Exercice 2.8 - Solution :

```
import numpy as np
import os
# a) Définir une fonction qui crée un fichier « data1.txt » avec 10 lignes : 00 , 01 , . . . , 99

# Méthode-1
def create_fic_int(ficname, n_lines, n_colonnes):
    file = open(ficname, "w")
    for i in range(n_lines):
        line = ""
        for j in range(n_colonnes):
            line = line + "{}".format(j+10*i)
            if (j < n_colonnes-1):
                line = line + " , "
            else:
                line = line + " \n"
        file.write(line)

    file.close()
```

Exercice 2.8 - Solution :

```
# Méthode-2  : Créer d'abord la liste de TOUTES les lignes

def create_fic_int(ficname, n_lines, n_colonnes):
    file = open(ficname, 'w’)

    Liste = [ [ i + j*10 for i in range(n_colonnes) ] for j in range(n_lines) ]
    for line in Liste:
        for i in range(len(line)):
            file.write(str(line[i]))
            if i < (n_colonnes-1):
                file.write(',')
            else:
                file.write('\n’)

    file.close()
```

Exercice 2.8 - Solution :

```
# Méthode-3 : utiliser .join(liste) pour convertir une liste en une chaine

def create_fic_int(ficname, n_lines, n_colonnes):
    file = open(ficname, 'w’)

    Liste = [ [ i + j*10 for i in range(n_colonnes) ] for j in range(n_lines) ]
    for line in Liste:
        line = np.array(line, dtype=str)     # dtype=str car .join() fonctionne sur des strings
        Line_str = ","
        Line_str = Line_str.join(line)
        file.write( Line_str+'\n’ )

    file.close()
```

Exercice 2.8 - Solution :

```
# Méthode-4
def create_fic_int(ficname, n_lines, n_colonnes):

    file = open(ficname, "w")
    for i in range(n_lines):
        line = ""
        for j in range(n_colonnes):
            line = line + "{}{}".format(i,j)
            if (j < n_colonnes-1):
                line = line + " , "
            else:
                line = line + " \n"
        file.write(line)

    file.close()

create_fic_int("data1.txt", 10, 10)
```

Exercice 2.8 - Solution :

```
# Charger un fichier en liste Numpy,  si nécessaire, positionner le cwd (Current Working Directory)
cwd = os.getcwd()
print ( cwd )
os.chdir('C:\\Users\\uname\\test')

b) Charger un fichier dans une matrice Numpy

data1 = np.loadtxt("data1.txt", delimiter=',')

c) Afficher la matrice data1 et ses dimensions

print( data1.shape )
print( data1 )

print (data1[:8,:])      # d) Affiche les 8 premières lignes de data1
print (data1[:8,0])      # e) Affiche les 8 premières lignes de la colonne 0 de data1
print (data1[:8,1])      # f) Affiche les 8 premières lignes de la colonne 1 de data1
```

Exercice 2.8 - Solution :

```
# g) Définir une fonction create_fic_str ():
#      Entrées : fic_name, n_lignes, n_colonnes   ,
#      Sortie   : crée un fichier fic_name de chaines avec  n_lignes et n_colonnes
# Méthode-1
def create_fic_str(ficname, n_lines, n_colonnes):
    file = open(ficname, "w")
    Lettres = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J']
    for i in range(n_lines):
        line = ""
        c = Lettres[i]
        for j in range(n_colonnes):
                line = line + c+str(j+1)
                if (j < n_colonnes-1):
                        line = line + " , "
                else:
                        line = line + " \n"
        file.write(line)
    file.close()
```

Exercice 2.8 - Solution :

```
# Méthode-2  : Créer d'abord la liste de TOUTES les lignes

def create_fic_str(ficname, n_lines, n_colonnes):
    file = open(ficname, 'w')
    code = ord('A')
    Liste = [ [ "{}{}".format(chr(code+j), i+1) for i in range(n_colonnes) ] for j in range (n_lines) ]
    for line in Liste:
        for i in range(len(line)):
            file.write(line[i])
            if i < (n_colonnes-1):
                file.write(',')
            else:
                file.write('\n')

    file.close()
```

Exercice 2.8 - Solution :

```
# Méthode-3 : utiliser .join(liste) pour convertir une liste en une chaine

def create_fic_str(ficname, n_lines, n_colonnes):
    file = open(ficname, 'w')
    code = ord('A')
    Liste = [ [ "{}{}".format(chr(code+j), i+1) for i in range(n_colonnes) ] for j in range (n_lines) ]
    for line in Liste:
        Line_str = ","
        Line_str = Line_str.join(line)
        file.write( Line_str+'\n' )

    file.close()
```

Exercice 2.8 - Solution :

 # Méthode-4

```
def create_fic_str(ficname, n_lines, n_colonnes):

    file = open(ficname, "w")
    code = ord('A')
    for i in range(n_lines):
        line = ""
        lettre = chr(code+i)
        for j in range(n_colonnes):
                line = line + lettre+str(j+1)
                if (j < n_colonnes-1):
                        line = line + " , "
                else:
                        line = line + " \n"
        file.write(line)

    file.close()
```

Exercice 2.8 - Solution :

```
#    Appeler la fonction avec fic_name = ‘data2.txt’, et n_lignes = 10, et n_colonnes = 10
create_fic_str("data2.txt", 10, 10)

# h) Charger un fichier dans une matrice Numpy, si nécessaire, positionner le cwd (Current Working Directory)
cwd = os.getcwd()
print ( cwd )
os.chdir('C:\\Users\\uname\\test')

data2 = np.loadtxt("data2.txt", delimiter=',', dtype=str)

i) Afficher la matrice data2 et ses dimensions
print( data2.shape )
print( data2 )

print(data2[0,:8])      # j) Affiche les 8 premières colonnes de la ligne 1ère ligne
print(data2[1,:8])      # k) Affiche les 8 premières colonnes de la ligne 2ème ligne
print(data2[2,1:3])     # l) Affiche la 2ème et 3ème colonnes de la 3ème ligne
```