**EXERCICE SHELL\_4**

Table des matières

[Chapitre 1 QUESTION 0 : 2](#_Toc95075439)

[Chapitre 2 Question 1 2](#_Toc95075440)

[Chapitre 3 QUESTION 2 : 3](#_Toc95075441)

[Chapitre 4 : QUESTION 3 : 3](#_Toc95075442)

[Chapitre 5 : QUESTION 4 : 4](#_Toc95075443)

[Chapitre 6 : Question 5 : 5](#_Toc95075444)

[Chapitre 7 : QUESTION 6 : 5](#_Toc95075445)

[Chapitre 8 : QUESTION 7 : 6](#_Toc95075446)

[Chapitre 9 : QUESTION 8 : 6](#_Toc95075447)

[Chapitre 10 : QUESTION 9 : 7](#_Toc95075448)

# 

# QUESTION 0 :

00 - Tester les différentes variables spéciales suivantes :

$?

$#

$0

$n (n: valeur numérique, sauf zéro)

$!

# Question 1

Faire une fonction qui permet de calculer la moyenne arithmétique des cinq chiffres donnés en paramètre

function moy() {

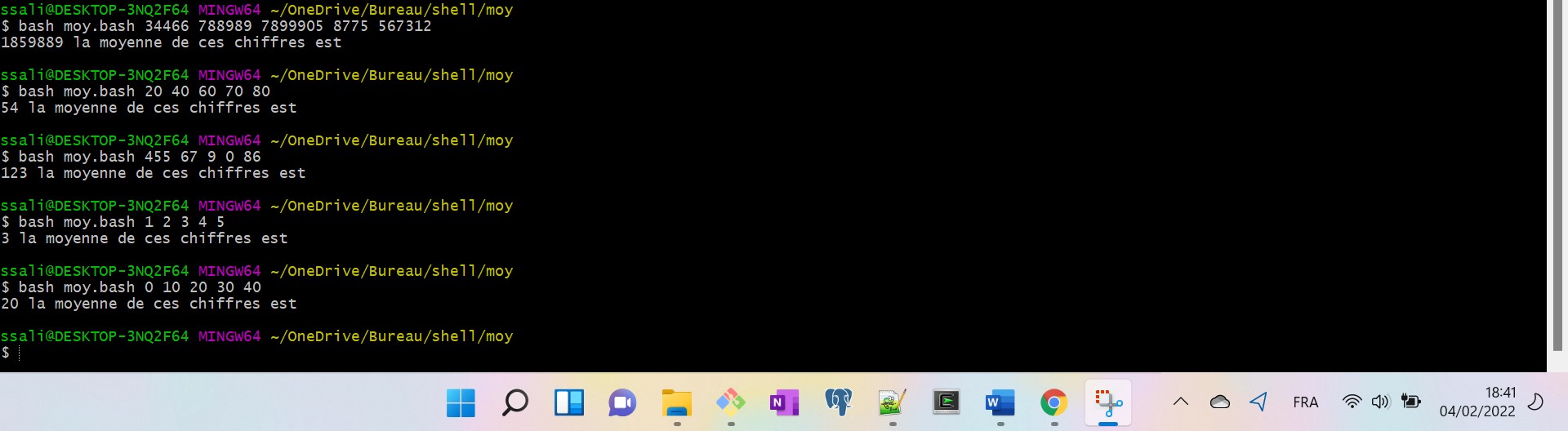
somme=$((($1+$2+$3+$4+$5)/5))

echo $somme "la moyenne de ces chiffres est"

}

var=$(moy $1 $2 $3 $4 $5)

echo $var



# QUESTION 2 :

02 - Faire une fonction qui compare deux chaines de caractères et qui retourne « chaine1 » est plus grande que « chaine2 »

function ma\_fonction() {

var1="$1"

var2="$2"

if [ ${#var1} -eq ${#var2} ]; then

echo "$1 est de meme taille que $2";

elif [ ${#var1} -gt ${#var2} ]; then

echo "$1 est plus grand que $2"

else

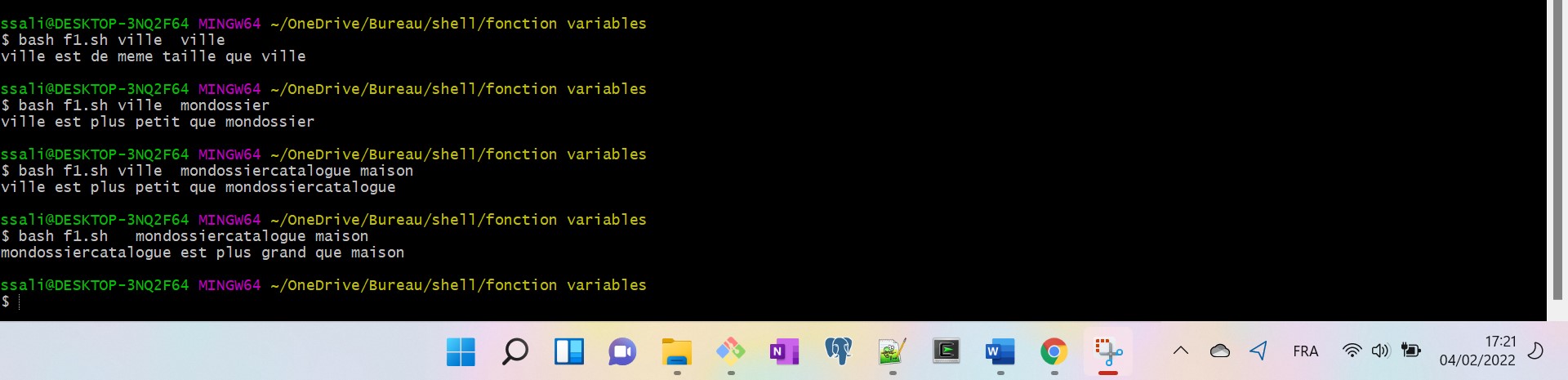
echo "$1 est plus petit que $2"

fi

}

var=$(ma\_fonction $1 $2)

echo $var



# : QUESTION 3 :

Faire un programme qui permet de conjuguer les verbes du premier groupe (ER). Le verbe à conjuguer sera passé en paramètre. Seuls les verbes en ER devront être accepte, afficher un message d’erreur sinon.

function conjugaison() {

if [[ $1 == \*"er" ]]; then

echo "je" $1 | sed 's/.$//'

echo "tu" $1 | sed 's/.$/s/'

echo "il/elle" $1 | sed 's/.$//'

echo "nous" $1 | sed 's/..$/ons/'

echo "vous" $1 | sed 's/.$/z/'

echo "ils/elles" $1 | sed 's/.$/nt/'

else

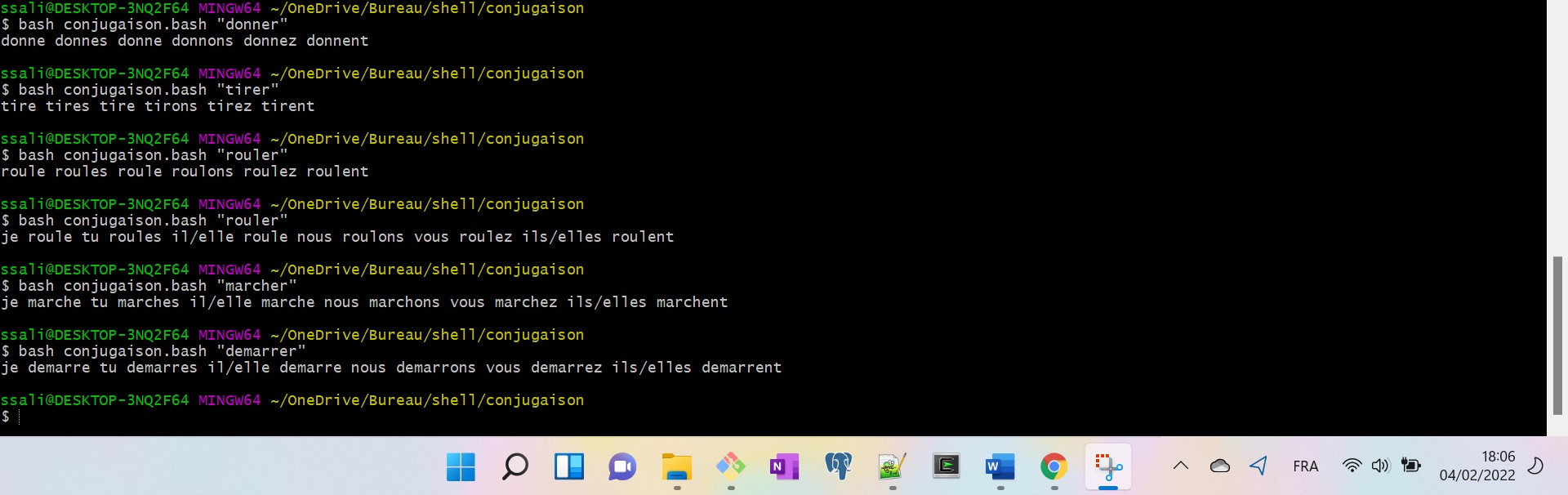
echo "verbe non correcte; veuillez entrez un verbe du premier groupe"

fi

}

var=$(conjugaison $1)

echo $var



# : QUESTION 4 :

04 – Faire une recherche d’un fichier en paramètre et affiche toute l’arborescence si présent

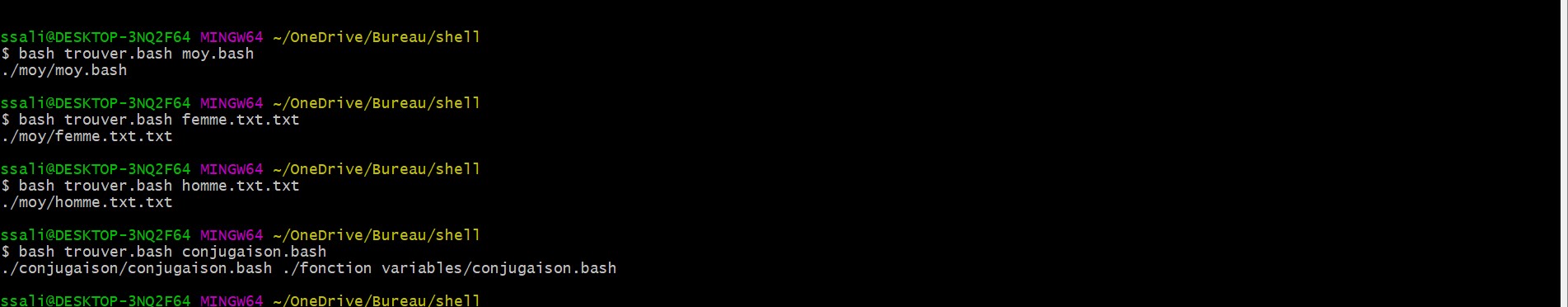
function trouver\_fichier() {

find -iname $1

}

var=$(trouver\_fichier $1)

echo $var



# : Question 5 :

- Faire une fonction qui convertit une valeur binaire (8 digits) et retourne sa valeur décimale

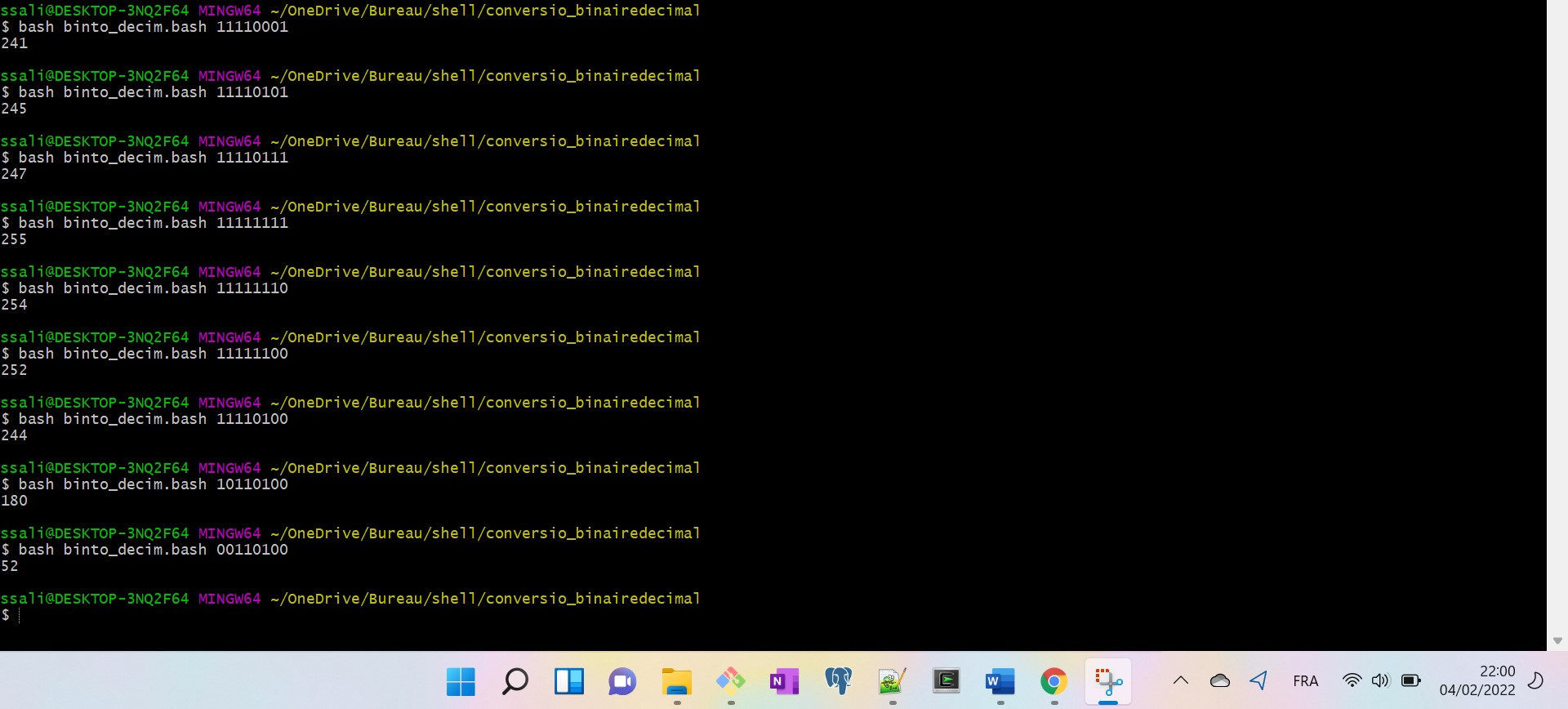
function bin\_decim() {

echo "le décimal de ce chiffre est ""$((2#$1$2$3$4$5$6$7$8))"

}

var=$(bin\_decim $1 $2 $3 $4 $5 $6 $7 $8)

echo $var



# : QUESTION 6 :

06 - Faire une fonction qui convertit une valeur décimale et retourne sa valeur binaire (8 digits).

function decim\_bin {

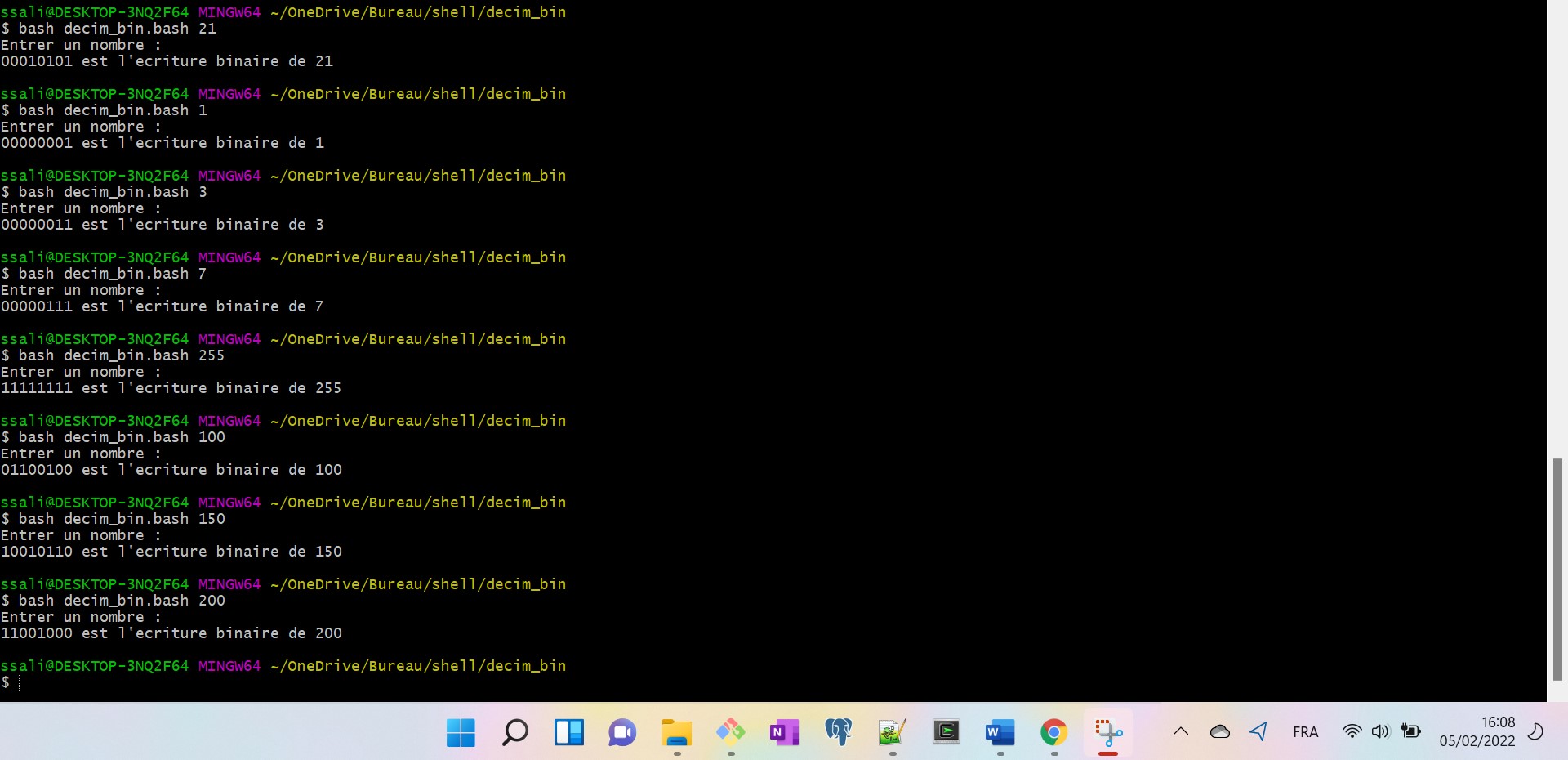
D2B=({0..1}{0..1}{0..1}{0..1}{0..1}{0..1}{0..1}{0..1});

echo ${D2B[$1]} "est l'ecriture binaire de $1"

}

var=$(decim\_bin $1)

echo $var



# : QUESTION 7 :

Faire un script qui génère un courrier de publipostage (création de plusieurs fichier texte) à partir d’une liste de contact (civilite;nom;prenom;adresse;codePostale;ville), contenu dans un fichier « contact.csv », données en paramètre.

Exemple Template :

La question je sais qu’il faut la résoudre en boucle while en définissant les champs comme l’exemple qu’on avait fait en classe mais j’ai tenté quand même la boucle for ça m’avait pris énormément de temps sans aboutir à un résultat concret.

# : QUESTION 8 :

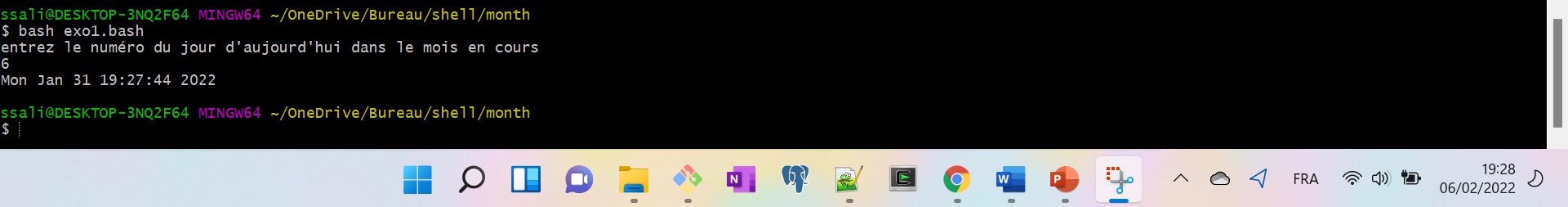
09 Faire une fonction qui retourne le dernier jour du mois passé en paramètre (format : lundi 31 janvier)

echo "entrez le numéro du jour d'aujourd'hui dans le mois en cours"

read var

monthlast\_DATE=$(date -d " +- $var days")

echo $monthlast\_DATE



# : QUESTION 9 :

10 faire une fonction qui calcule le nombre de jours ouvrés entre deux dates passées en paramètre

Question difficile car c’est trop délicat de trouver tous les jours fériés d’une année. (paque, ascension, 14 juillet, etc) donc si on ne se contente qu’aux jours de weekend l’exercice perdra en complexité

J’ai trouvé un script qui calcule le nombre de semaines

echo $(( ($(date --date="2022O101" +%s) - $(date --date="221231" +%s) )/(60\*60\*24\*7) )) 🡺Y

et un autre qui calcule le nombre de jours totaux.

echo $(( ($(date --date="2022O101" +%s) - $(date --date="221231" +%s) )/(60\*60\*24) )) 🡺X

à partir de la si on considère qu’ chaque semaine c’est deux jours de weekend

on peut faire le nomre de jours ouvrés = x-y