Table of Contents

[The Bash Shell 1](#_Toc535770549)

[For Mac Users: Terminal 1](#_Toc535770550)

[For Windows Users: Git Bash 2](#_Toc535770551)

[Directory = Folder 2](#_Toc535770552)

[Overview of Bash Commands 2](#_Toc535770553)

[How to Open Your Bash Shell: 2](#_Toc535770554)

[Print Working Directory: pwd 2](#_Toc535770555)

[List Files and Directories: ls 2](#_Toc535770556)

[Make Directories: mkdir name\_of\_directory 3](#_Toc535770557)

[Change Directory: cd path\_of\_directory 3](#_Toc535770558)

[ Going Backwards 3](#_Toc535770559)

[ Going Home: 3](#_Toc535770560)

[ Going to Desktop 3](#_Toc535770561)

[ Going to Documents 3](#_Toc535770562)

[ Tab to Autocomplete 3](#_Toc535770563)

[ Going Forward Many Levels at Once 3](#_Toc535770564)

[Glossary 4](#_Toc535770565)

# The Bash Shell

We will be using the Bash Shell for this class. The Bash Shell is a Command Line Interface (CLI) used to interact with your computer. You are already used to interacting with your computer using the General User Interface (GUI). The GUI is the graphical representation that allows you to do things like open files and folders, create new folders, and more by pointing and clicking around on your computer. A CLI allows you to do the same thing, except through written commands rather than pointing and clicking.

## For Mac Users: Terminal

Mac OS users have it easy. A Bash shell comes automatically installed with all Mac operating systems. Just search for a program called **Terminal**, which allows you to access the Bash Shell.

## For Windows Users: Git Bash

Windows users will need to download Git and configure it properly so that you can use the program **Git Bash** as your Bash Shell. If you are a Windows Users, please make sure that you have watched the video on setting up your Python Environment first.

# Directory = Folder

Before we move on, keep in mind that directory is fancy computer science speak for folder. When you hear directory you can just substitute the word folder.

# Overview of Bash Commands

We will learn bash commands to do the following

* Print Working Directory: **pwd**
* List Files and Directories: **ls**
* Make Directories: **mkdir folder\_name**
* Change Directories: **cd path/to/folder**

# How to Open Your Bash Shell:

Open Your Bash Shell (Terminal for Mac OS users and Git Bash for Windows Users).

# Print Working Directory: pwd

* Will show you the path of your currently working directory of your Bash Shell
* Tells you which directory you are in

# List Files and Directories: ls

* Shows you the files and folders of your current working directory
* You can give the ls command an argument indicating the directory whose files and folder we want to list. For instance, **ls .** will print the contents of the current working directory. The “.”means “this folder”.
* ls ~ will print the contents of our home folder. The “~” means “home”.

# Make Directories: mkdir *name\_of\_directory*

* We can use the mkdir command to make a directory.
* We must give it the name of the directory we wish to create.
* Let’s create a temp folder.

# Change Directory: cd *path\_of\_directory*

* We can use cd to change into another directory.
* We must give it the path of the directory we wish to create.
* YOU MUST USE THE RIGHT **RELATIVE PATH**

## Going Backwards

* + **Go Back 1 Level**
    - **cd ..**
  + **Go back two levels**
    - cd ../..
  + **Go Back 3 levels**
    - cd ../../…
  + And so on…

## Going Home:

* + cd ~

## Going to Desktop

* + cd ~/Desktop

## Going to Documents

* + cd ~/Documents

## Tab to Autocomplete

* + Use tab to autocomplete the name of an existing folder.

## Going Forward Many Levels at Once

* + To go to a directory many levels down write the full relative path of the folder.
  + For instance cd /work/memos/august/ will go inside of the directory august as long as the path /work/memos/august/ is the correct relative path.

# Glossary

**General User Interface (GUI):** The normal way you operate your computer, by pointing and clicking.

**Command Line Interface (CLI):** A way to operate your computer using written commands.

**Bash Shell:** The CLI we will be using in this course.

**Terminal:** A program that provides a Bash Shell for Mac users

**Git Bash:** A program that provides a Bash Shell for Windows Users.

**Directory:** Directory means folder

**Working Directory:** In your Bash Shell, your working directory is the directory you are currently inside of. You can print the path of your working directory by typing pwd.

**Path:** A path is the address of a file or folder in your computer. When you first open your Git Terminal your Bash shell you be inside your home user folder.

**Argument:** An argument in bash, is any information provided alongside a bash command. For instance, the command **mkdir temp** will make a folder called temp. The **temp** argument is given to the **mkdir** command indicating that we should creating a directory called temp.

**Relative path:** A relative path is a path relative to your current working folder. For instance, if you are inside of ~/Documents/homework/chapter4/answers/ then answers is your current working directory. The relative path to your homework folder relative to answers is ../.. and if you are inside of ~/Documents then the relative path to the chapter4 folder is homework/chapter4 .