Name: Kevin Tran

PID: A16809193

A **process** is what our computer keeps track of for each command while it is running. This includes:

- Where its standard input comes from (typing or file) and where its standard output should go (terminal or file)
- Its memory (like where all the variables and objects are stored) and code (Java bytecode, bash script code)
- The command line arguments it started with
- Any ports it is listening on, any files it is reading from or writing to.... and lots more!

When a **process** ends, it has a **return code** or **exit code**. The rule is that `0` means success and **non-`0`** means error.
In **bash**, the **exit code** of the **last process that ran** is stored in the special variable: **`$?`** *(yes, that's a dollar sign then a question mark)*

run-exit-code.sh

```
set -e
javac ExitCode.java
java ExitCode 0
echo $?
java ExitCode 1
echo $?
java ExitCode 3
echo $?
```

ExitCode.java

```
class ExitCode {
  public static void main(String[] args) {
    System.out.println("Exiting with code " + args[0]);
    System.exit(Integer.parseInt(args[0]));
  }
}
```

Where might it be useful to use or check an exit code?

in areas where an error might occur

or where you want to see what the variable is currently stored as

What happens when you submit a PA in CSE12?

It goes through gradescope's autograder

it takes some time but will run public/hidden test cases

we then get a score based on our implentation

```
bash-3.2$ bash run-exit-code.sh
# what's the output? why?
```

Exiting with code 0
0
Exiting with code 1

could it be that the last process that ran was stored in this case 0?

```
bash-3.2$ ls does-not-exist.txt
ls: does-not-exist: No such file or directory
bash-3.2$ echo $? # fill in a guess/the result below
```

returns the variable stored $? (in this case it's 1)

```
bash-3.2$ cat BadFile.java
class Bad {
  private statik void mane(Strong[] orgs) {}
}
bash-3.2$ javac BadFile.java
BadFile.java:2: error: <identifier> expected
  private statik void mane(Strong[] orgs) {}
                ^
1 error
bash-3.2$ echo $?
1
bash-3.2$ echo $? # fill in a guess/the result below
```

there must be nothing there? or it could be last known value (1)?

```
bash-3.2$ cat GoodFile.java
class Main {
  public static void main(String[] args) { }
}
bash-3.2$ javac GoodFile.java
bash-3.2$ echo $? $ fill in a guess/the result below
```

whatever args is?

Name: Kevin Tran	**PID:** A16809193 **(page 2)**

These are a few commonly-used commands. There are lots more! We could fill pages with them

`find «path»:` Recursively traverse the given path and list all files in that directory and subdirectories
  `-name “pattern”` option to only match files with a specific pattern (like **`find . -name "*.java"`**)
  (**`find`** has many more options! You can look them up online or with **`man`**)

`wc «file»`: Print the number of lines, words, and characters in a file or files

`grep «string» «files»:` Search a file or files for the given string, print matching lines
  `-r` option that can a directory path and search it recursively (instead of just one or more listed files)

`«command» > «file»` Save the output of the command in the given file. Overwrites the file!

`*` (asterisk, star) Used to create **patterns**, which expands to all matching paths.
Examples: `lib/*.jar`, `*.txt`

`echo «arguments»` Print the arguments to the terminal

https://github.com/bluesky-social/social-app

Bluesky, also known as **Bluesky Social**, is a microblogging social platform and a public benefit corporation**.** The service is focused on microblogging, and has been called "Twitter-like". Bluesky Social was made open source under the MIT license in May 2023.[27]

```
social-app % ls
Dockerfile                    bskyweb                       patches
Gemfile                       docs                          plugins
LICENSE                       eas.json                      scripts
Makefile                      google-services.json.example  src
… a few more files …
```

This is a big codebase! Hundreds of files.
What questions could we ask that we can answer with these commands?