

COMP2113 Programming Technologies
ENGG1340 Computer Programming II
Module 1 Checkpoint Exercise

Name: YUAN Wenxuan
University ID: 3036292740

Instructions:

For each single question or each group of questions in the Checkpoint exercise, please type your answer right after the question in this Word document. Please refer to the example below.

Checkpoint 0:

What is the meaning of the command “date”?

Ans: The “date” command prints the current date of the current machine

Checkpoint 1.1

Now, let’s try to answer the following questions. Although you haven’t been taught the meaning of the following commands, you can display the manual page of these commands and learn their meanings by yourself

1. Why do we need to learn command line although we can use a GUI to control a computer?
2. What is the meaning of `ls -t`?
3. What is the meaning of the command `pwd`?
4. What is the meaning of the command `rm`?
5. What is the meaning of the command `mv`?
6. Suppose that the `fileA` does not exist in your present working directory, what is the meaning of the command `touch fileA`?
7. What is the meaning of the command `tar`?
8. What is the command for creating an archive `files.tar` from two files named `fileA` and `fileB`?

Ans:

1. The command line helps control the computer automatically or remotely.
2. This command lists the directory contents sorted by modification time, and the newest one comes first.
3. This command prints the full filename of the current working directory.
4. This command removes files or directories.
5. This command moves or renames files.
6. This command creates fileA in the current directory.
7. This command is an archiving utility.
8. `tar -cf files.tar fileA fileB`

Checkpoint 1.2a

Assume we have logged in Ubuntu and started a bash shell. The current directory is the home directory, i.e., ~ . We want to perform the following tasks sequentially. For each of the tasks below, please state the shell command(s) used to perform it.

1. Create a new subdirectory “*assignments*” under ~.
2. Create a new subdirectory “*assignment 1*” under “*assignments*”. (Note that we are creating one subdirectory “*assignment 1*” but not two subdirectories “assignment” and “1”)
3. Remove the directory “*assignments*” and all its subdirectories.

Ans:

1. `mkdir assignments`
2. `mkdir "assignments/assignment 1"`
3. `rm -r assignments`

Checkpoint 1.2b

[Self-learning question] - You need to search for the information on the Internet to answer this question.

There is another way to modify the permission, which is called the Absolute mode.

- a. Explain the meaning of `chmod 666 hello.txt`
- b. Explain the meaning of `chmod 700 hello.txt`
- c. What is the `chmod` command, in absolute mode, to set the following permission for *hello.txt*?

User permissions			Group Permission			Other permission		
r	w	x	-	w	-	r	-	-

- d. The administrator says that “One does not simply 777 their entire server”, explain what the problem is if we `chmod 777` for all the files.

Ans:

- a) Set the permission for *hello.txt* so everyone can read and write it, but no one can execute it.
- b) Set the permission for *hello.txt* so that its owner can read, write, and execute it while others cannot do so.
- c) `chmod 724 hello.txt`
- d) It grants full permission to everyone, causing a major security risk.

Checkpoint 1.3

Now you may have a doubt: I understand how **diff** works, but why the output is claimed to be the difference between the two files?

Consider the two files below:

<pre>\$cat question1A Apple Boy Cat Dog Egg</pre>	<pre>\$cat question1B Boy Cat Egg</pre>
---	---

Note that file **question1B** is created by removing “Apple” and “Dog” from the file **question1A**.

- A. What will be the output if we execute the following command (Please try to think about the output before trying it in the shell)? Please explain your answer.

```
$diff question1A question1B
```

- B. What will be the output if we execute the following command (Please try to think about the output before trying it in the shell)? Please explain your answer.

```
$diff question1B question1A
```

Ans:

- a) 1d0
 < Apple
 4d2
 < Dog
- b) 0a1
 > Apple
 2a4
 > Dog
-

Checkpoint 1.4

This is a challenging exercise! You need to understand the shell commands and the techniques introduced in the previous sections to work on this task.

The following C++ program *gen4.cpp* reads in a 4-character string from the input and generates all possible permutations from the 4 characters.

```
//gen4.cpp
#include <iostream>
#include <string>
int main() {
    std::string s;
    std::cin >> s;
    for (int i = 0; i < s.length(); i++) {
        for (int j = 0; j < s.length(); j++) {
            for (int k = 0; k < s.length(); k++) {
                for (int l = 0; l < s.length(); l++) {
                    if (i != j && i != k && i != l && j != l && j != k && k != l) {
                        std::cout << s[i] << s[j] << s[k] << s[l] << std::endl;
                    }
                }
            }
        }
    }
    return 0;
}
```

To compile *gen4.cpp*

```
$ g++ gen4.cpp -o gen4
```

The input of the program should be stored in the file *gen4_input.txt* with the following content.

```
lopo
```

gen4_input.txt

1. Give **ONE** command (one line of command(s)) to run the *gen4* with *gen4_input.txt* as input and redirect the result to a file named *gen4_output.txt*.

Hints:

```
$ [your_command]
$ cat gen4_output.txt
lopo
loop
lpoo
lpoo
...
$ wc gen4_output.txt
24 24 120 gen4_output.txt
```

*The output file should contain all permutations of the letters 'l', 'o', 'p' and 'o'. There should be 24 permutations in total.

Ans: `cat gen4_input.txt | ./gen4 > gen4_output.txt`

2. Give **ONE** command to sort the words in *gen4_output.txt* in alphabetical order, and then also remove the adjacent duplicate lines and finally store the result in a file named *sort_uniq.txt*.

Hints: Consider the command **uniq**

```
$ [your command]
$ cat sort_uniq.txt.
loop
lopo
lpoo
olop
olpo
oolp
oopl
oplo
opol
plooo
polo
pool
$ wc sort_uniq.txt
12 12 60 sort_uniq.txt
```

*There should be 12 unique words total.

Ans: `sort gen4_output.txt | uniq > sort_uniq.txt`

3. Give **ONE** command to check the spelling in *sort_uniq.txt* and store the misspelled words into another file named *misspell.txt*.

Ans: `spell sort_uniq.txt > misspell.txt`

4. Now *sort_uniq.txt* contains all distinct generated words, and *misspell.txt* contains all misspelled words. The differences between the two files are the meaningful 4-character words. Give **ONE** command to return the correctly spelled words as shown below:

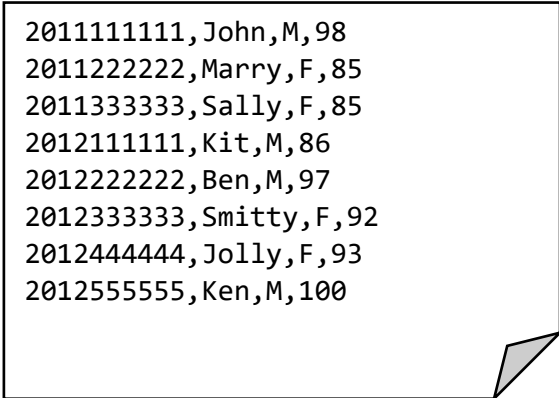
```
$ [your command]
< loop
< polo
< pool
```

Hints: Consider the command **diff** and **grep**.

Ans: `diff sort_uniq.txt misspell.txt | grep -f sort_uniq.txt`

Checkpoint 1.5

Consider the file *question1.txt*.



```
2011111111,John,M,98
2011222222,Marry,F,85
2011333333,Sally,F,85
2012111111,Kit,M,86
2012222222,Ben,M,97
2012333333,Smitty,F,92
2012444444,Jolly,F,93
2012555555,Ken,M,100
```

Figure 1 *question1.txt*

1. Give ONE command to return the lines that contain the record of Kit

Hints:

```
$ [Your command]
2012111111,Kit,M,86
```

Ans: `grep Kit question1.txt`

2. Give ONE command to find the students with UID begin with “**2012**” (i.e., To find the lines that begin with 2012)

Hints:

```
$ [Your command]
2012111111,Kit,M,86
2012222222,Ben,M,97
2012333333,Smitty,F,92
2012444444,Jolly,F,93
```

```
2012555555, Ken, M, 100
```

Ans: `grep ^2012 question1.txt`

3. Give ONE command to return the lines that contain the record of the students who are both:

- UID start at **2012**, and
- Name starts with the characters **J** or **S**

Hints:

```
$ [Your command]  
2012333333, Smitty, F, 92  
2012444444, Jolly, F, 93
```

Ans: `grep ^2012.*,.*[JS] question1.txt`