EXERCICE 1

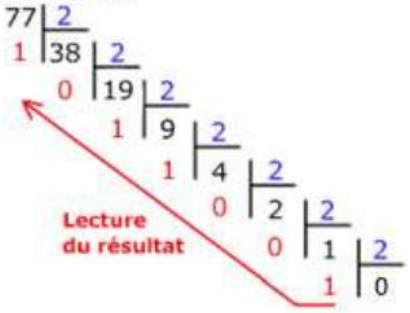
Écrire une fonction Python appelée `nb_repetitions` qui prend en paramètres un élément `elt` et un tableau `tab` (type `list`) d'éléments du même type et qui renvoie le nombre de fois où l'élément apparaît dans le tableau.

Exemples :

```
>>> nb_repetitions(5, [2, 5, 3, 5, 6, 9, 5])
3
>>> nb_repetitions('A', ['B', 'A', 'B', 'A', 'R'])
2
>>> nb_repetitions(12, [1, '!', 7, 21, 36, 44])
0
```

EXERCICE 2

Pour rappel, la conversion d'un nombre entier positif en binaire peut s'effectuer à l'aide des divisions successives comme illustré ici :



Voici une fonction Python basée sur la méthode des divisions successives permettant de convertir un nombre entier positif en binaire :

Compléter la fonction `binaire`.

```
def binaire(a):
    '''convertit un nombre entier a en sa representation
    binaire sous forme de chaine de caractères.'''
```

Exemples :

```
>>> binaire(0)
'0'
>>> binaire(77)
'1001101'
```

EXERCICE 3

Écrire une fonction `recherche_indices_classement` qui prend en paramètres un entier `elt` et un tableau d'entiers `tab` représenté par une liste Python, et qui renvoie trois listes Python d'entiers:

- la première liste contient les indices des valeurs du tableau `tab` strictement inférieures à `elt` ;
- la deuxième liste contient les indices des valeurs du tableau `tab` égales à `elt` ;
- la troisième liste contient les indices des valeurs du tableau `tab` strictement supérieures à `elt`.

Exemples :

```
>>> recherche_indices_classement(3, [1, 3, 4, 2, 4, 6, 3, 0])
([0, 3, 7], [1, 6], [2, 4, 5])
>>> recherche_indices_classement(3, [1, 4, 2, 4, 6, 0])
([0, 2, 5], [], [1, 3, 4])
>>>recherche_indices_classement(3, [1, 1, 1, 1])
([0, 1, 2, 3], [], [])
>>> recherche_indices_classement(3, [])
([], [], [])
```

EXERCICE 4

Une professeure de NSI décide de gérer les résultats de sa classe sous la forme d'un dictionnaire :

- les clefs sont les noms des élèves ;
- les valeurs sont des dictionnaires dont les clefs sont les types d'épreuves sous forme de chaîne de caractères et les valeurs sont les notes obtenues associées à leurs coefficients dans une liste.

Avec :

```
resultats = {
    'Dupont': {
        'DS1': [15.5, 4],
        'DM1': [14.5, 1],
        'DS2': [13, 4],
        'PROJET1': [16, 3],
        'DS3': [14, 4]
    },
    'Durand': {
        'DS1': [6 , 4],
        'DS2': [8, 4],
        'PROJET1': [9, 3],
        'IE1': [7, 2],
        'DS3': [12, 4]
    }
}
```

L'élève dont le nom est Durand a ainsi obtenu au DS2 la note de 8 avec un coefficient 4.

La professeure crée une fonction moyenne qui prend en paramètre le nom d'un de ses élèves et renvoie sa moyenne arrondie au dixième. Si l'élève n'a pas de notes, on considère que sa moyenne est nulle. Si le nom donné n'est pas dans les résultats, la fonction renvoie None.

Compléter le code de la professeure ci-dessous :

```
def moyenne(nom, resultats):
    '''Renvoie la moyenne de l'élève nom, selon le dictionnaire
    resultats. Si nom n'est pas dans le dictionnaire,
    la fonction renvoie None.'''
```

Exemples :

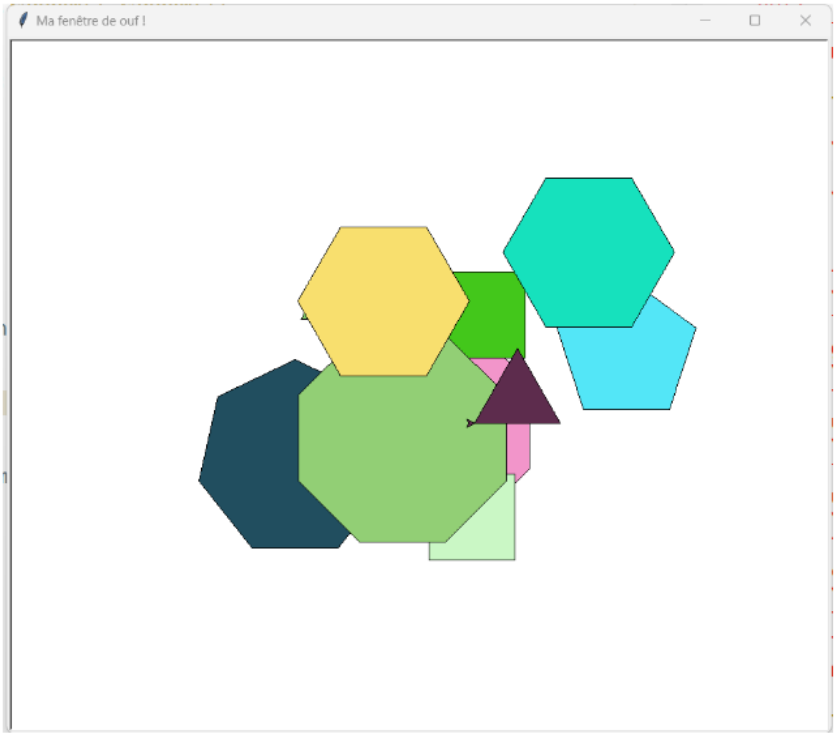
```
>>> moyenne("Dupont", resultats)
14.5
>>> moyenne("Durand", resultats)
8.5
```

EXERCICE 5

AVEC TURTLE :

⇨ Ecrire un code qui permette de dessiner 10 figures dont la forme polygonale, la position et les couleurs sont définies aléatoirement :

- la couleur en utilisant la fonction random()
- la forme en utilisant la fonction randint(3,8) qui définira le nombre de côté du polygone. La longueur des cotés sera de 100 px.
- Les coordonnées du 1er point sont définies avec la fonction randint(-200,200)



Info intéressante : Pour un polygone de n coté, la tortue devra tourner d'un angle de $\frac{360}{n}$ degrés .