# Week 04 Tutorial Sample Answers

1. A COMP2041 student wrote this script, named **start_lab04.sh**, to run before the Week 4 lab.

```
$ cat start_lab04.sh
#! /bin/dash

cd ~/labs/04

ex1=jpg2png
ex2=email_image
ex3=date_image
ex4=tag_music
```

But when he ran his script, it didn't seem to work:

```
$ pwd
/home/z1234567
$ ./start_lab04.sh
$ pwd
/home/z1234567
$ echo $ex1 $ex2 $ex3 $ex4
```

Why not? How can the sctipy be fixed?

ANSWER:

A shell script is executed by a separate shell so changes to its working directory affect only it.
Similarly. changes to variables in it affect only it.

You can indicate that the commands in a file are to be run by the current shell rather than executed as a separate program
like this:

```
$ . ./start_lab04.sh
$ pwd
/home/z1234567/labs/04
$ echo $ex1 $ex2 $ex3 $ex4
jpg2png email_image date_image tag_music
```

Or start a new shell inside the script, that will inherit the current environment:

```
$ cat start_lab04.sh
#! /bin/dash

cd ~/labs/04

ex1=jpg2png
ex2=email_image
ex3=date_image
ex4=tag_music

bash
```

2. Write a shell script, `is_business_hours` which exits with a status of 0 if the current time is between 9am - 5pm, and otherwise exits with a status of 1.

HINT:

The *date* command prints the current time in a format like this:

```
$ date
Tue 23 Aug 2022 08:42:23 AEST
```

ANSWER:

```
#! /bin/dash

current_hour="$(date "+%H")"
test "$current_hour" -ge 9 -a "$current_hour" -lt 17
```

3. COMP2041 student Shruti has a 'friends' subdirectory in her home directory that contains images of her many friends. Shruti likes to view these images often and would like to have them appear in other directories within her CSE account so she has written a shell script to symlink them to the current directory:

```
for image_file in $(ls ~/friends); do
    ln -s "~/friends/$image_file" .
done
```

The links created by Shruti's script are broken.

Why? How can she fix her script?

ANSWER:

The shell does not replace tilde (~) with the user's home directory inside double-quotes,
and does not handle spaces in filenames correctly.
For example:

```
$ echo ~
/home/shruti
$ echo "~"
~
$ touch "a b"
$ for f in $(ls); do echo "$f"; done
a
b
```

This should work for Shruti:

```
for image_file in ~/friends/*; do
    ln -s "$image_file" .
done
```

4. The course code for COMP2041 has been changed to COMP2042 and the course code for COMP9044 has been changed to COMP9042.

Write a shell script, `update_course_code.sh` which appropriately changes the course code in all the files it is given as arguments.

ANSWER:

```
#! /bin/dash

for file in "$@"; do

  if [ ! -f "$file" ]; then
    continue
  fi

  temporary_file="$(mktemp)"

  sed -E 's/COMP2041/COMP2042/g; s/COMP9044/COMP9042/g' "$file" > "$temporary_file" &&
  mv "$temporary_file" "$file"

  rm -f "$temporary_file"

done
```

Alternatively a single line solution is possible using `sed -i` (--inplace), which is widely (but not universally) supported:

```
$ sed -Ei 's/COMP2041/COMP2042/g; s/COMP9044/COMP9042/g' "$@"
```

5. Modify `update_course_code.sh` so if given a directory as an argument it updates the course codes in files found in that directory and its sub-directories.

ANSWER:

```
#!/bin/dash

for file in "$@"; do

  if [ -d "$file" ]; then
    ./"$0" "$file"/* "$file"/.[!.]* "$file"/..?*
  fi

  if [ ! -f "$file" ]; then
    continue
  fi

  temporary_file="$(mktemp)"

  sed 's/COMP2041/COMP2042/g; s/COMP9044/COMP9042/g' "$file" > "$temporary_file" &&
  mv "$temporary_file" "$file"

  rm -f "$temporary_file"

done
```

Alternatively a single line solution is possible using `sed -i` (--inplace), which is widely (but not universally) supported:

```
$ find "$@" -type f -exec sed -i 's/COMP2041/COMP2042/g; s/COMP9044/COMP9042/g' {} \;
```

6. CSE systems have a command, `mlalias`, which prints information about a specified mail alias.
For example:

```
$ mlalias cs2041.22T2.tut04
        Alias: cs2041.22T2.tut04
  Description: example alias for COMP2041/9055 22T2
        Flags: personal, public, unprivileged, moderated, open
    Addresses:
                z5334819
                z5406076
                z5300720
                z5003865
                ...
                z5506985
                z5522312
                z5550177
                z5370610
       Owners: cs2041
      Senders:
```

Convert the output of the `mlalias` command into a new line separated list of CSE usernames,
like this:

```
z5334819
z5406076
z5300720
z5003865
...
z5506985
z5522312
z5550177
z5370610
```

ANSWER:

```
$ mlalias cs2041.22T2.tut04 |
sed -n '/Addresses/,/Owners/p' |
head -n -1 |                       # Remove last line
tail -n +2 |                       # Remove first line
sed -E 's/^\s*//; s/\s*$//' |      # Remove leading and traling spaces
cut -d@ -f1                        # Remove domain from email address
```

alternatively:

```
$ mlalias -r cs2041.22T2.tut04 | # Use "raw" output mode
cut -d: -f3 |                    # Take field 3 (addresses)
tr ',' '\n' |                    # replace commas with newlines
cut -d@ -f1                      # Remove domain from email address
```

7. CSE system have a command, `acc` , which prints information about a specified user.
   For example:

```
$ acc z5555555
            User Name : z5555555              Aliases :
                  Uid : 25068                     Gid : 25068
              Expires : 19Dec 2023

               Groups :
        Group classes :
         User classes : 3978_Student, COMP1521t1_Student[10may2022]
                      : COMP1531t1_Student[10may2022], COMP2041t2_Student[26sep2022]
                      : COMP2121t2_Student[26sep2022], COMP2511t1_Student[26sep2022]
                      : COMP1511t1_Tutor[10may2022], COMP2041t2_Tutor[26sep2022]
         Misc classes :

                 Name : Mr Doe, John (John Doe)
             Position : Casual Academic (Sch: Computer Science & Eng)
          UNSW Number : 5555555
            UNSW Mail : z5555555@unsw.edu.au
            UNSW Home : //INFPWFS219.ad.unsw.edu.au/Student038$/z5555555
             CSE Home : /import/kamen/3/z5555555
```

   Write a pipeline which converts the output of acc into a new line separated list of courses the person is enrolled as a student in, like this:

```
COMP1521
COMP1531
COMP2041
COMP2121
COMP2511
```

   Make sure you don't include COMP1511 in which John tutors.

ANSWER:

```
$ acc z5555555 |
sed -n '/^$/,/^$/p' |
cut -d: -f2 |
tr ',' '\n' |
sed -nE 's/.*([A-Z]{4}[0-9]{4})t[0-3]_Student.*/\1/p'
```

8. Use the pipeines from the above 2 questions to write shell commands which print a list of courses taken by COMP2041 students with counts of how many COMP2041 students take each,
   like this:

```
7 COMP6771
4 COMP3511
3 COMP4952
3 COMP4951
3 COMP3141
2 COMP9417
...
```

ANSWER:

```
#! /bin/dash

mlalias cs2041.22T2.tut04 |
sed -n '/Addresses/,/Owners/p' |
head -n -1 |
tail -n +2 |
sed -E 's/^\s*//; s/\s*$//' |
cut -d@ -f1 |
grep '.' |
while read zid; do
    acc "$zid" |
    sed -n '/^$/,/^$/p' |
    cut -d: -f2 |
    tr ',' '\n' |
    sed -nE 's/.*([A-Z]{4}[0-9]{4})t[0-3]_Student.*/\1/p'
done |
sort |
uniq -c |
sort -rn
```

9. Write a shell script named `is_prime.sh` which given an integer as an argument, tests whether it is prime and prints a suitable message:

```
$ is_prime.sh 42
42 is not prime
$ is_prime.sh 113
113 is prime
```

Your script should exit with a non-zero exit status if its argument is not prime.

Write a second script named `primes.sh` which uses the first script to print all primes less than a specified value, e.g:

```
$ primes.sh 100
2
3
5
7
11
13
17
...
79
83
89
97
```

ANSWER:

is_prime.sh

```
#!/bin/dash

# test whether the specified integer is prime

if [ $# -ne 1 ] || ! [ "$1" -eq "$1" ] 2>/dev/null || [ "$1" -lt 0 ]; then
  echo "Usage: $0 <positive-number>"
  exit 2
fi

number=$1

if [ "$number" -eq 1 ]; then
  echo "$number is not prime"
  exit 1
fi

i=2

while [ $((i * i)) -le "$number" ]; do
  if [ $((number % i)) -eq 0 ]; then
    echo "$number is not prime"
    exit 1
  fi
  i=$((i + 1))
done

echo "$number is prime"

exit 0
```

primes.sh

```
#!/bin/dash

# print the prime numbers less than the specified argument

if [ $# -ne 1 ] ||  ! [ "$1" -eq "$1" ] 2> /dev/null || [ "$1" -lt 0 ]; then
    echo "Usage: $0 <number>"
    exit 2
fi

limit=$1
p=2

while [ "$p" -lt "$limit" ]; do
  if ./is_prime.sh "$p" > /dev/null; then
    echo $p
  fi
  p=$((p + 1))
done
```

10. We are working on a C program in an editor, and we'd like to run a script in another window which recompiles the program every time we save the file in the editor.

    Write a shell script named `recompile.sh` which given a C file as argument, recompiles the program if the file changes and if the compile succeeds runs the program.

    Hint: you can use the program `stat` to print the time when the files was last modified (as seconds since 1970-01-01) , like this:

```
$ stat -c '%Y' main.c
1615346166
```

ANSWER:

```
#! /bin/dash

C_source_file="$1"

while true; do
    sleep 1

    # get modification time of file in seconds since 01/01/70
    modification_time=$(stat -c '%Y' "$C_source_file") || continue

    test "$modification_time" = "$last_modification_time" && continue

    # compile file and run if compile successful
    clang "$C_source_file" && ./a.out

    last_modification_time="$modification_time"
done
```

Repeatedly checking if a file has changed is not ideal. It consumes CPU/power.

The program `inotifywait` is available on many Linux system linux (not CSE).

`inotifywait -e main.c` will exit when `main.c` is modified

`inotifywait` is efficient because waits for the operating system interface to notify it the file has changed..

Modify your script to use `inotifywait`

ANSWER:

```
#! /bin/dash

C_source_file="$1"

while inotifywait -e modify "$C_source_file"; do
    # compile file and run if compile successful
    dcc "$C_source_file" && ./a.out
done
```

11. The shell variable **`$PATH`** contains a colon-separated list of directories. which will be searched when executing a command.

    Write a shell script named `which.sh` which given a program name as argument, uses to ls to print details of matching files in directories in **`$PATH`**

    For example:

```
$ ./which.sh cat
-rwxr-xr-x 1 root root 43936 Sep 24 18:36 /bin/cat
$ ./which.sh clang
llrwxrwxrwx 1 root root 24 Jan 12 01:49 /usr/bin/clang -> ../lib/llvm-11/bin/clang
$ ./which.sh lost
lost not found
```

The shell builtin **`which`** does something similar:

```
$ which cat
/bin/cat
$ which clang
/usr/bin/clang
$ which lost
$
```

but don't try using **`which`**. Use the usual programs we've been using to write scripts such as **tr** and **test**.

Think about if any characters in directory names migh break your answer, e.g spaces.

ANSWER:

```
#! /bin/dash

if test $# = 0; then
    echo "Usage $0: <program>" 1>&2
    exit 1
fi

for program in "$@"; do
    program_found=''
    directories=$(echo "$PATH" | tr ':' ' ')
    for directory in $directories; do
        f="$directory/$program"
        if test -x "$f"; then
            ls -ld "$f"
            program_found=1
        fi
    done
    if test -z $program_found; then
        echo "$program not found"
    fi
done
```

```
#! /bin/dash

if test $# = 0; then
    echo "Usage $0: <program>" 1>&2
    exit 1
fi

for program in "$@"; do

    n_path_components=$(echo "$PATH"|tr -d -c :|wc -c)

    index=1
    while test $index -le "$n_path_components"; do
        directory=$(echo "$PATH"|cut -d: -f$index)
        f="$directory/$program"
        if test -x "$f"; then
            ls -ld "$f"
            program_found=1
        fi
        index=$((index + 1))
    done
    test -n $program_found || echo "$program not found"
done
```

```
#! /bin/dash

if test $# = 0; then
    echo "Usage $0: <program>" 1>&2
    exit 1
fi

for program in "$@"; do
    echo "$PATH" |
    tr ':' '\n' |
    while read directory; do
        f="$directory/$program"
        test -x "$f" && ls -ld "$f"
    done |
    grep '.' || echo "$program not found"
done
```