1. Code à comprendre :

Que donne l'exécution du script suivant :

```
1  def inutile(a) :
2      for i in range(10) :
3          mot = "DS"
4      return mot
5
6  b = inutile(2020)
7  print(b)
```

```
>>> (executing file "exercice1.py")
DS
```

2. Code qui retourne le max de 2 nombres :

La fonction *maxi()* retourne le maximum des 2 nombres mis en argument. Compléter ci-dessous le code de cette fonction :

```
1  # Fonctions
2  def maxi(a,b) :
3      if a > b :
4          return a
5      else :
6          return b
7
8
9  # Programme principal
10 c = maxi(80,-5)
11 print(c)
12 print(maxi(-5, 80))
```

L'exécution de ce code donne :

```
>>> (executing file "exercice2.py")
80
80
```

3. Code qui échange 2 lettres d'une phrase :

La fonction *echange()* définit la page 2, prend en argument 3 chaines de caractères : une phrase et 2 lettres. Elle retourne cette même phrase en ayant **échangé** les caractères identiques à la 1ère lettre par la 2nd lettre :

Ainsi, en exécutant cette fonction comme cela dans le programme principal :

```
16 # Programme principal
17 first_phrase = echange_lettre("le coronavirus","a","e")
18 print(first_phrase)
19
20 next_phrase = echange_lettre("opéra","a","o")
21 print(next_phrase)
```

on obtient dans le shell :

```
>>> (executing file "exercice2.py")
la coronevirus
apéro
```

3.1. Compléter le script de la fonction *echange()* :

```
1  # Fonctions
2  def echange_lettre(phrase,lettre_1,lettre_2) :
3      phrase_modifiee =""
4
5      for l in phrase :
6          if l == lettre_1 :
7              phrase_modifiee = phrase_modifiee + lettre_2
8          elif l == lettre_2 :
9              phrase_modifiee = phrase_modifiee + lettre_1
10         else :
11             phrase_modifiee = phrase_modifiee + l
12
13     return phrase_modifiee
14
```

3.2. Valeurs prises par les variables :

Pour la seconde exécution : `next_phrase = echange_lettre("opéra","a","o")`

Compléter le tableau ci-dessous qui donne les valeurs des différentes variables utilisées, au cours de cette exécution.

phrase	*lettre_1*	*lettre_2*	*l*	*phrase_modifiee*	*next_phrase*
''opéra''	''a''	''o''	''o''	'' '' + ''a'' = ''a''	
			''p''	''a'' + ''p'' = ''ap''	
			''é''	''ap'' + ''é'' = ''apé''	
			''r''	''apé'' + ''r'' = ''apér''	
			''a''	''apér'' + ''o'' = ''apéro''	''apéro''

4. Fonction qui compare 2 nombres :

```
# Fonctions
def compare(a,b) :
    if a == b :
        message = str(a) + " = " + str(b)
    elif a < b :
        message = str(a) + " < " + str(b)
    else :
        message = str(a) + " > " + str(b)
    return message


# Programme principal
c = compare(80,-5)
print(c)
print(compare(-5,80))
print(compare(-5,-5))
```

L'exécution du script précédent donne :

⇨ Compléter ce script.

(< , = , > , sont des caractères comme les autres)

```
>>> (executing file "exercice2.py")
80 > -5
-5 < 80
-5 = -5
```

5. Code qui donne des punitions au lycée :

```
# fonctions
def punition(fait,recidive):
    if fait == "bavardages" :
        peine = 2
    elif fait == "insultes" :
        peine = 3
    elif fait == "portable" :
        peine = 4

    if recidive == "oui" :
        peine = peine * 2
        return fait+" et récidive : "+str(peine)+" heures de colles"
    else :
        return fait+" : "+str(peine)+" heures de colles"
```

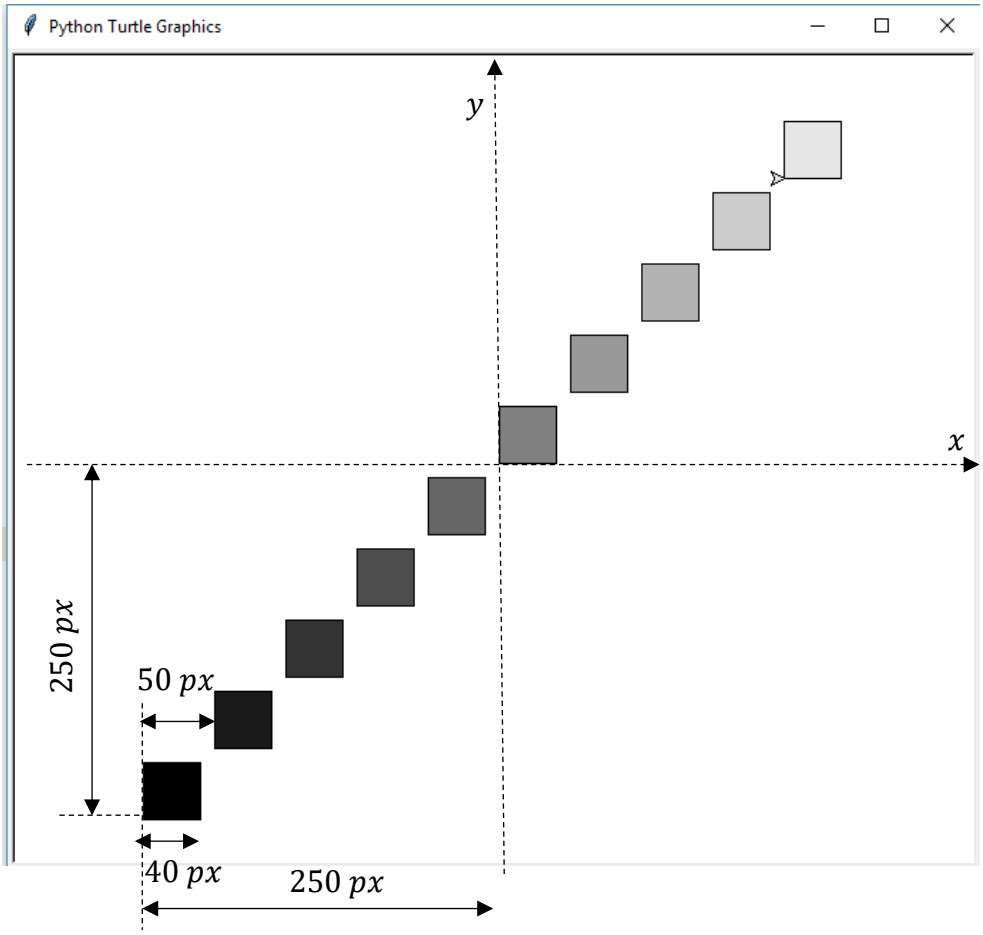
```
# main
message1 = punition("bavardages","non")
print(message1)

message2 = punition("bavardages","oui")
print(message2)

message3 = punition("insultes","non")
print(message3)

message4 = punition("insultes","oui")
print(message4)

message5 = punition("portable","non")
print(message5)

message6 = punition("portable","oui")
print(message6)
```

Le code précédent permet d'obtenir dans le shell cela :

⇨ Compléter ce code.

```
>>> (executing file "exercice5.py")
bavardages : 2 heures de colles
bavardages et récidive : 4 heures de colles
insultes : 3 heures de colles
insultes et récidive : 6 heures de colles
portable : 4 heures de colles
portable et récidive : 8 heures de colles
```

6. Code qui trace des carrés :



En exécutant le code qui suit et qui est incomplet, on obtient la fenêtre graphique ci-dessus.

⇨ Compléter ce code en utilisant une boucle for … in range() si possible.

```
1  from turtle import *
2
3  # Fonctions
4  def carre(a,x,y,r,g,b) :
5      up()
6      goto(x,y)
7      fillcolor(r,g,b)
8      down()
9      begin_fill()
10     for i in range(4) :
11         forward(a)
12         left(90)
13     end_fill()
14
15 # Main
16
17 for i in range(10) :
18     c = i * 0.1
19     x = -250 + 50*i
20     y = -250 + 50*i
21     carre(40,x,y,c,c,c)
22
23 exitonclick()
```