Exercice2-1 : fichiers

Codez une fonction **compte_mots** qui prend en entrée un nom de fichier et retourne le nombre de "mots" dans le fichier en question (séparés par des espaces).

```
# Créé par claude, le 09/11/2025 en Python 3.7

def compte_mots(nom_de_fichier):
  texte = open(nom_de_fichier, "r", encoding="utf-8").read()
  return len(texte.split())
  # fractionne le texte / à un caractère spécifique
  #ici espace
  # resultat = liste de mots
#text = "Ceci est un exemple"
#print(text.split())
# donne : ["Ceci", "est", "un", "exemple"]
nb=compte_mots("toto.txt")
print("il y a ",nb," mots dans le fichier toto.txt")
```

```
1 un deux trois quatre
2 Cinq six
3 sept
4 huit neuf
```

```
*** Remote Interpreter Reinitialized ***
il y a  9  mots dans le fichier toto.txt
```

Exercice 2-2 : lecture ecriture fichier avec des fonctions

Le fichier `Notes.txt` contient les notes reçues par des élèves à un examen de Python.

1. Ecrire une fonction **`ouvreNote()`** qui ouvre le fichier, et renvoie la liste des notes sous la forme d'une liste d'entiers.
2. Ecrire une fonction **`calculeMoyenne()`** qui calcule la moyenne du groupe d'élève
3. Ecrire une fonction **`mini()`** qui retourne la moins bonne note
4. Ecrire une fonction **`maxi()`** qui retourne la meilleure note
5. Ecrire une fonction **`admis()`** qui crée un nouveau fichier **`listeAdmis.txt`**, avec en face de chaque note la mention *admis* si la note est supérieure ou égale à 10, ou la mention *recalé* si la note est strictement inférieure à 10.

```
1 12;10;8;6;4
2 15;13;5;19;11
```

```
*** Remote Interpreter Reinitialized ***
Notes :  [12, 10, 8, 6, 4, 15, 13, 5, 19, 11]
Moyenne =  10.3
mini =  4
Maxi =  19
>>>
```

Fichier : notes.txt résultat excécution

```python
# Créé par claude, le 09/11/2025 en Python 3.7
def ouvreNote():
    try:
        # Ouvre le fichier en mode lecture
        with open("Notes.txt", "r", encoding="utf-8") as fichier:
            contenu = fichier.read()

            # Sépare les valeurs possibles par espaces, virgules ou sauts de ligne
            separateurs = [",", ";", "\n", "\t"]
            for sep in separateurs:
                contenu = contenu.replace(sep, " ")

            # Convertit chaque élément en entier
            notes = [int(x) for x in contenu.split() if x.strip().isdigit()]

            return notes

    except FileNotFoundError:
        print("Erreur : le fichier 'Notes.txt' est introuvable.")
        return []
    except ValueError:
        print("Erreur : certaines valeurs dans le fichier ne sont pas des entiers.")
        return []
```

```python
def calculeMoyenne(notes):
    if not notes:  # Vérifie si la liste est vide
        print("Erreur : la liste de notes est vide.")
        return 0.0
    return sum(notes) / len(notes)

def mini(Notes):
    if not Notes:  # Vérifie si la liste est vide
        print("Erreur : la liste des notes est vide.")
        return None
    return min(Notes)

def maxi(Notes):
    if not Notes:  # Vérifie si la liste est vide
        print("Erreur : la liste des notes est vide.")
        return None
    return max(Notes)

L_Notes =ouvreNote()
print(L_Notes)

print("Moyenne = ",calculeMoyenne(L_Notes))
print("mini = ",mini(L_Notes))
print("Maxi = ",maxi(L_Notes))
```

```python
def Fresultats(Notes) :
    with open("resultats.txt", "w", encoding="utf-8") as f:
        for note in Notes:
            statut = "admis" if note >= 10 else "recalé"
            f.write(f"{note} : {statut}\n")
    print("Fichier 'resultats.txt' créé avec succès !")

Fresultats(L_Notes)
```

```
*** Remote Interpreter Reinitialized ***
Notes :  [12, 10, 8, 6, 4, 15, 13, 5, 19, 11]
Moyenne =  10.3
mini =  4
Maxi =  19
Fichier 'resultats.txt' créé avec succès !
```

resultats.txt

```
12 : admis
10 : admis
8 : recalé
6 : recalé
4 : recalé
15 : admis
13 : admis
5 : recalé
19 : admis
11 : admis
```

Exercice 2-3 : modification du contenu d'un fichier

Écrivez un script qui permette de créer et de relire aisément un fichier texte.

Votre programme demandera d'abord à l'utilisateur d'entrer le nom du fichier. Ensuite il lui proposera le choix, soit d'enregistrer de nouvelles lignes de texte, soit d'afficher le contenu du fichier.

```
# Créé par claude, le 09/11/2025 en Python 3.7
def sansDC(ch):
    "cette fonction renvoie la chaîne ch diminuée de son dernier caractère"
    nouv = ""
    i, j = 0, len(ch) - 1
    while i < j:
        nouv = nouv + ch[i]
        i = i + 1
    return nouv

def ecrireDansFichier():
    of = open(nomF, 'a')
    while 1:
        ligne = input("entrez une ligne de texte (ou <Enter>) : ")
        if ligne == '':
            break
        else:
            of.write(ligne + '\n')
    of.close()

def lireDansFichier():
    of = open(nomF, 'r')
    while 1:
        ligne = of.readline()
        if ligne == "":
            break
        # afficher en omettant le dernier caractère (= fin de ligne)
        print(sansDC(ligne))
    of.close()
```

```
nomF = input('Nom du fichier à traiter : ')
choix = input('Entrez "e" pour écrire, "c" pour consulter les données : ')

if choix =='e':
    ecrireDansFichier()
else:
    lireDansFichier()
```

Exercice 2-4 : utilisation de la biblio TKINTER

Pré requis test des fonctions simple du doc : Module TKINTER

a) Créez un court programme qui dessinera les 5 anneaux olympiques dans un rectangle de fond blanc. Un bouton « Quitter » doit permettre de fermer la fenêtre.

```
# Créé par claude, le 09/11/2025 en Python 3.7
from tkinter import *

# Coordonnées X,Y des 5 anneaux :
coord = [[20,30], [120,30], [220, 30], [70,80], [170,80]]
# Couleurs des 5 anneaux :
coul = ["red", "yellow", "blue", "green", "black"]

base = Tk()
can = Canvas(base, width =335, height =200, bg ="white")
can.pack()
bou = Button(base, text ="Quitter", command =base.quit)
bou.pack(side = RIGHT)
# Dessin des 5 anneaux :
i = 0
while i < 5:
  x1, y1 = coord[i][0], coord[i][1]
  can.create_oval(x1, y1, x1+100, y1 +100, width =2, outline =coul[i])
  i = i + 1
base.mainloop()
```

b) Modifiez le programme ci-dessus en y ajoutant 5 boutons. Chacun de ces boutons provoquera le tracé de chacun des 5 anneaux.

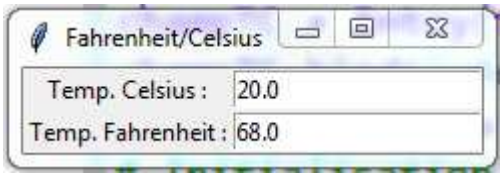
```python
# Créé par claude, le 09/11/2025 en Python 3.7
import tkinter as tk

# Dessin des 5 anneaux :
def dessineCercle(i):
  x1, y1 = coord[i][0], coord[i][1]
  can.create_oval(x1, y1, x1+100, y1 +100, width =2, outline =coul[i])


# Coordonnées X,Y des 5 anneaux :
coord = [[20,30], [120,30], [220, 30], [70,80], [170,80]]
# Couleurs des 5 anneaux :
coul = ["red", "yellow", "blue", "green", "black"]

base = tk.Tk()
can = tk.Canvas(base, width =335, height =200, bg ="white")
can.pack()
bouton = tk.Button(base, text ="Quitter", command =base.quit)
bouton.pack(side = tk.RIGHT)

# Installation des 5 boutons :
# On utilise la fonction lambda afin de pouvoir passer un argument à la fonction
for i in range(5):
    tk.Button(base, text=str(i+1),
              command = lambda n=i: dessineCercle(n)).pack(side =tk.LEFT)
base.mainloop()
```

Exercice 2-5 : *interaction et réaction liées dans une fenêtre créée par TKINTER*

Écrivez un petit programme qui fait apparaitre une fenêtre avec deux champs : l'un indique une température en degrés Celsius, et l'autre la même température exprimée en degrés Fahrenheit. Chaque fois que l'on change une quelconque des deux températures, l'autre est corrigée en conséquence.

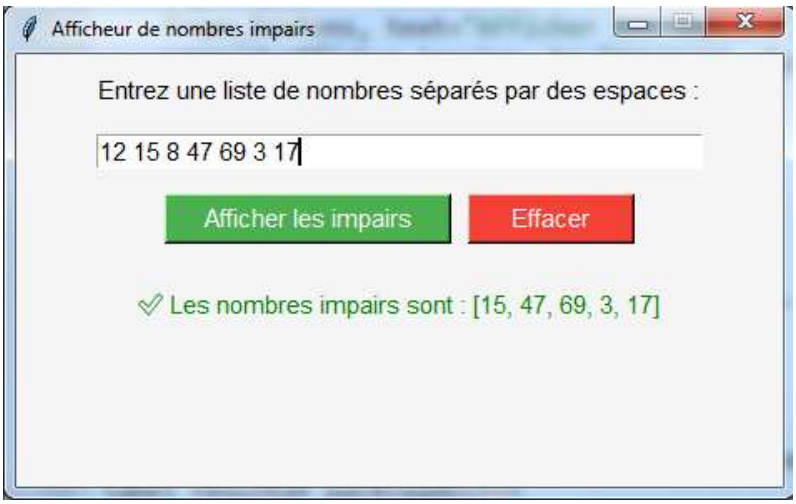
Pour convertir les degrés Fahrenheit en Celsius et vice-versa, on utilise la formule TF=TC * 1,80 + 32

```
# Créé par claude, le 09/11/2025 en Python 3.7
# Conversions de températures Fahrenheit <=> Celsius
from tkinter import *

def convFar(event):
    "valeur de cette température, exprimée en degrés Fahrenheit"
    tF = eval(champTC.get())
    varTF.set(str(tF*1.8 +32))

def convCel(event):
    "valeur de cette température, exprimée en degrés Celsius"
    tC = eval(champTF.get())
    varTC.set(str((tC-32)/1.8))

fen = Tk()
fen.title('Fahrenheit/Celsius')

Label(fen, text='Temp. Celsius :').grid(row =0, column =0)
# "variable tkinter" associée au champ d'entrée. Cet "objet-variable"
# assure l'interface entre TCL et Python (voir notes, page 165) :
varTC =StringVar()
champTC = Entry(fen, textvariable =varTC)
champTC.bind("<Return>", convFar)
champTC.grid(row =0, column =1)
# Initialisation du contenu de la variable tkinter :
varTC.set("100.0")

Label(fen, text='Temp. Fahrenheit :').grid(row =1, column =0)
varTF =StringVar()
champTF = Entry(fen, textvariable =varTF)
champTF.bind("<Return>", convCel)
champTF.grid(row =1, column =1)
varTF.set("212.0")
fen.mainloop()
```



Exercice 2-6 :

Reprenez l'exercice qui extrait les nombres impairs d'une liste et intégrez le dans une fenêtre graphique avec zone de saisie, bouton de traitement et bouton d'effacement.




```python
import tkinter as tk

def afficher_impairs():
    """Affiche les nombres impairs saisis par l'utilisateur."""
    entree = champ_saisie.get()
    try:
        # Conversion de la saisie en liste d'entiers
        nombres = [int(x) for x in entree.split()]
        impairs = [n for n in nombres if n % 2 != 0]

        # Mise à jour de l'affichage
        if impairs:
            label_resultat.config(text=f"□ Les nombres impairs sont : {impairs}", fg="green")
        else:
            label_resultat.config(text="□ Aucun nombre impair trouvé.", fg="orange")
    except ValueError:
        label_resultat.config(text="□ Veuillez entrer uniquement des nombres séparés par des espaces.", fg="red")
```

```python
def effacer():
    """Efface la saisie et le résultat."""
    champ_saisie.delete(0, tk.END)
    label_resultat.config(text="", fg="black")

# --- Interface graphique ---
fenetre = tk.Tk()
fenetre.title("Afficheur de nombres impairs")
fenetre.geometry("460x260")
fenetre.resizable(False, False)
fenetre.config(bg="#F5F5F5")

# Label d'instruction
label = tk.Label(
    fenetre,
    text="Entrez une liste de nombres séparés par des espaces :",
    font=("Arial", 11),
    bg="#F5F5F5"
)
label.pack(pady=10)
```

```
# Champ de saisie
champ_saisie = tk.Entry(fenetre, width=45, font=("Arial", 11))
champ_saisie.pack(pady=5)

# Cadre pour les boutons
frame_boutons = tk.Frame(fenetre, bg="#F5F5F5")
frame_boutons.pack(pady=10)

# Bouton "Afficher"
bouton_afficher = tk.Button(
    frame_boutons, text="Afficher les impairs",
    command=afficher_impairs, bg="#4CAF50", fg="white", font=("Arial", 11), width=18
)
bouton_afficher.grid(row=0, column=0, padx=5)

# Bouton "Effacer"
bouton_effacer = tk.Button(
    frame_boutons, text="Effacer",
    command=effacer, bg="#f44336", fg="white", font=("Arial", 11), width=10
)
bouton_effacer.grid(row=0, column=1, padx=5)

# Label de résultat
label_resultat = tk.Label(fenetre, text="", font=("Arial", 11), bg="#F5F5F5")
label_resultat.pack(pady=15)

# Boucle principale
fenetre.mainloop()
```

A l'aide de Pyinstaller on va pouvoir générer un .exe directement utilisable sans passer par python.

Ouvrir une fenêtre de commande :

pip install pyinstaller

ouvrir une fenêtre console dans le répertoire où se trouve le code Python :

pyinstaller --noconsole --onefile nombres_impairs.py (nom du script)

Une fois la commande terminée, PyInstaller crée deux dossiers :

- `build/` → fichiers temporaires
- `dist/` → ton **exécutable final**

On peut ajouter une **icône personnalisée** (fichier `.ico`) :

pyinstaller --noconsole --onefile --icon=mon_icone.ico nombres_impairs.py

Exemple sup :

Lecture fichier notesB .csv affichage des notes

```
# Créé par claude, le 01/12/2025 en Python 3.7
import csv

# Nom du fichier CSV
fichier = "notesB.csv"

try:
    with open(fichier, newline='', encoding='utf-8') as csvfile:
        lecteur = csv.reader(csvfile)

        # Lecture de l'en-tête
        header = next(lecteur)
        print("En-tête :", header)

        print("\nContenu du fichier :")
        for ligne in lecteur:
            nom, note = ligne
            print(f"{nom} : {note}")

except FileNotFoundError:
    print(f"Erreur : le fichier '{fichier}' est introuvable.")
except Exception as e:
    print("Erreur lors de la lecture du fichier :", e)
```