Gabriela ARBELAEZ CHAZY

Activité sur les fonctions

Les fonctions affines

Exercice 1:

1. https://py3.codeskulptor.org/#user309_XvTJnLNcISbYT9f.py
La valeur de la variable *prix* après l'exécution du programme est 165 euros.

2. Ce script fait un calcul d'un nouveau prix en augmentant un taux au prix initial. Dans ce cas le taux est 10% et le prix 150 euros.

3. https://py3.codeskulptor.org/#user309_dwNji4y2eSccDND.py

Exercice 2:

1. Les lignes 2 et 3 servent à calculer la valeur du volume si h est inférieur à 2,5. Si c'est le cas, on calcule le volume 6*h.

2. La ligne 5 sert à calculer la valeur du volume si h est supérieur à 2,5 donc la condition d'avant n'est pas accomplie. C'est comme mettre sinon le calcul du volume est 12,5+h.

Exercice 3:

1. a. https://py3.codeskulptor.org/#user309_Y9JV9u4f34SbWRA.py

b. https://py3.codeskulptor.org/#user309_70s8PkHCW4nasO4.py

2. Ce script permet de calculer une décroissance annuelle de 2% d'une population de 3000 habitants. La boucle *for* est utilisée pour répéter la décroissance annuelle, pour chaque année inclus on multiplie la population par 0,98 ce qui revient à une réduction de 2%.

3. https://py3.codeskulptor.org/#user309_s4dN7FlMOxPRNO2.py

Exercice 4:

1. n ← 0
euros ← 1200
Tant que euros ≤ 1900
n + 1
euros + 80
Fin tant que

2. https://py3.codeskulptor.org/#user309_UGQ6o99ZU3qP6Gq.py

Exercice 5:

1. La fonction *tarifs* calcule le coût total pour un nombre donné d'entrées avec les deux formules de tarification (Formule A et Formule B).

2. La fonction *tarifs* renvoie 2 valeurs, par exemple:
https://py3.codeskulptor.org/#user309_N01NDEN5PRhQ0pz.py

3. À partir de 7 entrées, la formule B est plus avantageuse que la formule A.
https://py3.codeskulptor.org/#user309_er1MA9WKcHKTfM4.py

Exercice 6:

1. Si x ≤ -1, alors:
f ← 2x +1
Sinon si -1 < x < 3 alors:
f←3x+2
Sinon(donc x ≥ 3):
f←-x+4

2. https://py3.codeskulptor.org/#user309_vIb9w0tsv3AKAZN.py

x	-5	-3	-1	0	2	3	11
f(x)	-9	-5	-1	2	8	1	-7

Les fonctions carré et cube

Exercice 1:
https://py3.codeskulptor.org/#user309_8XZ7OAzUh7iZo2E.py
J'ai mis un exemple avec R=6

Exercice 2:
1. $f(x)=-0,5x^3+1$

2.

x	-4	-2	-1,5	-1	0	2,5	4	5,5	7
f(x)	33	5	2,6	1,5	1	-6,8	-31	-82,1	-170,5

https://py3.codeskulptor.org/#user309_6GgIfeX7Ht2nArn.py

3. Pour calculer $f(\sqrt{7})$ il faut saisir sur python:
import math
f(math.sqrt(7))
https://py3.codeskulptor.org/#user309_8yALH1PU2WNJEAJ.py

4. a. Lorsqu'on observe les valeurs de f(x), nous pouvons constater que f(x) diminue à mesure que x augmente en R.

b. Définir seuil_f()
n←0
Tant que f(n) est supérieur ou égal à -1000, faire:

- Incrémenter n de 1

Fin Tant que
Retourner n

Exercice 3:

1. https://py3.codeskulptor.org/#user309_DMvZ1GS92s3nuJS.py
On obtient une énergie cinétique de 720 J.

2. https://py3.codeskulptor.org/#user309_Q6Kf7Svn1RQFih3.py
J'ai pris les mêmes valeurs qu'avant pour prouver que le programme fonctionne.

Travaux pratiques

Partie 1

1. La ligne 4 du programme calcule une nouvelle valeur approchée de la racine carrée de 2 en appliquant la méthode d'approximation de Héron.

2. Le rôle de n est de représenter le nombre de répétitions de la boucle.

3.https://py3.codeskulptor.org/#user309_LfTgQu94IuwzmJ1.py

n	1	4	10	12
valeur de L	1,5	1,414	1,414	1,414

4. On constate que les résultats du programme deviennent de plus en plus proches de la valeur exacte de $\sqrt{2}$ avec l'augmentation du nombre de répétitions.

5. Pour comparer la valeur de heron(10) avec la valeur approchée de √2 donnée par Python, on peut utiliser l'instruction suivante:
https://py3.codeskulptor.org/#user309_NrzjITVoPpRsuXj.py

6. a.

b.

c.

Partie 2:

1. n exécutant la fonction approx_sqrt2(n) avec les valeurs n=3 et n=6, le résultat est une approximation de √2 avec la précision souhaitée. La fonction approx_sqrt2 augmente la valeur de l jusqu'à atteindre le nombre le plus proche de √2 dans le cadre autorisé par n répétitions.

2. La fonction effectue une boucle for avec n+1 itérations. Dans chaque itération, la condition while l**2<2 augmente l d'une fraction de plus en plus petite (10**-n), jusqu'à ce que le carré de l soit proche de 2. Cela garantit que la valeur retournée est de plus en plus proche de la racine carrée de 2, avec une précision déterminée par la valeur de n.

3. L'efficacité de la méthode d'Héron est généralement supérieure à celle de la méthode par balayage. Cela est dû au fait que la méthode d'Héron est une méthode itérative à convergence rapide utilisant des approximations successives, tandis que la méthode par balayage est plus lente, car elle effectue une augmentation linéaire jusqu'à trouver une approximation. Par conséquent, pour obtenir une haute précision, la méthode d'Héron est préférable.