

## 1. Code qui compte le nombre de caractères d'un string – Mode Débogage de Pyzo :

⇒ Enregistrer le code ci-contre dans un fichier nommé `ex_1b.py` :

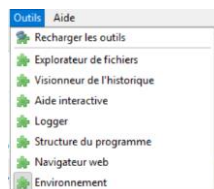
⇒ Que retourne la fonction `nb_caractere` après exécution ?

```

exercice_1.py
1 # Fonctions
2 def nb_caracteres(chaine) :
3     nb = 0
4     for l in chaine :
5         nb = nb + 1
6     return nb
7
8 # Main
9 n = nb_caracteres("Vacances")
10 print(n)


```

⇒ Afficher la fenêtre *Environnement*



de pyzo :

⇒ Insérer un point d'arrêt en cliquant ici .

⇒ Exécuter alors le script (*F5 clavier*) et cliquer sur  pour l'exécuter ligne par ligne et **compléter ci-contre le tableau** (à reproduire), **donnant les valeurs des variables** étape par étape :

The screenshot shows the Pyzo IDE with the code from the previous block. A red dot indicates a breakpoint is set at line 3, `nb = 0`. The command prompt shows the command `>>> (executing file "exercice_1.py")` and the function call `(nb_caracteres)>>>`. The Environment window is open, showing the current state of variables:

Nom	Type	Repr
chaine	str	'Vacances'
n	int	8
nb_caracteres	function	<function nb_caracteres at 0x000001C86>

nb	chaine	l
0	Vacances	V
1	Vacances	aa
2	Vacances	c

**AIDE : La fenêtre *Environnement* affiche la valeur des différentes variables à chacune des étapes de l'exécution.**


2. Code qui inverse l'ordre des lettres dans une chaîne de caractère :

⇒ Enregistrer le code ci-contre dans un fichier nommé *ex\_2b.py* :

```

1  # Fonctions
2  def nb_caracteres(chaine) :
3      a_l_envers = ""
4      for l in chaine :
5          a_l_envers = l + a_l_envers
6      return a_l_envers
7
8  # Main
9  mot_1 = nb_caracteres("Nadal")
10 mot_2 = nb_caracteres(mot_1)
11 print(mot_1)
12 print(mot_2)

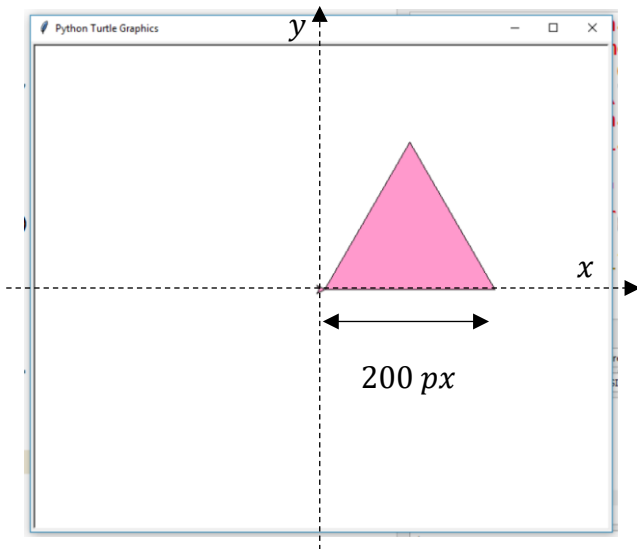
```

⇒ Réaliser une exécution normale (F5) ou étape par étape (avec point d'arrêt et ) et compléter le tableau ci-dessous, donnant la valeur des variables au cours de l'exécution : **s'aider ou pas de la fenêtre Environnement qui affiche le contenu des variables ....**

	<i>mot_1</i>	<i>mot_2</i>	<i>chaine</i>	<i>a_l_envers</i>	<i>l</i>
<i>Etape 1</i>			<i>Nadal</i>		<i>N</i>
<i>Etape 2</i>			<i>Nadal</i>	<i>N</i>	<i>a</i>
<i>Etape 3</i>			<i>Nadal</i>	<i>aN</i>	
<i>Etape 4</i>			<i>Nadal</i>		
<i>Etape 5</i>			<i>Nadal</i>		
<i>Etape 6</i>			<i>Nadal</i>		
<i>Etape 7</i>					
<i>Etape 8</i>					
<i>Etape 9</i>					
<i>Etape 10</i>					
<i>Etape 11</i>					
<i>Etape 12</i>					
<i>Etape 13</i>					

### 3 Code qui trace un triangle équilatéral en couleur :

L'exécution du code ci-contre donne le tracé ci-dessous (le repère a été rajouté, il n'est pas tracé) :



```
1 from turtle import *
2
3 # Fonctions
4 def triangle(a,x,y,r,g,b) :
5     [redacted]
6     [redacted]
7     fillcolor(r,g,b)
8     [redacted]
9     [redacted]
10    [redacted]
11    [redacted]
12    [redacted]
13    [redacted]
14    [redacted]
15 # Main
16 triangle(200,0,0,1,0.6,0.8)
17
18 exitonclick()
```

⇒ Compléter ce code afin d'obtenir le même résultat.

(INFO) : si  $r = 1$  et  $g = b = 0$ , on a du rouge

si  $r = b = 0$  et  $g = 1$ , on a du green

si  $r = g = 0$  et  $b = 1$ , on a du blue)

### 4 Code qui trace plusieurs triangles roses

⇒ Faire un copier/coller de la fonction triangle dans un nouveau fichier *ex\_4b.py*.

Compléter alors ce code en utilisant cette fonction, afin d'obtenir le dessin ci-dessous à son exécution :

