Feuille Info_1 : Les notions de base de Python

Attention : certain programme donne des erreurs à l'exécution, vous devez l'indiquer par **"Erreur"**.

Exercice 1 : (∗)

```
for i in range(3):
    print("Bonjour ", i)
```

```
>>> # script executed
```

Exercice 2 : (∗)

```
a = 2+3*4
print(a)
```

```
>>> # script executed
```

Exercice 3 : (∗)

```
for k in range(2):
    print(k)
```

```
>>> # script executed
```

Exercice 4 : (∗)

```
def carre(n):
    return n * n

print(carre(5))
```

```
>>> # script executed
```

Exercice 5 : (∗)

```
t = 'Berthelot'
U = t.upper()
n = 0
for c in U:
    if c == 'T':
        n = n+1
print(n)
```

```
>>> # script executed
```

Exercice 6 : (∗)

```
t = 'Berthelot'
U = t.upper()
n = 0
for c in U:
    if c == 'T':
        n = n+1
        print(n)
```

```
>>> # script executed
```

Exercice 7 : (∗)

```
t = 'Berthelot'
U = t.upper()
n = 0
for c in U:
    if c == T:
        n = n+1
print(n)
```

```
>>> # script executed
```

Exercice 8 : (∗)

```
L = [1, 3]
for x in L:
    L.append(x)
print(L)
```

```
>>> # script executed
```

Exercice 9 : (∗)

```
a, b = 2, 3
if a < b:
    print("Affichage 1")
else:
    print("Affichage 2")
```

```
>>> # script executed
```

Exercice 10 : (∗)

```
def f(x):
     return x**x

n = 1
while f(n) < 10:
   n = n + 1
print(n)
```

```
>>> # script executed
```

Exercice 11 : (∗)

```
k = 0
while k < 10:
   k = k + 3
   print(k)
```

```
>>> # script executed
```

Exercice 12 : (∗)

```
L = [1, 2]
while len(L) < 4:
   L.append(0)
print(L)
```

```
>>> # script executed
```

Exercice 13 : (∗)

```
x = 5
y = x
x = x + 1
print(y)
```

```
>>> # script executed
```

Exercice 14 : (∗)

```
def mystere(n):
    if n % 2 == 0:
        return "Pair"
    else:
        return "Impair"

print(mystere(7))
```

```
>>> # script executed
```

Exercice 15 : (∗)

```
L = [0, 1, 2]
for i in range(len(L)):
    L[i] = L[i] + 1

print(L)
```

```
>>> # script executed
```

Exercice 16 : (∗)

```
s = "abcdef"
print(s[3])
```

```
>>> # script executed
```

Exercice 17 : (∗∗)

```
n = 5
fact = 1
while n > 1:
    fact = fact * n
    n = n - 1
print(fact)
```

```
>>> # script executed
```

Exercice 18 : (∗∗)

```
i = 0
s = 0
while i < 4:
    s = s + i
    i = i + 1
print(s)
```

```
>>> # script executed
```

Exercice 19 : (∗)

```
a = 1
while a < 10:
    print(a)
    a = a * 2
```

```
>>> # script executed
```

Exercice 20 : (∗)

```
L = []
x = 0
while x < 3:
    L.append(x)
    x = x + 1
print(L)
```

```
>>> # script executed
```

Exercice 21 : (∗)

```
x = 10
while x > 0:
    if x % 3 == 0:
        print("Divisible par 3 :", x)
    x = x - 1
```

```
>>> # script executed
```

Exercice 22 : (∗∗)

```
a = 2 ** 3 ** 2
print(a)
```

```
>>> # script executed
```

Exercice 23 : (∗)

```
x = 4
y = 3
z = 2
r = x + y * z ** 2
print(r)
```

```
>>> # script executed
```

Exercice 24 : (∗)

```
a = 10
b = 3
c = a / b * b
print(c == a)
```

```
>>> # script executed
```

Exercice 25 : (∗∗)

```
a = True
b = False
c = a + b * 3
print(c)
```

```
>>> # script executed
```

Exercice 26 : (∗∗)

```
a = 2
b = 3
c = 4
res = (a + b) * c // 2 ** 2
print(res)
```

```
>>> # script executed
```

Exercice 27 : (∗)

```
def f(x):
    return x + 2

def g(x):
    return x * 3

a = f(2) + g(1)
print(a)
```

```
>>> # script executed
```

Exercice 28 : $(*)$

```
def h(x, y):
    return x ** y

res = h(2, 1 + 2)
print(res)
```

```
>>> # script executed
```

Exercice 29 : $(*)$

```
def f(a, b):
    return a + b * 2

x = f(1 + 2, 3)
print(x)
```

```
>>> # script executed
```

Exercice 30 : $(**)$

```
def f(x):
    return x * x

def g(x):
    return x + 1

res = f(g(2)) + g(f(2))
print(res)
```

```
>>> # script executed
```

Exercice 31 : $(*)$

```
def min3(a, b, c):
    return min(a + b, c)

x = min3(2, 3 * 2, 10)
print(x)
```

```
>>> # script executed
```

Exercice 32 : $(*)$

```
x = 10

def f():
    x = 5
    return x + 2

print(f())
print(x)
```

```
>>> # script executed
```

Exercice 33 : $(* * *)$

```
a = 2

def g():
    a = a + 1
    return a

print(g())
```

```
>>> # script executed
```

Exercice 34 : (∗∗)

```
b = 4

def h(b):
    return b * 2 + 1

print(h(3))
print(b)
```

```
>>> # script executed
```

Exercice 35 : (∗)

```
y = 1

def f(x):
    y = x * 2
    return y + x

print(f(3))
print(y)
```

```
>>> # script executed
```

Exercice 36 : (∗)

```
m = 1

def f():
    m = 2
    return m ** 2

def g():
    return m + 3

print(f() + g())
```

```
>>> # script executed
```

Exercice 37 : (∗)

```
def f():
    return x + 2

x = 10

print(f())
print(x)
```

```
>>> # script executed
```

Exercice 38 : (∗)

```
x = 10

def f():
    x = 5
    return x * 2 + 1

print(f())
print(x)
```

```
>>> # script executed
```

Exercice 39 : (∗)

```
y = 3

def g():
    return y * y + 2

print(g())
y = 4
print(g())
```

```
>>> # script executed
```

Exercice 40 : (∗∗)

```
x = 5

def h():
    x = x + 1
    return x

print(h())
```

```
>>> # script executed
```

Exercice 41 : (∗∗)

```
a = 2

def f(a):
    return a + 3 * a - 1

print(f(4))
print(a)
```

```
>>> # script executed
```

Exercice 42 : (∗)

```
L = [1, 2, 3]

def f():
    return L[1] + 2

print(f())
```

```
>>> # script executed
```

Exercice 43 : (∗)

```
def f():
    return L[0] + 3

L = [4, 5]

print(f())
```

```
>>> # script executed
```

Exercice 44 : (∗∗)

```
L = [10]

def f():
    L = [1]
    return L[0] + 5

print(f())
print(L)
```

```
>>> # script executed
```

Exercice 45 : (∗∗)

```
L = [2, 4]

def f():
    L[0] = L[0] + 3
    return L

print(f())
print(L)
```

```
>>> # script executed
```

Exercice 46 : (∗∗)

```
L = [1, 2]

def f():
    L.append(3)

f()
print(L)
```

```
>>> # script executed
```

Exercice 47 : (∗∗)

```
def f(x):
    return x**n

L = []
for n in range(3):
    L.append(f(2))
print(L)
```

```
>>> # script executed
```

Exercice 48 : (∗∗)

```
L = [1]

def f():
    L = L + [2]
    return L

print(f())
```

```
>>> # script executed
```

Exercice 49 : (∗)

```
L = []
for k in range(11):
    L.append(k/10)
print(L)
```

```
>>> # script executed
```

Exercice 50 : (∗∗)

```
def f(x):
    return x**2

f = f(3)
print(f)
f = f(4)
print(f)
```

```
>>> # script executed
```

Exercice 51 : (∗)

```
a = True
b = False
print(a and b)
```

```
>>> # script executed
```

Exercice 52 : (∗)

```
x = 5
y = 0
print((x > 0) and (y != 0))
```

```
>>> # script executed
```

Exercice 53 : (∗)

```
a = True
b = False
print(not (a or b))
```

```
>>> # script executed
```

Exercice 54 : (∗)

```
x = 3
y = 7
z = 10
print((x < y) or (z < y) and not (x == 3))
```

```
>>> # script executed
```

Exercice 55 : (∗)

```
val = 0
if not val:
    print("Condition vraie")
else:
    print("Condition fausse")
```

```
>>> # script executed
```

Exercice 56 : (∗)

```
a = True
b = True
c = False
print((a and b) or (not c))
```

```
>>> # script executed
```

Exercice 57 : (∗)

```
s = "abc"
t = "de"
print(s + t)
```

```
>>> # script executed
```

Exercice 58 : (∗)

```
s = "ha"
print(s * 3)
```

```
>>> # script executed
```

Exercice 59 : (∗)

```
s = "Python"
print(s[0], s[-1])
```

```
>>> # script executed
```

Exercice 60 : (∗)

```
s = "Bonjour"
print(len(s))
```

```
>>> # script executed
```

Exercice 61 : (∗∗)

```
def f2(mot):
    v = "aeiouy"
    for c in mot.lower():
        if c in v:
            return True
    return False

print(f2("Anne"))
print(f2("SNCF"))
```

```
>>> # script executed
```

Exercice 62 : (∗)

```
def f3(mot):
    if len(mot) > 5:
        return mot.upper()
    else:
        return mot.lower()

print(f3("python"))
print(f3("code"))
```

```
>>> # script executed
```

Exercice 63 : (∗∗)

```
def f5(s1, s2):
    return len(s1) == len(s2)

print(f5("chat", "chien"))
print(f5("lune", "mars"))
```

```
>>> # script executed
```

Exercice 64 : (∗∗)

```
def f1(a, b):
    return (a > 0) and (b > 0)

print(f1(3, 5))
print(f1(-2, 7))
```

```
>>> # script executed
```

Exercice 65 : (∗∗)

```
def f3(a, b, c):
    return (a == b) and not (b == c)

print(f3(2, 2, 3))
print(f3(4, 4, 4))
```

```
>>> # script executed
```

Exercice 66 : (∗)

```
def f5(n):
    return not (n < 10 or n > 20)

print(f5(15))
print(f5(25))
```

```
>>> # script executed
```