Bash Scripting and Globs

Exams Graded

What does a shell do?

Enter commands

Bash scripting

Command string is parsed

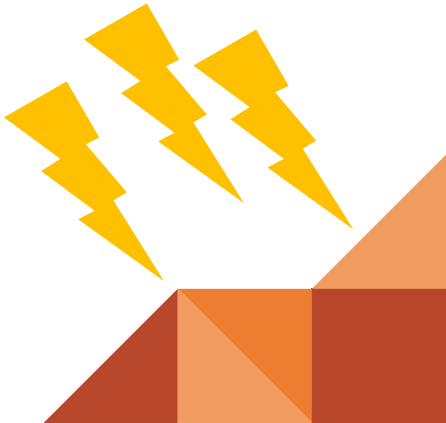
```
$ mv source_file.c destination
```

Globs/Ranges

```
1  char *args[]={
2    "mv",
3    "source_file.c",
4    "destination"
5  };
6  execvp("mv", args);
```

Bash loads the arguments
and executes the command

Stuff happens



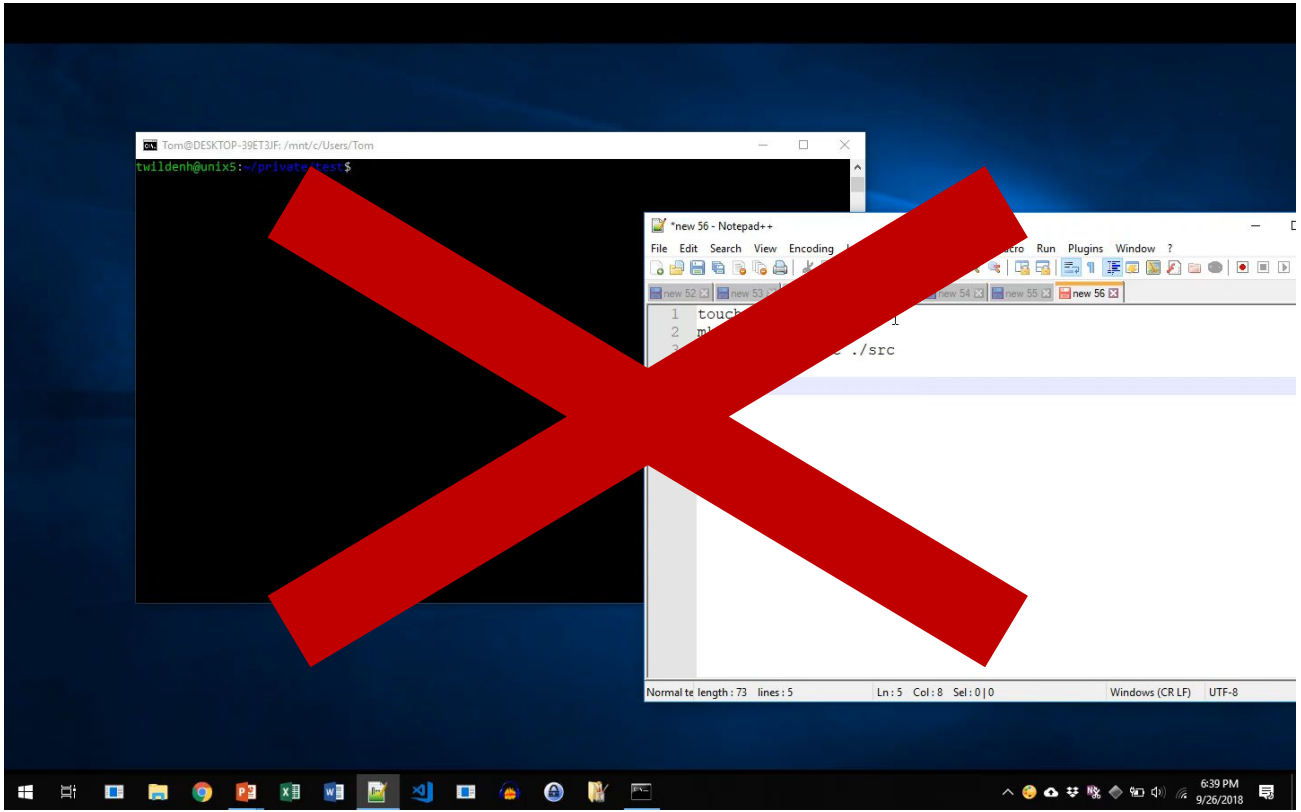
Command Review

Task	Command
Show directory contents	`ls`
Change directory	`cd directory_name`
Move a file	`mv file.txt location`
Rename a file	`mv file.txt renamed.txt`
Copy a file	`cp file.txt copy.txt`
Execute a binary	`./binary_file` ~~`binary_file`~~ `path/binary_file`
Print something	`echo "Hello World"`

Bash scripting

- Sometimes you run the same set of commands many times.
- Retyping commands isn’t very much fun
  - Unless you like typing
    - Don’t use vim
  - I don't like typing
- There’s a simple solution…

Solution?



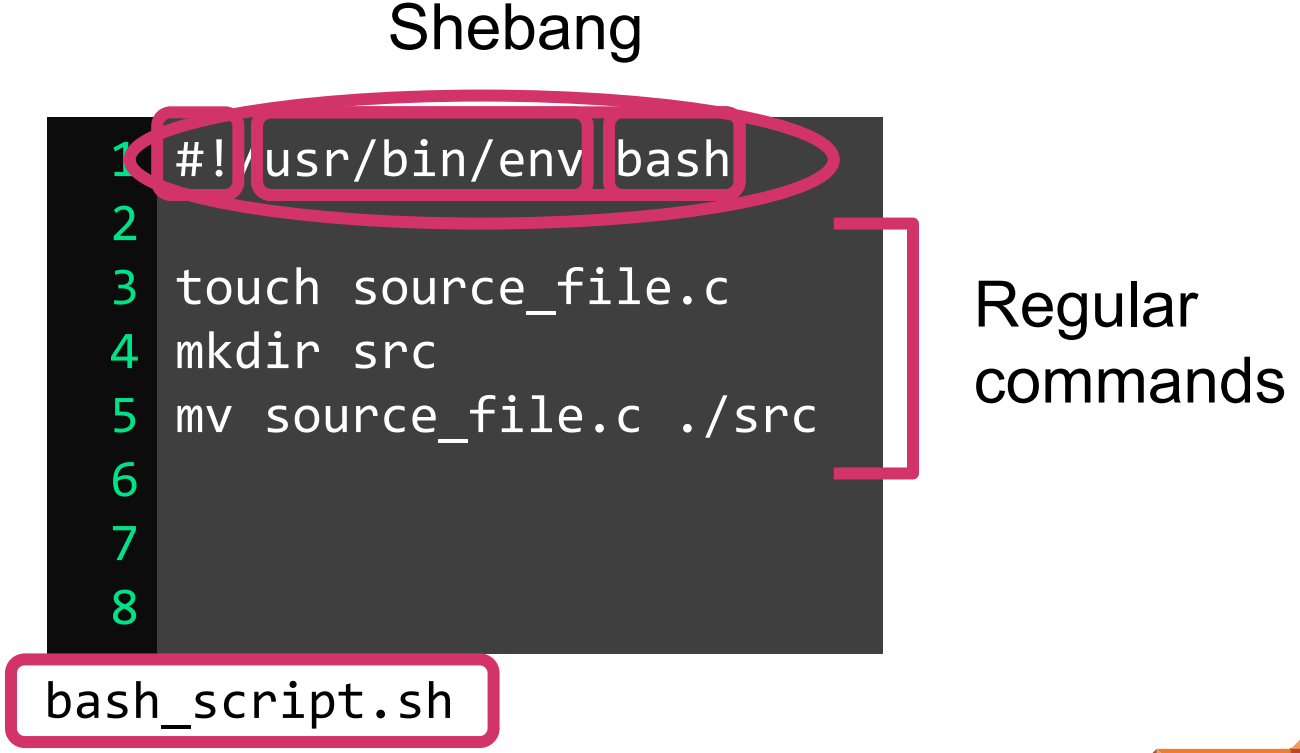
<-- Don’t do this.

Better solution: Bash scripting

- Lets you run a sequence of commands from a file
- Can be executed like a binary file

```
1  #!/usr/bin/env bash
2
3  touch source_file.c
4  mkdir src
5  mv source_file.c ./src
6
7
8
```

Example


Shebang
#!/usr/bin/env bash
touch source_file.c
mkdir src
mv source_file.c ./src
Regular commands
bash_script.sh

chmod

- Files are not executable by default

```
twildenh@unix5:~/private/script$ ./script.sh
-bash: ./script.sh: Permission denied
```

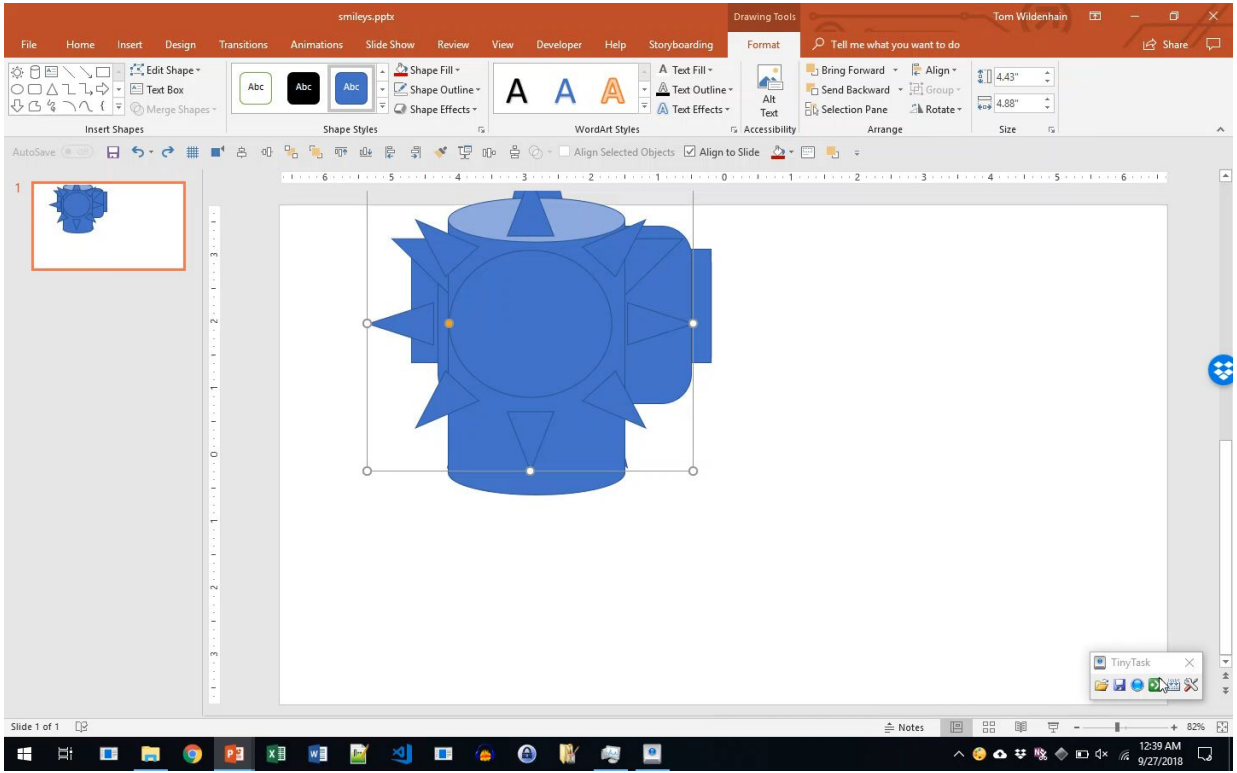
- Have to add executable permission
  - `chmod +x script.sh`
- Then we can run the script

```
twildenh@unix5:~/private/script$ ./script.sh
Hello World!
```

Bash as a programming language

- Bash also supports commands for conditionals, loops, and variables
- Automation is one of the key advantages of using a terminal

Great Impractical Ideas: Automation with TinyTask



Bash scripting summary

- Bash scripts end in a .sh extension
- Always start with a shebang
  - `#!/usr/bin/env bash`
- Add permissions with **`chmod +x script.sh`**

Globs and Ranges

What does a shell do?

Enter commands

Command string is parsed

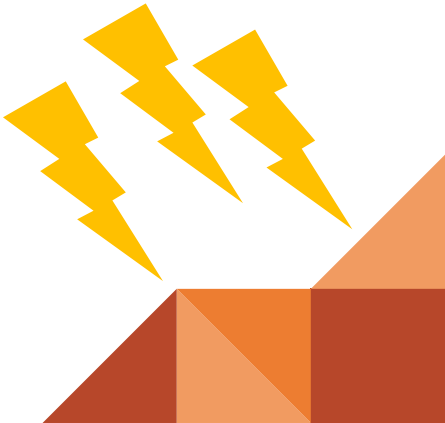
```
$ mv source_file.c destination
```

Globs/Ranges

```
1  char *args[]={
2    "mv",
3    "source_file.c",
4    "destination"
5  };
6  execvp("mv", args);
```

Bash loads the arguments
and executes the command

Stuff happens



Globs and Ranges

```
mv file{1..3}.txt dst/
```

```
mv file1.txt file2.txt file3.txt dst/
```

Ranges - { .. }

- Can be used to expand into many strings
  - Given a comma-separated sequence of words, it will expand into every permutation
  - {a,b,c} => a, b, c
  - {1,2,3}plusSome => 1plusSome, 2plusSome, 3plusSome
- You can use multiple ranges in a single line
  - {a,b,c}.{1,2,3} => a.1, a.2, a.3, b.1, b.2, b.3, c.1, c.2, c.3
  - Ranges can also figure out what you want in some cases use ..
  - {1..10} => 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
  - {a..f} => a, b, c, d, e, f

Globs

- Used to match several argument names without typing all of them out
  - `rm *.txt` removes files with the .txt extension, no matter what their names are
- Special Character: ?
  - ? matches to a, b, c, 1, 2, 3, etc…
- Special Character: *
  - Matches to any number of any character
  - * matches to any string
- Can be combined with normal text to limit matches
  - `project*.pdf` matches to any file that starts with project and ends with pdf

Quiz

Matches	Pattern
`file1 file2 file3`	`file? OR file{1..3}`
`file1 file2 item1 item2`	`{file,item}{1,2}`
`file4.pdf readme.pdf`	`*.pdf`
`file2 file3 file4.pdf`	`file{2..4}*`

Directory Contents

```
file1
file2
file3
file4.pdf
readme.pdf
item1
item2
```

Strings

- `echo * Bash scripting is fun *`
- `echo "Bash scripting is "*fun*""`
- Arguments containing spaces/special characterscan be written in quotes
  - `echo "* Bash scripting is fun *" -> Bash scripting is fun`
- They can also be written in single quotes
  - `echo 'Bash scripting is "*fun*"' -> Bash scripting is "fun"`

String Escaping

- Special characters can also be escaped with backslash
  - `echo "Bash scripting is \"fun\"" -> Bash scripting is "fun"`
- In single quotes, escape characters are ignored.
  - `echo 'Bash scripting is \"fun\"' -> Bash scripting is \"fun\"`

Bash scripting summary

- Bash scripts end in a .sh extension
- Always start with a shebang
  - `#!/usr/bin/env bash`
- Add permissions with **`chmod +x script.sh`**

Lab pro tips

- Forcelab is out!
- Don’t forget the shebang **#!/usr/bin/env bash**
- Don't forget to do **chmod +x script.sh**
- Do not copy message from the pdf, the apostrophes are different!
- Extratation on PowerPoint Turing-Machines!