# IT 1100 : Introduction to Operating Systems

# Chapter 2

---

## The Shell Prompt

The prompt on each Linux machine may look different. But on scratch it will use this format:

`username@machinename:~$`

When I am connected to scratch it will look like this:

`smorgan@scratch:~$`

---

## The Shell Prompt

The `@` symbol is just what it looks like - an `at` symbol.

The `:` is just a separator. It acts like a space without being a space. We will use colons as separators throughout the semester.

The `~` (tilda) means that you are currently in your own home directory. If you are in a different directory it will display that for you here. Aka - the `Current Working Directory`.

---

## The Shell Prompt

Standard Users have limited permissions it is best to be logged in as a standard user as often as possible. The dollar sign `$` at the end of the prompt signifies that we are in Standard User mode.

Super Users have all permissions at all times, ie Administrative privileges. If there is a hash `#` at the end of the prompt it signifies that we are in Super User or Root mode. Avoid Super User mode if at all possible.

**Remember - Super Users have all power, all the time, to do all damage.**

---

## The home directory

In Linux there are two different meanings to the words `home directory`. It is essential to know the difference.

- The first is the main directory called home. This is where every user's personal files are stored. It's path looks like this: `/home/`
- The second is the place where each user stores his personal files. It's path looks something like this: `/home/s/smorgan/`

---

## The Shell Prompt

The user home directory is denoted in several ways. All of the following mean the same thing.

- /home/s/smorgan/
- ~ (tilda)
- my home directory
- my $HOME directory

$HOME is where you want to be.

---

## File and Directory Paths

When we declare a path to a file or directory we do so using forward slashes. Everytime we want to enter a

new directory we add another slash.

`home/s/smorgan/fruit/salad.txt`

When accessing directories the final slash is optional. Files will never end in a slash.

---

## Absolute vs Relative Path

This is a very important concept. Knowing the path to get from where you are to where you want to be gives you tremendous power.

We will be studying this all semester long.

**Absolute Path** is the path from the root ( / ) of the directory system to where you want to go.

---

## Absolute vs Relative Path

**Absolute path always begins with a slash**.

- /home/smorgan/mydirectory

**Relative Path** is the path from where you are to where you want to go.

**Relative path always begins without a slash**.

- mydirectory or mydirectory/

---

## Absolute vs Relative Path

Linux also has special notations it uses that make it easier for us to declare relative paths.

- The use of a . (dot or period) denotes start at our current directory.
- The use of two .. (dots) denotes start at our parent directory or up one level.
- Using the dots is always a relative path.

---

**More Examples**

- Are these Absolute or Relative Paths? Remember the final slash is optional for directories and files will never end in a slash.
- /home/smorgan or /home/smorgan/
- smorgan or smorgan/
- . or ./
- .. or ../
- mydirectory or mydirectory/
- ./smorgan/mydirectory or ./smorgan/mydirectory/
- ../smorgan/mydirectory or ../smorgan/mydirectory/
- /home/smorgan/myfile.txt
- smorgan/myfile.txt
- ./smorgan/myfile.txt
- ../smorgan/myfile.txt

---

** Special note **

Linux is case sensitive, thus:

- file1.txt is different than File1.txt
- /home/tom is different than /home/TOM

Linux doesn’t care about file extensions! (Application software might like it)

---

**Command summary**

- pwd
- ls
- cd
- cd ~