Écrire un programme Python fonctionnel

Alexandre Roulois (Université Paris Cité, LLF, CNRS)

Table of contents

Objectif

En randonnée dans le Vercors, vous avez pour objectif de franchir le col Vert (1776 m) qui, depuis votre point de départ de Villard-de-Lans, présente une dénivelée totale de 697 m annoncée par votre guide à 5,31 % de moyenne. Calculez la distance à parcourir !

Trigonométrie

Problème de trigonométrie classique :

- pente 5,31 % = 5,31 m de dénivelée tous les 100 m
- calcul de la tangente de l'angle : $tan(\alpha) = \frac{5,31}{100}$
- fonction inverse pour obtenir la valeur de l'angle (en radians) : $arctan(\alpha) = 0,053$
- conversion en degrés décimaux : $\frac{arctan(\alpha) \times 180}{\pi} = 3,039°$
- soit : $cos(\alpha) = \frac{\text{adjacent}}{\text{hypoténuse}}$ donc : $\vec{AC'} = \frac{100}{cos(\alpha)} \approx 100,14$
- règle linéaire : $\vec{AC} = \vec{AC'} \times \frac{697}{5,31} \approx 13144,67$

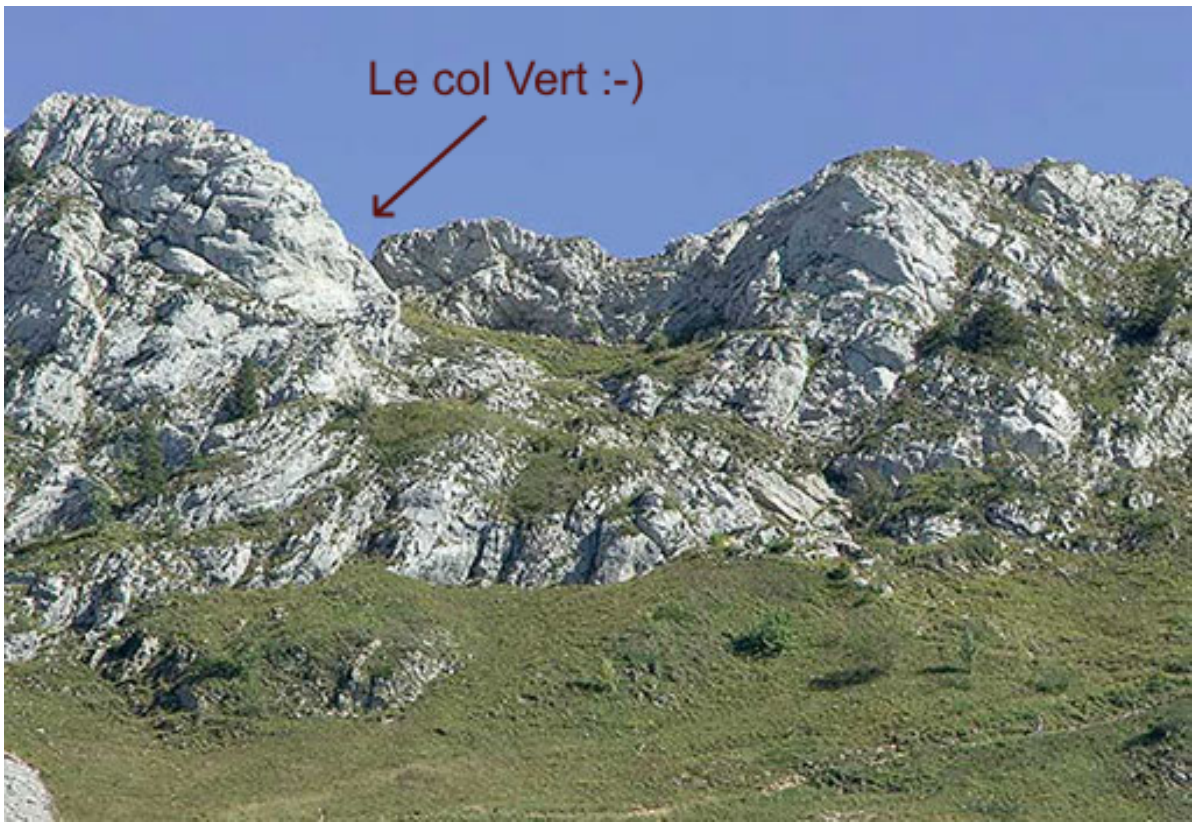

Figure 1: Franchir le col Vert

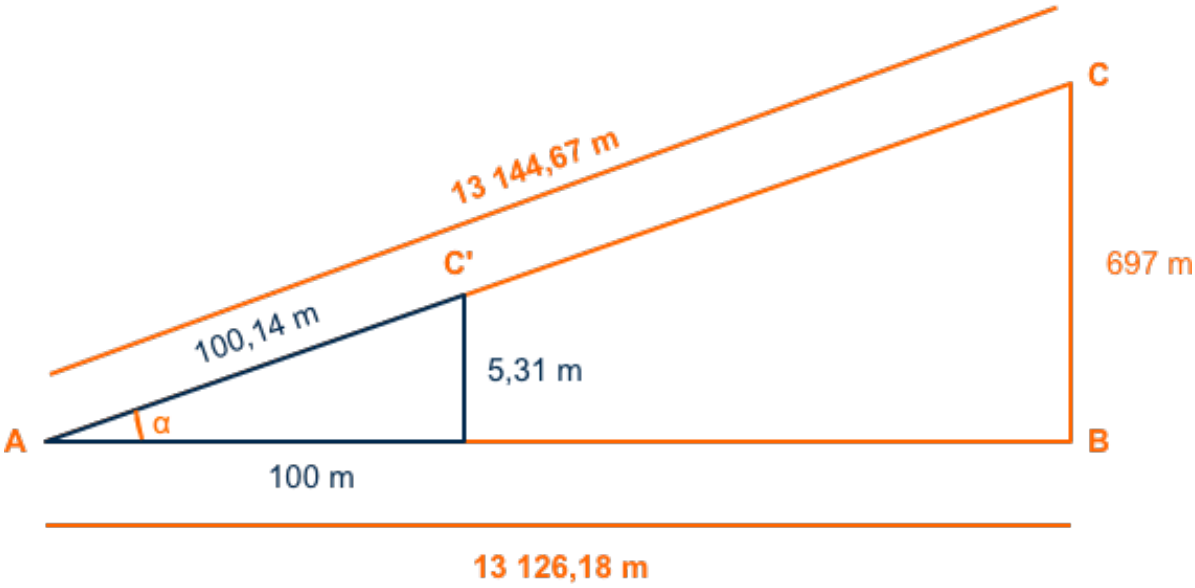

Figure 2: Schéma de la randonnée du point du vue de la trigonométrie

Écriture du programme avec Python

Dans un premier temps, tout programme débute par un préambule :

```
#!/usr/bin/env python
#-*- coding: utf-8 -*-
```

Comme les fonctions trigonométriques ne sont pas disponibles dans le noyau de Python, mais dans une librairie connexe (*math*), il faut l'activer :

```
import math
```

Ensuite, dans la procédure principale, on enregistre les données du problème :

```
if __name__ == "__main__":

  alt = 697               # altitude
  slope = 5.31            # difference in height over 100 m
  alpha = slope / 100     # mesuring tangent
```

On mesure ensuite l'arc tangente du nombre afin d'obtenir une mesure en radians. La fonction trigonométrique `atan( )` est disponible dans le module *math* :

```
arc = math.atan(alpha)      # arctan function
```

On peut désormais calculer l'hypoténuse grâce à la fonction `cosinus()` :

```
hypo = 100 / math.cos(arc)  # cosinus function
```

Enfin, on calcule la distance totale :

```
distance = hypo * (alt / slope)
```

Sans oublier de l'afficher :

```
print(distance)
```

Résolution avec le théorème de Pythagore

Le résultat aurait pu s'obtenir plus facilement par application du théorème de Pythagore (mais c'eût été moins drôle) :

> Dans un triangle rectangle, le carré de la longueur de l'hypoténuse est égal à la somme des carrés des longueurs des deux autres côtés.

Autrement dit, si dans un triangle ABC rectangle en B, le vecteur BC mesure 697 m et que tous les 100 m le long du vecteur AB on s'élève de 5,31 m, alors $AB = \frac{BC}{5.31} \times 100$ soit :

```
bc = 697
slope = 5.31
ab = (bc / slope) * 100

print(ab)
```

D'après le théorème, on sait que $AC^2 = BC^2 + AB^2$:

```
ac_square = (bc ** 2) + (ab ** 2)

print(ac_square)
```

Il ne reste plus qu'à déterminer la racine carrée du vecteur AC pour connaître la longueur de l'hypoténuse :

```
ac = math.sqrt(ac_square)

print(ac)
```

Un programme fonctionnel

Écrivons un programme plus pratique et mieux structuré. De quoi aura-t-on besoin ?

1. importer le module `math` ;
2. écrire une fonction pour calculer l'hypoténuse ;
3. écrire une fonction `main()` reprise dans la procédure principale ;
4. écrire la procédure principale.

Étape 1 : importer le module `math`

```
#!/usr/bin/env python
#-*- coding: utf-8 -*-

#
# Load modules
```

```
#
import math
```

Étape 2 : écrire la fonction qui calcule l'hypoténuse

```
def hypotenuse(x, y):
  """Calculates the hypotenuse thanks to
  the Pythagorean theorem in a right triangle.

  Keyword arguments:
  x -- first cathetus
  y -- second cathetus
  """
  square = (x ** 2) + (y ** 2)
  hypotenuse = math.sqrt(square)

  # rounded to two digits from the decimal point
  return round(hypotenuse, 2)
```

Étape 3 : écrire la fonction `main()`

```
def main():

  # writes the argument to standard output,
  # then reads a line from input and returns it as a string.
  x = input("Quelle est la mesure de la première cathète (en cm) ?")
  y = input("Quelle est la mesure de la seconde cathète (en cm) ?")

  x, y = int(x), int(y)

  print(f"La mesure de l'hypoténuse vaut {hypotenuse(x, y)} cm.")
```

Étape 4 : écrire la procédure principale

```
#
# Main procedure
#
if __name__ == "__main__":

    main()
```

Il reste un problème à régler : s'assurer que les entrées de l'utilisateur ou de l'utilisatrice soient bien des chiffres !

```
def main():

  while True:
    try:
      # Writes the argument to standard output,
      # then reads a line from input and returns it as a string.
      x = int(input("Quelle est la mesure de la première cathète (en cm) ?"))
      y = int(input("Quelle est la mesure de la seconde cathète (en cm) ?"))
    except ValueError:
      print("Veuillez entrer un chiffre")
    else:
      print(f"La mesure de l'hypoténuse vaut {hypotenuse(x, y)} cm.")
      break
```