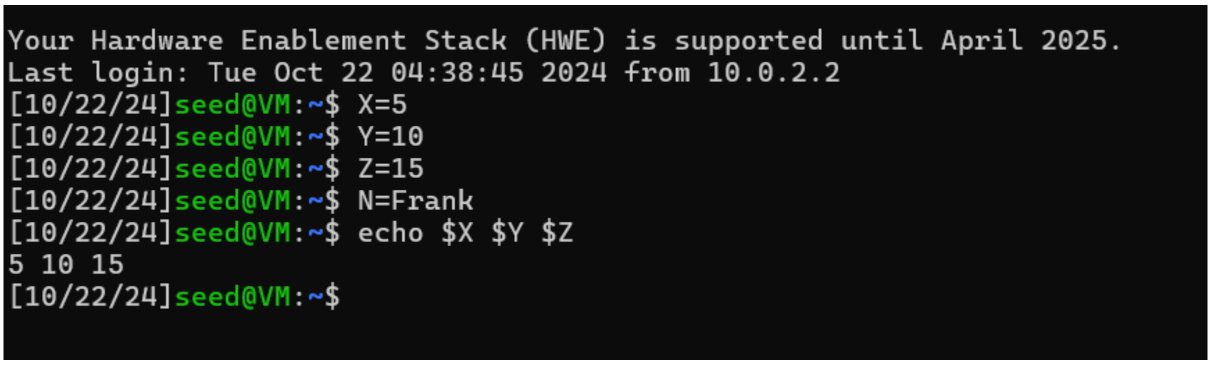
Name: Nguyễn Sỹ Thủy

Student ID: ITITIU21326

Laboratory Session 4

Shell Script

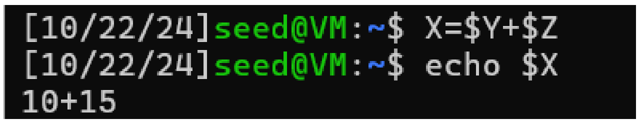
Question 1. What will echo $X $Y $Z output?



```
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Tue Oct 22 04:38:45 2024 from 10.0.2.2
[10/22/24]seed@VM:~$ X=5
[10/22/24]seed@VM:~$ Y=10
[10/22/24]seed@VM:~$ Z=15
[10/22/24]seed@VM:~$ N=Frank
[10/22/24]seed@VM:~$ echo $X $Y $Z
5 10 15
[10/22/24]seed@VM:~$
```

The output is 5 10 15 as echo is command to display

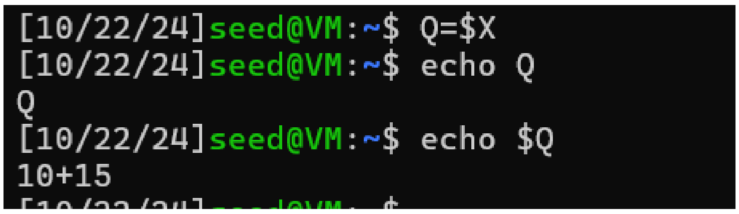
Question 2. What will X store after executing X=$Y+$Z?



```
[10/22/24]seed@VM:~$ X=$Y+$Z
[10/22/24]seed@VM:~$ echo $X
10+15
```

The result is 10+15

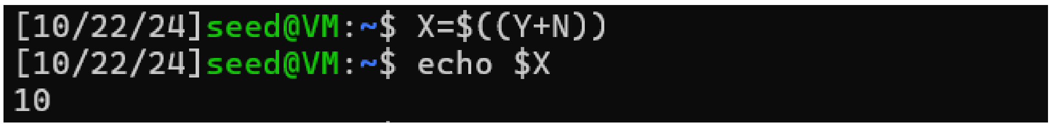
Question 3. What is wrong with the following statement? Q=$X



```
[10/22/24]seed@VM:~$ Q=$X
[10/22/24]seed@VM:~$ echo Q
Q
[10/22/24]seed@VM:~$ echo $Q
10+15
```

There is nothing wrong with Q=$X as Q is gained the value so when echo $Q, value of $X will be display

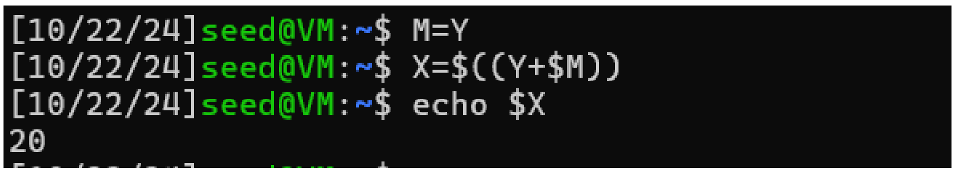
Question 4. Does the following result in an error? If not, what value does X obtain? X=$((Y+N))



```
[10/22/24]seed@VM:~$ X=$((Y+N))
[10/22/24]seed@VM:~$ echo $X
10
```

The value is 10 and it is not creating any error

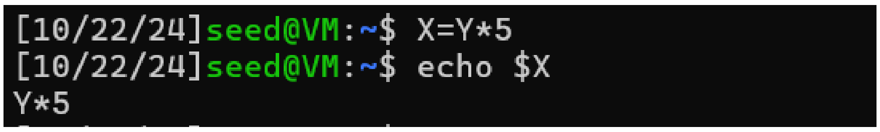
Question 5. Does the following result in an error? If not, what value does X obtain? Explain how shell compute X.



```
[10/22/24]seed@VM:~$ M=Y
[10/22/24]seed@VM:~$ X=$((Y+$M))
[10/22/24]seed@VM:~$ echo $X
20
```

It will not result an error, the value that X obtain is 20, cleary seen that M =Y and YU =10 so Y +M = 10+10 =20

Question 6. What is the value stored in X after the instruction let "X=Y*5"?

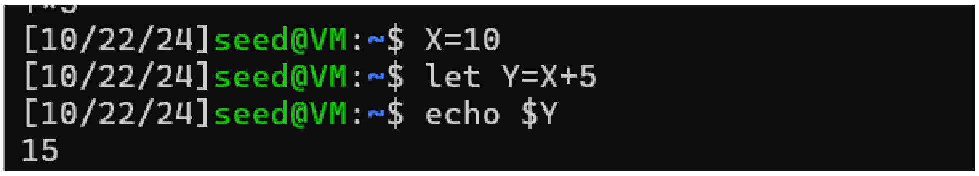


```
[10/22/24]seed@VM:~$ X=Y*5
[10/22/24]seed@VM:~$ echo $X
Y*5
```

The value of X will be displayed as Y*5 as there are no $ to gain the value of Y

Question 7. What does the command let do?

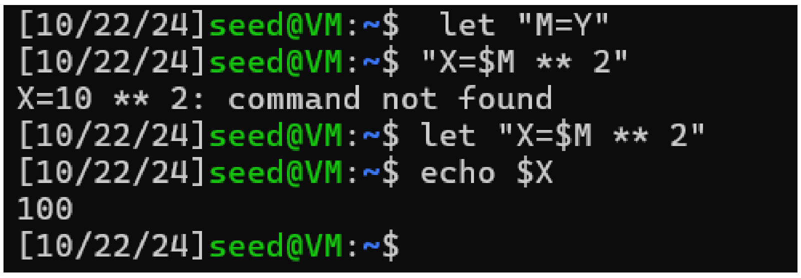
The let command is used to perform arithmetic operations on shell variables. It allows you to manipulate the values of variables with arithmetic expressions.



```
[10/22/24]seed@VM:~$ X=10
[10/22/24]seed@VM:~$ let Y=X+5
[10/22/24]seed@VM:~$ echo $Y
15
```

Example of let command

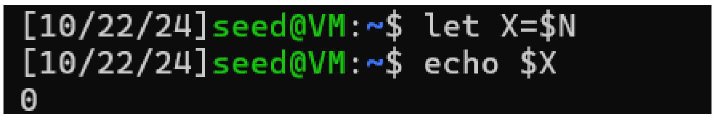
Question 8. What is the value stored in X after the instruction let "M=Y" "X=$M ** 2"? Explain how shell compute X



```
[10/22/24]seed@VM:~$  let "M=Y"
[10/22/24]seed@VM:~$ "X=$M ** 2"
X=10 ** 2: command not found
[10/22/24]seed@VM:~$ let "X=$M ** 2"
[10/22/24]seed@VM:~$ echo $X
100
[10/22/24]seed@VM:~$
```

We see the resule is 100, as let is the command for algorithm M=Y=10 ; M**2 is M square 2 = 10*10 =100, then we use echo command to display the result

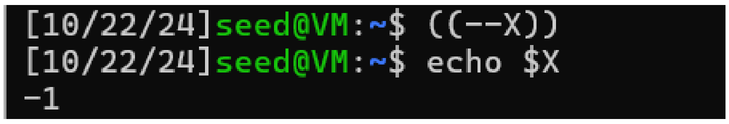
Question 9. Does the following result in an error? If not, what value does X obtain? Explain the result. let X=$N



```
[10/22/24]seed@VM:~$ let X=$N
[10/22/24]seed@VM:~$ echo $X
0
```

The value obtained in X is 0 as N = Frank is a character and not a value

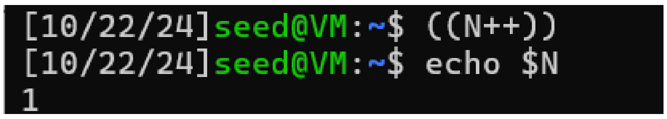
Question 10. What is the value stored in X after the instruction ((--X))?



```
[10/22/24]seed@VM:~$ ((--X))
[10/22/24]seed@VM:~$ echo $X
-1
```

The previous question show that X =0 so when we type ((--X)) it will show X = -1

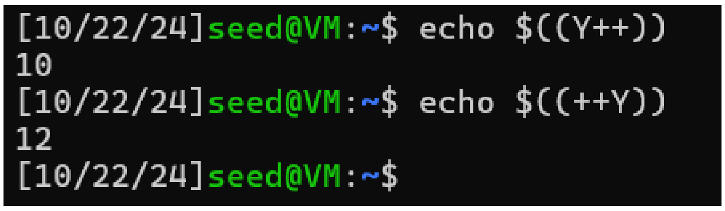
Question 11. What is the value stored in N after the instruction ((N++))?



```
[10/22/24]seed@VM:~$ ((N++))
[10/22/24]seed@VM:~$ echo $N
1
```

The value of ((N++)) is 1

Question 12. What is the result of the instruction echo $((Y++)) ? Would echo $((++Y)) do something different?



```
[10/22/24]seed@VM:~$ echo $((Y++))
10
[10/22/24]seed@VM:~$ echo $((++Y))
12
[10/22/24]seed@VM:~$
```

The innitial value of Y =10; after Y++ the resuklt is saved as 11, type the ++Y again, the value will increase by 1 number so ++Y =12

Section 2

Assume A=one and B=two

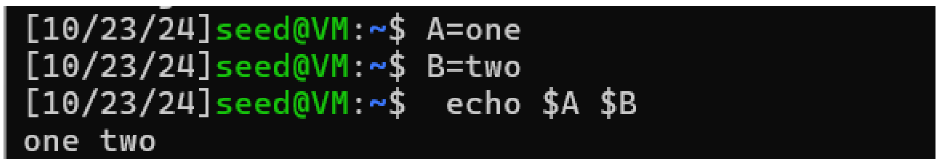
Question 13. Which of the following are legal variable names in bash? VARIABLE A_VARIABLE; VARIABLE_1 1_VARIABLE variable A-VARIABLE while WHILE.

In bash, there are specific rules for legal variable names. The rules are:

1. Variable names must start with a letter or an underscore (_).
2. Variable names can contain letters, numbers, and underscores.
3. Variable names cannot start with a number.
4. Hyphens (-) and special characters (like !, @, $) are not allowed in variable names

```
VARIABLE; A_VARIABLE; VARIABLE_1; variable; WHILE are valid names
```

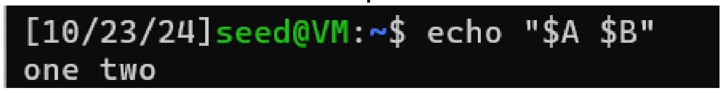
Question 14. What is output with the statement echo $A $B?



```
[10/23/24]seed@VM:~$ A=one
[10/23/24]seed@VM:~$ B=two
[10/23/24]seed@VM:~$  echo $A $B
one two
```

The result after executing is one two as they display the value of A and B

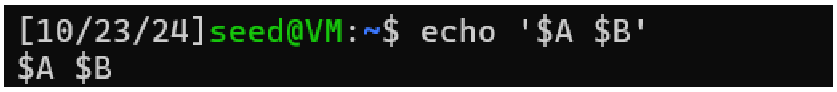
Question 15. What is output with the statement echo "$A $B"?



```
[10/23/24]seed@VM:~$ echo "$A $B"
one two
```

It gives the same value as question 14

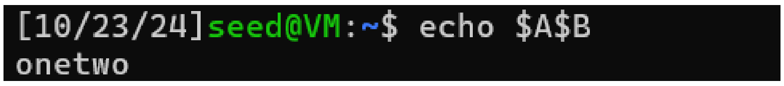
Question 16. What is output with the statement echo '$A $B'?



```
[10/23/24]seed@VM:~$ echo '$A $B'
$A $B
```

There is only one ' which indicates the inside so echo '$A $B' which means print $A $B instead of its value

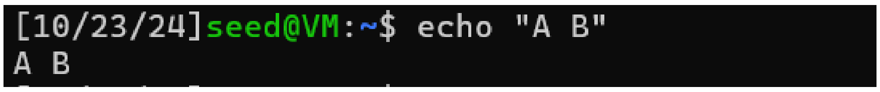
Question 17. What is output with the statement echo AB?



```
[10/23/24]seed@VM:~$ echo $A$B
onetwo
```

There is no space so onetwo is stick together

Question 18. What is output with the statement echo "A B"?



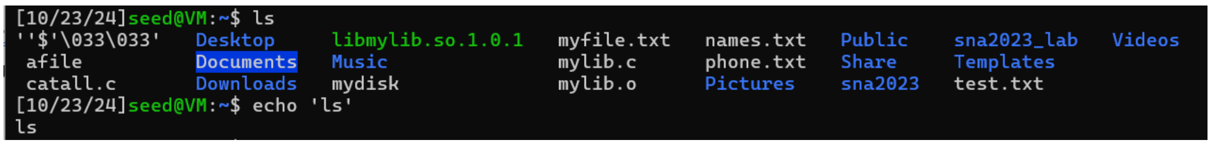
```
[10/23/24]seed@VM:~$ echo "A B"
A B
```

There is no $ before A and B , so echo will print A, B instead of it's value

Question 19. What is the difference in the output of the following two statements?

ls

echo `ls`



```
[10/23/24]seed@VM:~$ ls
''$'\033\033'   Desktop     libmylib.so.1.0.1   myfile.txt   names.txt   Public    sna2023_lab   Videos
 afile          Documents   Music               mylib.c      phone.txt   Share     Templates
 catall.c       Downloads   mydisk              mylib.o      Pictures    sna2023   test.txt
[10/23/24]seed@VM:~$ echo 'ls'
ls
```

ls is the command to display the list of files

echo 'ls' which means print ls word

Question 20. You have executed the script foo.sh as ./foo.sh 5 10 15 20. What are the values of each of the following? $0, $1, $2, $3, $4, $5, $#, $*.

When execute a bash script ./foo.sh 5 10 15 20, each positional parameter is assigned a value based on its position, and there are special variables used to access them. Here's what happens:

$0: The name of the script (in this case, ./foo.sh).

$1: The first argument passed to the script (in this case, 5).

$2: The second argument passed to the script (in this case, 10).

$3: The third argument passed to the script (in this case, 15).

$4: The fourth argument passed to the script (in this case, 20).

$5: Undefined or empty (no fifth argument was passed).

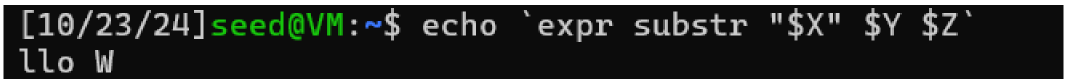
$#: The number of arguments passed to the script (in this case, 4).

$*: A single string containing all the arguments passed, separated by spaces (5 10 15 20).

Section 3

Assume X="Hello World", Y=3, Z=5, and S="acegiklnor".

Question 21. What is output from the statement echo `expr substr "$X" $Y $Z`?



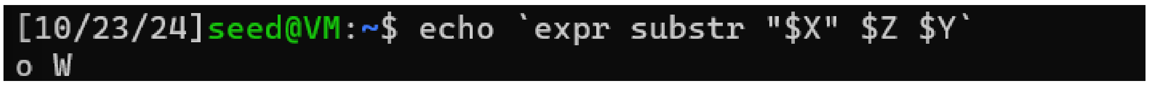
```
[10/23/24]seed@VM:~$ echo `expr substr "$X" $Y $Z`
llo W
```

The expr substr command in bash extracts a substring from a given string. The syntax is:

expr substr string position length

Here string is Hello World, position is 3 which starts from l, length = 5 (include a space) so we see llo W.

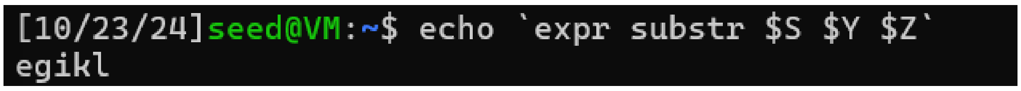
Question 22. What is output from the statement echo `expr substr "$X" $Z $Y`?



```
[10/23/24]seed@VM:~$ echo `expr substr "$X" $Z $Y`
o W
```

Here string is Hello World, position is 5 which starts from o, length = 3 (include a space) so we see o W.

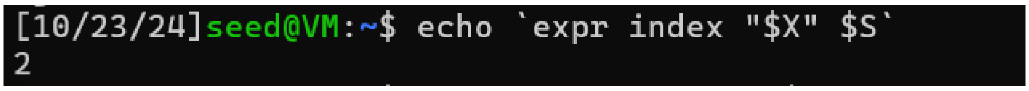
Question 23. What is output from the statement echo `expr substr $S $Y $Z`?



```
[10/23/24]seed@VM:~$ echo `expr substr $S $Y $Z`
egikl
```

Here string is acegiklnor, position is 3 which starts from e, length = l (include a space) so we see egikl.

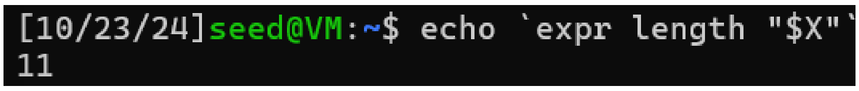
20. What is output from the statement echo `expr index "$X" $S`?



```
[10/23/24]seed@VM:~$ echo `expr index "$X" $S`
2
```

Index indicates the value of the character “Hello World” is 2 so index is 2

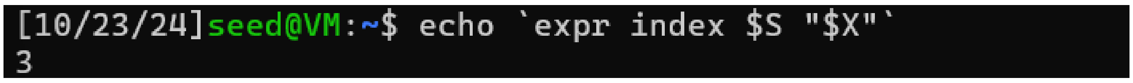
Question 24. What is output from the statement echo `expr length "$X"`?



```
[10/23/24]seed@VM:~$ echo `expr length "$X"`
11
```

Length is used to detect each characters, including the spaces “ Hello World” has 11 characters, so it is 11

Question 25. What is output from the statement echo `expr index $S "$X"`?



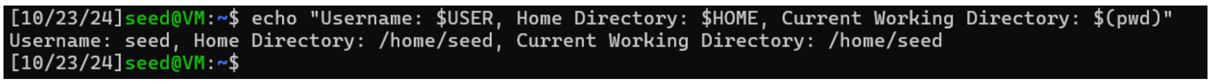
```
[10/23/24]seed@VM:~$ echo `expr index $S "$X"`
3
```

The expr index command in bash searches for the first occurrence of any character from a search string in a target string. The syntax is:

expr index string characters

Here, it is similar in the l option, so it is 3 "He***ll***o World" ;"acegik***l***nor"

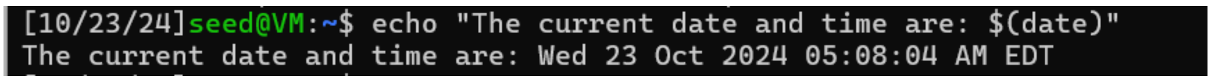
Question 26. Write an echo statement to output on one line the current user's username, home directory, and current working directory.



```
[10/23/24]seed@VM:~$ echo "Username: $USER, Home Directory: $HOME, Current Working Directory: $(pwd)"
Username: seed, Home Directory: /home/seed, Current Working Directory: /home/seed
[10/23/24]seed@VM:~$
```

We use echo command to write out $USER : name of the user; $HOME is home directory; $(pwd) is the current working directory in the command

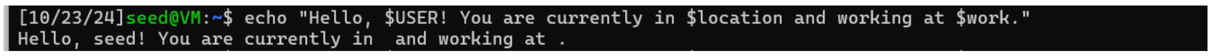
Question 27. Write an echo statement to say "the current date and time are" followed by the result of the date command.



```
[10/23/24]seed@VM:~$ echo "The current date and time are: $(date)"
The current date and time are: Wed 23 Oct 2024 05:08:04 AM EDT
```

$(date) is the command to display the time

Question 28. Assume we have variables location and work which store the city name and company name of the current user. Write an echo statement to output a greeting message including the user's username, his/her city name, and the company name.

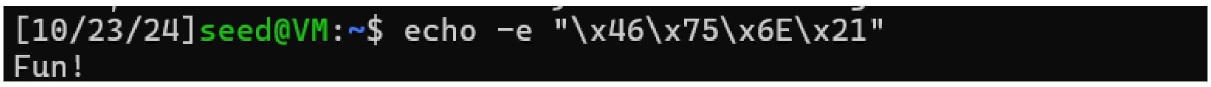


```
[10/23/24]seed@VM:~$ echo "Hello, $USER! You are currently in $location and working at $work."
Hello, seed! You are currently in  and working at .
```

The command is very similar to other questions

Question 29. Locate a listing of all ASCII characters (man ascii). Using the notation *n*xHH, write an echo statement to output the string "Fun!". For example, echo "*n*x61"

2



```
[10/23/24]seed@VM:~$ echo -e "\x46\x75\x6E\x21"
Fun!
```

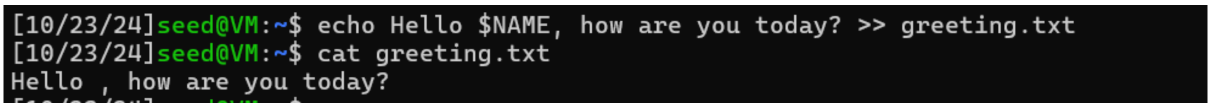
n bash, you can use the echo command with the -e option to enable interpretation of backslash escapes, including the ASCII character notation \xHH where HH is the hexadecimal representation of the ASCII character.

For the string "Fun!", the ASCII values are:

F = \x46; u = \x75; n = \x6E; ! = \x21

Question 30. What does the following instruction do assuming NAME stores your name?

echo Hello $NAME, how are you today? >> greeting.txt



```
[10/23/24]seed@VM:~$ echo Hello $NAME, how are you today? >> greeting.txt
[10/23/24]seed@VM:~$ cat greeting.txt
Hello , how are you today?
```

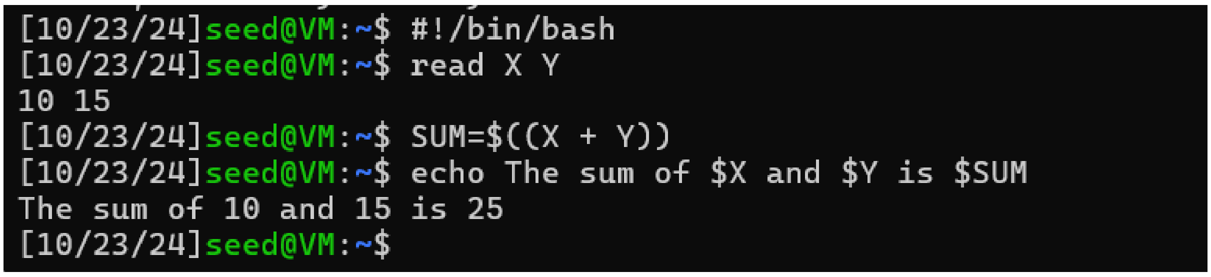
The command will input a file onto it, it just a command " Hello, how are you today

Question 31. The script below will input two values from the user and sum them. What is wrong with the script? (there is nothing syntactically wrong). How would you correct it?

#!/bin/bash

read X Y SUM=$((X+Y))

echo The sum of $X and $Y is $SUM

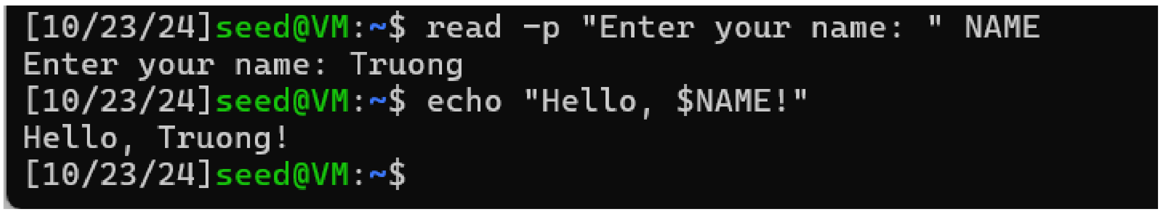


```
[10/23/24]seed@VM:~$ #!/bin/bash
[10/23/24]seed@VM:~$ read X Y
10 15
[10/23/24]seed@VM:~$ SUM=$((X + Y))
[10/23/24]seed@VM:~$ echo The sum of $X and $Y is $SUM
The sum of 10 and 15 is 25
[10/23/24]seed@VM:~$
```

There is no error on this code , as it calculate and print out the sum

Question 32. What is the -p option used for in read?

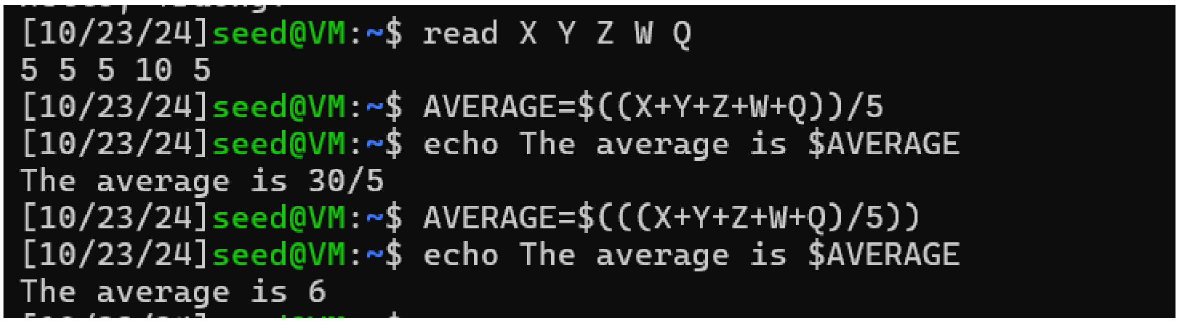
The -p option in the read command in bash is used to specify a prompt that will be displayed to the user before input is read



```
[10/23/24]seed@VM:~$ read -p "Enter your name: " NAME
Enter your name: Truong
[10/23/24]seed@VM:~$ echo "Hello, $NAME!"
Hello, Truong!
[10/23/24]seed@VM:~$
```

See that the command require user to input name and execute it

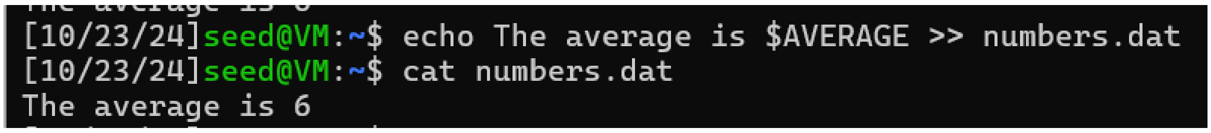
Question 33. Write a script to input five values from the user (keyboard) and output the average of the five values.



```
[10/23/24]seed@VM:~$ read X Y Z W Q
5 5 5 10 5
[10/23/24]seed@VM:~$ AVERAGE=$((X+Y+Z+W+Q))/5
[10/23/24]seed@VM:~$ echo The average is $AVERAGE
The average is 30/5
[10/23/24]seed@VM:~$ AVERAGE=$(((X+Y+Z+W+Q)/5))
[10/23/24]seed@VM:~$ echo The average is $AVERAGE
The average is 6
```

A simple command to read 5 values, then gain the funtion on it and print the sum

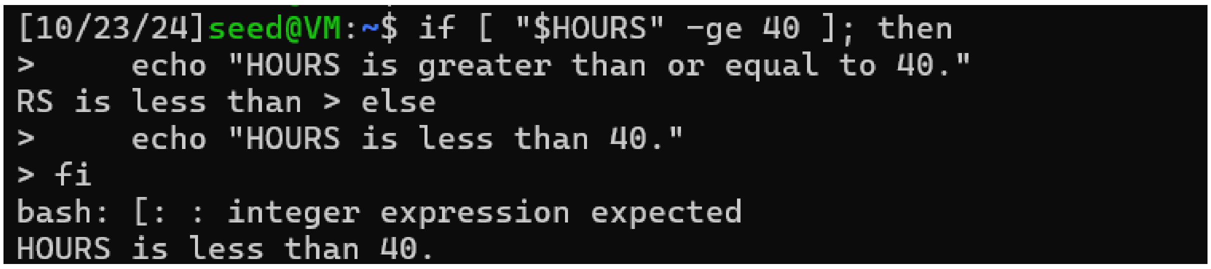
Question 34. What would you need to do so that the script from previous question would input the data from the disk file numbers.dat?



```
[10/23/24]seed@VM:~$ echo The average is $AVERAGE >> numbers.dat
[10/23/24]seed@VM:~$ cat numbers.dat
The average is 6
```

>> to gain the result in to the file and cat to print the results

Question 35. Write a conditional to test to see if HOURS is greater than or equal to 40.



```
[10/23/24]seed@VM:~$ if [ "$HOURS" -ge 40 ]; then
>     echo "HOURS is greater than or equal to 40."
RS is less than > else
>     echo "HOURS is less than 40."
> fi
bash: [: : integer expression expected
HOURS is less than 40.
```

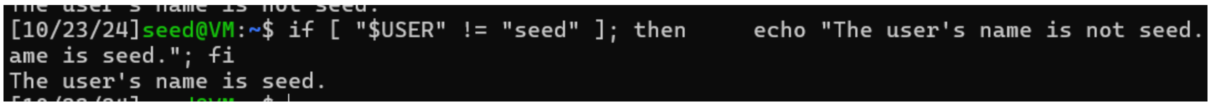
Pretty similar to final question, the command uses if to check if hours is equal to 40 or not.

if ["$HOURS" -ge 40]; then: This checks if the value of HOURS is greater than or equal to 40.

-ge: This is the operator used to compare integers for "greater than or equal to."

echo: This command prints the corresponding message based on the condition's outcome.

Question 36. Write a conditional to test to see if the user's name is not seed.



```
[10/23/24]seed@VM:~$ if [ "$USER" != "seed" ]; then     echo "The user's name is not seed.
ame is seed."; fi
The user's name is seed.
```

The command will check the user's name and bring the result, if it's not, it will return not