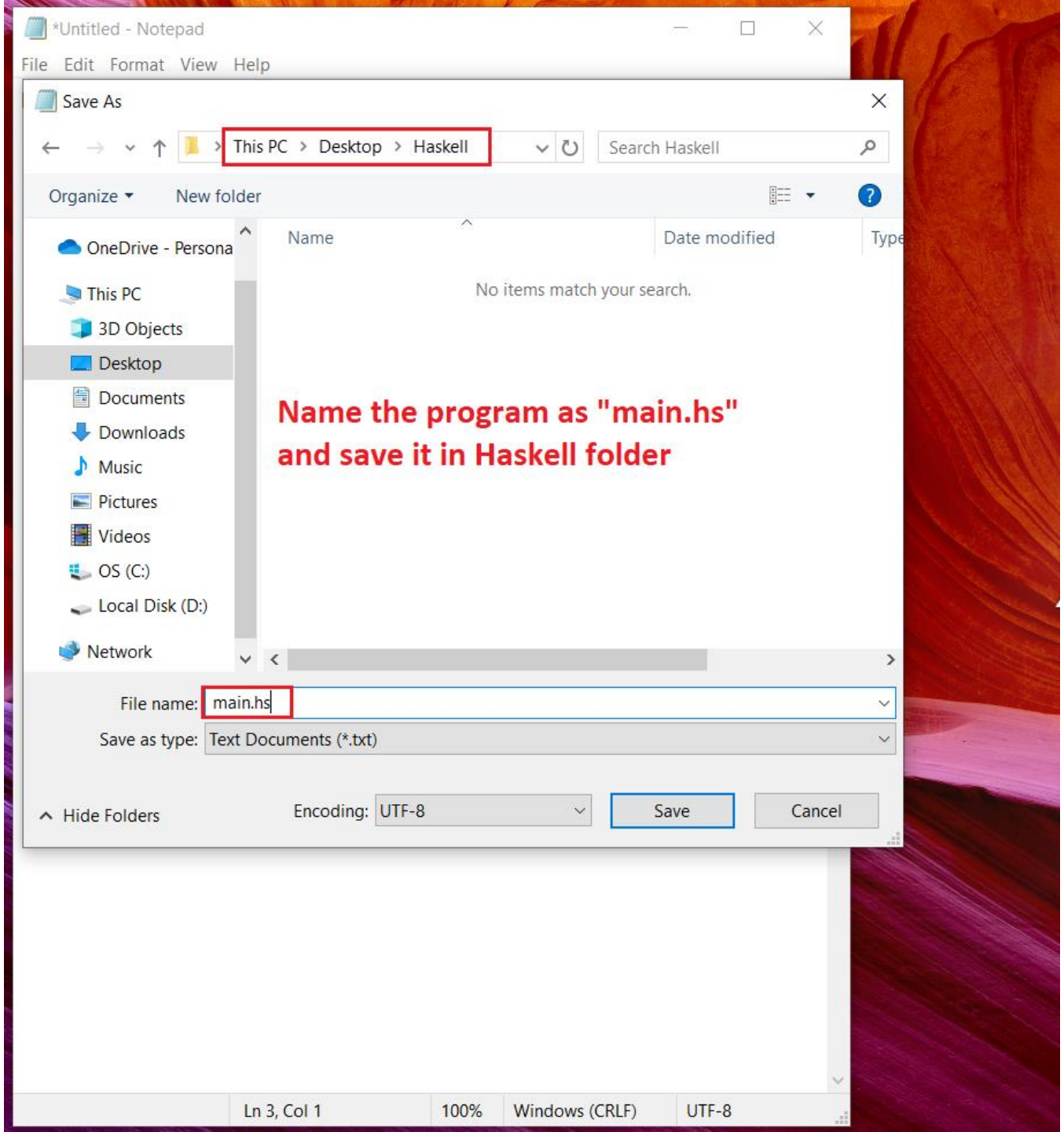
File Edit Format View Help

```
main = do
  putStrLn "Hello, everybody!"
```

Type a simple Haskell program with Notepad editor

Ln 3, Col 1 | 100% | Windows (CRLF) | UTF-8


*Untitled - Notepad
File Edit Format View Help
New Ctrl+N
New Window Ctrl+Shift+N
Open... Ctrl+O
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup...
Print... Ctrl+P
Exit
everybody!"
Save the Haskell program
Ln 3, Col 1
100%
Windows (CRLF)
UTF-8


*Untitled - Notepad
File Edit Format View Help
Save As
This PC > Desktop > Haskell
Search Haskell
Organize
New folder
OneDrive - Persona
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
OS (C:)
Local Disk (D:)
Network
Name
Date modified
Type
No items match your search.
Name the program as "main.hs"
and save it in Haskell folder
File name: main.hs
Save as type: Text Documents (*.txt)
Hide Folders
Encoding: UTF-8
Save
Cancel
Ln 3, Col 1
100%
Windows (CRLF)
UTF-8

Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User> cd Desktop\Haskell
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   2:45 PM             43 main.hs


PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe
PS C:\Users\User\Desktop\Haskell> ./main
Hello, everybody!
PS C:\Users\User\Desktop\Haskell> runghc main.hs
Hello, everybody!
PS C:\Users\User\Desktop\Haskell>
```

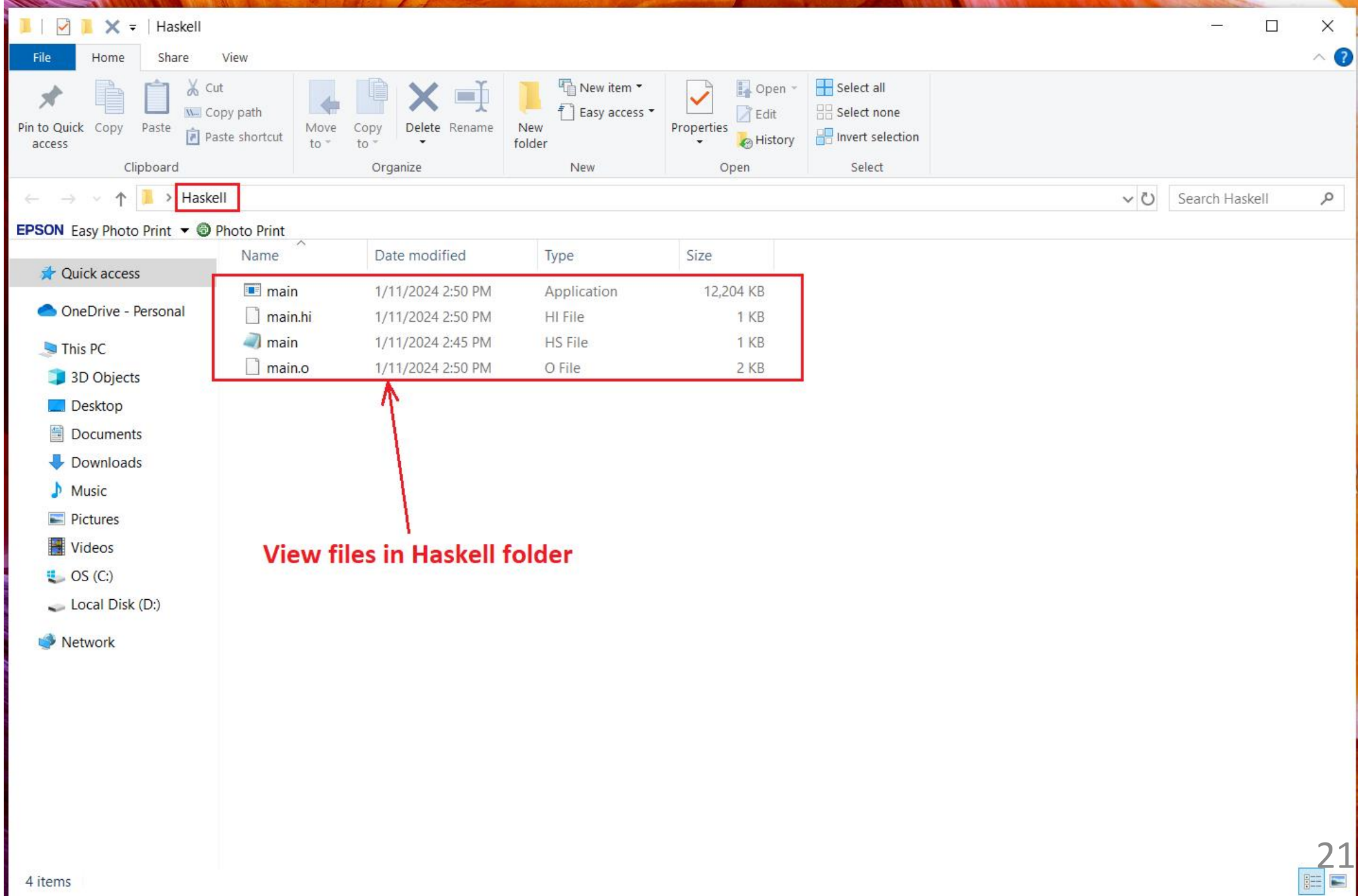
Change directory to Desktop\Haskell

List the content of Desktop\Haskell directory
There is only one Haskell program file "main.hs"

Compile the Haskell program "main.hs"
Two files generated: "main.o" and "main.exe"

Run the Haskell program to display a welcome message

Run the Haskell program directly without compilation


Haskell
File
Home
Share
View
Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut
Clipboard
Move to
Copy to
Delete
Rename
Organize
New folder
New item
Easy access
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
Haskell
Search Haskell
EPSON Easy Photo Print
Photo Print
Quick access
OneDrive - Personal
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
OS (C:)
Local Disk (D:)
Network
Name
Date modified
Type
Size
main 1/11/2024 2:50 PM Application 12,204 KB
main.hi 1/11/2024 2:50 PM HI File 1 KB
main 1/11/2024 2:45 PM HS File 1 KB
main.o 1/11/2024 2:50 PM O File 2 KB
View files in Haskell folder
4 items

Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User> cd Desktop\Haskell
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   2:45 PM             43 main.hs


PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe
PS C:\Users\User\Desktop\Haskell> ./main
Hello, everybody!
PS C:\Users\User\Desktop\Haskell> runghc main.hs
Hello, everybody!
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )

main.hs:1:1: warning: [-Wmissing-signatures]
    Top-level binding with no type signature: main :: IO ()
  |
1 | main = do
  | ^^^^
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
```

Re-compile the program by turning on all warning options

This is the "no type signature" warning message

main - Notepad

File Edit Format View Help

```
main::IO()
main = do
  putStrLn "Hello, everybody!"
```

Add this line of code that will eliminate the warning

Save the program after editing

Ln 1, Col 11 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe
PS C:\Users\User\Desktop\Haskell> ./main
Hello, everybody!
PS C:\Users\User\Desktop\Haskell> runghc main.hs
Hello, everybody!
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )

main.hs:1:1: warning: [-Wmissing-signatures]
    Top-level binding with no type signature: main :: IO ()
  |
1 | main = do
  | ^^^^
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
```

Re-compile the corrected program with all warning options turned on

No more warning message this time

main - Notepad

File Edit Format View Help

```
main::IO()
main = do
  type "Hello, everybody!"
```

Purposely introduce an error into the program code by changing "putStrLn" to "type"

Save the program

Ln 3, Col 7 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe
PS C:\Users\User\Desktop\Haskell> ./main
Hello, everybody!
PS C:\Users\User\Desktop\Haskell> runghc main.hs
Hello, everybody!
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )

main.hs:1:1: warning: [-Wmissing-signatures]
    Top-level binding with no type signature: main :: IO ()
  |
1 | main = do
  | ^^^^
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o ) [Source file changed]

main.hs:3:3: error: parse error on input `type'
  |
3 |   type "Hello, everybody!"
  |   ^^^^
PS C:\Users\User\Desktop\Haskell>
```

Compile the program again

There is an error at line 3

main - Notepad

File Edit Format View Help

```
main::IO()
main = do
  putStrLn "Hello, everybody!"
```

Revert the program code to remove the error by changing "type" to "putStrLn"

Save the program

Ln 3, Col 11 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe
PS C:\Users\User\Desktop\Haskell> ./main
Hello, everybody!
PS C:\Users\User\Desktop\Haskell> runghc main.hs
Hello, everybody!
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )

main.hs:1:1: warning: [-Wmissing-signatures]
    Top-level binding with no type signature: main :: IO ()
  |
1 | main = do
  | ^^^^
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc -Wall main.hs -fforce-recomp
[1 of 2] Compiling Main             ( main.hs, main.o )
[2 of 2] Linking main.exe [Objects changed]
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell>
PS C:\Users\User\Desktop\Haskell> ghc main.hs
[1 of 2] Compiling Main             ( main.hs, main.o ) [Source file changed]

main.hs:3:3: error: parse error on input `type'
  |
3 |   type "Hello, everybody!"
  |   ^^^^
PS C:\Users\User\Desktop\Haskell> clear
```

Clear screen

Select Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls

    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci>
```

List files in the directory

Start the interactive GHCi mode

Interactive GHCi mode is the preferred way of running Haskell programs

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci>
```

Applying built-in functions discussed in the lecture

Make sure you understand these functions

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
```

Edit a new Haskell source file "test.hs"

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
```

Untitled - Notepad

File Edit Format View Help

Notepad

Cannot find the test.hs file.

Do you want to create a new file?

Yes No Cancel

Click "Yes" to create a new file

Ln 1, Col 1 100% Windows (CRLF) UTF-8

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
```

test - Notepad

File Edit Format View Help

(2) Save the file

(3) Close the file

```
double x = x + x

quadruple x = double (double x)
```

(1) Type the above two functions

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci>
```

Return to GHCi mode

Windows PowerShell

```
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci>
```

Apply the functions just defined

Windows PowerShell

```
    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o


PS C:\Users\User\Desktop\Haskell> ghci
GHCi, version 9.4.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3*4
14
ghci> (2+3)*4
20
ghci> sqrt (3^2 + 4^2)
5.0
ghci> head [1,2,3,4,5]
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci> :edit test.hs
```

test - Notepad

File Edit Format View Help

(2) Save the file

(3) Close the file

```
double x = x + x

quadruple x = double (double x)

factorial n = product [1..n]

average ns = sum ns `div` length ns
```

(1) Add two functions "factorial" and "average"

Ln 7, Col 36 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file changed]
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> factorial 5
120
ghci> factorial 10
3628800
ghci> average [1,2,3,4,5]
3
ghci>
```

Reload the source file "test.hs"

Apply the newly added functions

Windows PowerShell

```
1
ghci> tail [1,2,3,4,5]
[2,3,4,5]
ghci> [1,2,3,4,5] !! 2
3
ghci> take 3 [1,2,3,4,5]
[1,2,3]
ghci> drop 3 [1,2,3,4,5]
[4,5]
ghci> length [1,2,3,4,5]
5
ghci> sum [1,2,3,4,5]
15
ghci> product [1,2,3,4,5]
120
ghci> [1,2,3] ++ [4,5]
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file c
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> factorial 5
120
ghci> factorial 10
3628800
ghci> average [1,2,3,4,5]
3
ghci> :edit test.hs
```

test - Notepad

File Edit Format View Help

(2) Save the file

(3) Close the file

```
double x = x + x

quadruple x = double (double x)

factorial n = product [1..n]

average ns = sum ns `div` length ns

a = b + c
 where
 b = 1
 c = 2

d = a * 2
```

(1) Add two more functions "a" and "b", without specifying any input parameter

Ln 14, Col 10 | 100% | Windows (CRLF) | UTF-8

Windows PowerShell

```
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file changed]
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> factorial 5
120
ghci> factorial 10
3628800
ghci> average [1,2,3,4,5]
3
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file changed]
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> a
3
ghci> d
6
ghci>
```

Reload the source file "test.hs"

Apply the two newly added functions

Windows PowerShell

```
[1,2,3,4,5]
ghci> reverse [1,2,3,4,5]
[5,4,3,2,1]
ghci> :edit test.hs
Ok, no modules loaded.
ghci> :load test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted )
Ok, one module loaded.
ghci> double 10
20
ghci> quadruple 10
40
ghci> take (double 2) [1,2,3,4,5,6]
[1,2,3,4]
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file changed]
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> factorial 5
120
ghci> factorial 10
3628800
ghci> average [1,2,3,4,5]
3
ghci> :edit test.hs
[1 of 2] Compiling Main             ( test.hs, interpreted ) [Source file changed]
Ok, one module loaded.
ghci> :reload
Ok, one module loaded.
ghci> a
3
ghci> d
6
ghci> :type +d length
length :: [a] -> Int
ghci> :type +d sum
sum :: [Integer] -> Integer
ghci> :type double
double :: Num a => a -> a
ghci> :type a
a :: Integer
ghci> :type d
d :: Integer
ghci>
```

Check the type of expressions

We will learn more about type of expressions in the next lecture.

Windows PowerShell

```
ghci> :type +d length
length :: [a] -> Int
ghci> :type +d sum
sum :: [Integer] -> Integer
ghci> :type double
double :: Num a => a -> a
ghci> :type a
a :: Integer
ghci> :type d
d :: Integer
ghci> :?
 Commands available from the prompt:

   <statement>                 evaluate/run <statement>
   :                           repeat last command
   :{\n ..lines.. \n:}\n       multiline command
   :add [*]<module> ...        add module(s) to the current target set
   :browse[!] [[*]<mod>]       display the names defined by module <mod>
                               (!: more details; *: all top-level names)
   :cd <dir>                   change directory to <dir>
   :cmd <expr>                 run the commands returned by <expr>::IO String
   :complete <dom> [<rng>] <s> list completions for partial input string
   :ctags[!] [<file>]          create tags file <file> for Vi (default: "tags")
                               (!: use regex instead of line number)
   :def[!] <cmd> <expr>        define command :<cmd> (later defined command has
                               precedence, ::<cmd> is always a builtin command)
                               (!: redefine an existing command name)
   :doc <name>                 display docs for the given name (experimental)
   :edit <file>                edit file
   :edit                       edit last module
   :etags [<file>]             create tags file <file> for Emacs (default: "TAGS")
   :help, :?                   display this list of commands
   :info[!] [<name> ...]       display information about the given names
                               (!: do not filter instances)
   :instances <type>           display the class instances available for <type>
   :issafe [<mod>]             display safe haskell information of module <mod>
   :kind[!] <type>             show the kind of <type>
                               (!: also print the normalised type)
   :load[!] [*]<module> ...    load module(s) and their dependents
                               (!: defer type errors)
   :main [<arguments> ...]     run the main function with the given arguments
   :module [+/-] [*]<mod> ...  set the context for expression evaluation
   :quit                       exit GHCi
   :reload[!]                  reload the current module set
                               (!: defer type errors)
   :run function [<arguments> ...] run the function with the given arguments
   :script <file>              run the script <file>
   :type <expr>                show the type of <expr>
   :type +d <expr>             show the type of <expr>, defaulting type variables
   :unadd <module> ...         remove module(s) from the current target set
   :undef <cmd>                undefine user-defined command :<cmd>
   ::<cmd>                     run the builtin command
   :!<command>                 run the shell command <command>
```

List all GHCi commands

Windows PowerShell

```
   +m            allow multiline commands
   +r            revert top-level expressions after each evaluation
   +s            print timing/memory stats after each evaluation
   +t            print type after evaluation
   +c            collect type/location info after loading modules
   -<flags>      most GHC command line flags can also be set here
                       (eg. -v2, -XFlexibleInstances, etc.)
                     for GHCi-specific flags, see User's Guide,
                     Flag reference, Interactive-mode options

 -- Commands for displaying information:

   :show bindings              show the current bindings made at the prompt
   :show breaks                show the active breakpoints
   :show context               show the breakpoint context
   :show imports               show the current imports
   :show linker                show current linker state
   :show modules               show the currently loaded modules
   :show packages              show the currently active package flags
   :show paths                 show the currently active search paths
   :show language              show the currently active language flags
   :show targets               show the current set of targets
   :show <setting>             show value of <setting>, which is one of
                                  [args, prog, editor, stop]
   :showi language             show language flags for interactive evaluation

 The User's Guide has more information. An online copy can be found here:

   https://downloads.haskell.org/~ghc/latest/docs/html/users_guide/ghci.html

ghci> :quit
Leaving GHCi.
PS C:\Users\User\Desktop\Haskell> ls


    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o
-a----         1/20/2024   7:38 PM            178 test.hs


PS C:\Users\User\Desktop\Haskell>
```

Quit the GHCi mode

Windows PowerShell

```
   +m            allow multiline commands
   +r            revert top-level expressions after each evaluation
   +s            print timing/memory stats after each evaluation
   +t            print type after evaluation
   +c            collect type/location info after loading modules
   -<flags>      most GHC command line flags can also be set here
                       (eg. -v2, -XFlexibleInstances, etc.)
                 for GHCi-specific flags, see User's Guide,
                 Flag reference, Interactive-mode options

 -- Commands for displaying information:

   :show bindings              show the current bindings made at the prompt
   :show breaks                show the active breakpoints
   :show context               show the breakpoint context
   :show imports               show the current imports
   :show linker                show current linker state
   :show modules               show the currently loaded modules
   :show packages              show the currently active package flags
   :show paths                 show the currently active search paths
   :show language              show the currently active language flags
   :show targets               show the current set of targets
   :show <setting>             show value of <setting>, which is one of
                                  [args, prog, editor, stop]
   :showi language             show language flags for interactive evaluation

 The User's Guide has more information. An online copy can be found here:

   https://downloads.haskell.org/~ghc/latest/docs/html/users_guide/ghci.html

ghci> :quit
Leaving GHCi.
PS C:\Users\User\Desktop\Haskell> ls
```

List all files in the current directory

```
    Directory: C:\Users\User\Desktop\Haskell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         1/11/2024   3:45 PM       12496384 main.exe
-a----         1/11/2024   3:45 PM            712 main.hi
-a----         1/20/2024   5:05 PM             55 main.hs
-a----         1/11/2024   3:45 PM           1835 main.o
-a----         1/20/2024   7:38 PM            178 test.hs


PS C:\Users\User\Desktop\Haskell> exit
```

Exit PowerShell

Exercises
from
Chapter 2 – First Steps

(2) Fix the syntax errors in the program below, and test your solution using GHCi.

```
N = a 'div' length xs
    where
        a = 10
       xs = [1,2,3,4,5]
```

Hint: Proper use of upper/lower case

(3) Show how the library function <u>last</u> that selects the last element of a list can be defined using the functions introduced in this lecture.

Hint: The last element of a list is the first element of the reverse list.

(4) Can you think of another possible definition?

Hint: The last element of a list is the nth element of the list, where n is the length of the list minus 1.

(5) Similarly, show how the library function <u>init</u> that removes the last element from a list can be defined in two different ways.

Hints: (1) It is defined as the sub-list of the original list with the first n elements, where n is the length of the original list minus 1.

(2) It is defined as the reverse of the tail of the reverse of the original list.

References
Haskell Installation

GHCup Home
https://www.haskell.org/ghcup/

GHCup Installation
https://www.haskell.org/ghcup/install/

Install Haskell Compiler & Cabal on Mac/Linux
https://www.youtube.com/watch?v=TLDl2t2dgwI

Haskell Installation Tutorial for Mac OS
https://www.youtube.com/watch?v=ve5DcE2RD0A

References
Functional Programming & Haskell

Functional Programming & Haskell - Computerphile

https://www.youtube.com/watch?v=LnX3B9oaKzw

An interview session with a member of the team that created Haskell: John Hughes, Professor of Computer Science at Chalmers University of Technology in Gothenburg.

Programming Paradigms - Computerphile

https://www.youtube.com/watch?v=sqV3pL5x8PI

References
Functional Programming & Haskell

What is functional programming | Easy way

https://www.youtube.com/watch?v=dAPL7MQGjyM

FP vs OOP | For Dummies

https://www.youtube.com/watch?v=08CWw_VD45w