

# Le projectile [si02] - Exercice

Karine Zampieri, Stéphane Rivière

Unisciel  algoprogram  Version 22 mai 2018

## Table des matières

<b>1</b>	<b>Le projectile / pgprojectile</b>	<b>2</b>
1.1	Distance de tir (distanceTir, saisirBoulet) . . . . .	2
1.2	Mouvement d'un projectile . . . . .	3
1.3	Test de tir . . . . .	5
<b>2</b>	<b>Références générales</b>	<b>6</b>

## Python - Le projectile (Solution)



Mots-Clés Simulation ■

Requis Axiomatique impérative (sauf Fichiers) ■

Optionnel Graphique ■

Difficulté ●○○



### Objectif

Cet exercice détermine puis trace la trajectoire d'un boulet de canon par simulation physique.

# 1 Le projectile / pgprojectile

## 1.1 Distance de tir (distanceTir, saisirBoulet)

### Distance de tir

Un boulet de canon qui sort avec une vitesse  $v_0$  et une inclinaison d'angle  $\alpha$  par rapport à l'horizontale atterrit plus loin (sur terrain plat) à une distance :

$$dist = v_0^2 \sin(2\alpha)/g$$

où  $g = 9.80665 \text{ m/s}^2$  est la constante gravitationnelle.



Définissez la constante `GTERRE=9.81` (accélération de la pesanteur).



Définissez le type `Boulet`, structure contenant la vitesse `v0` (réel) et l'angle d'inclinaison `alpha` (réel) initiaux d'un boulet.



Validez vos définitions avec la solution.

### Solution Python

```
GTERRE = 9.81
""" Accélération de la pesanteur """
class Boulet:
    """ Représente un Boulet """
    def __init__(self):
        self.v0 = 0.0
        """ vitesse initiale """
        self.alpha = 0.0
        """ angle d'inclinaison en radians """
```



Écrivez une fonction `distantTir(b)` qui calcule et renvoie la distance du tir d'un `Boulet b` (voir formule ci-dessus).

### Outil Python

La fonction sinus `sin(x)` est définie dans la bibliothèque `math`.

### Solution Paramètres

Entrants : Un `Boulet b`

Résultat de la fonction : Un réel



Validez votre fonction avec la solution.

**Solution Python**

```
def distanceTir(b):
    """ Distance d'un tir

    :param b: un Boulet
    :return: la distance du tir de b
    """
    return (b.v0 * b.v0 * math.sin(2.0 * b.alpha)) / GTERRE
```



Écrivez une procédure `saisirBoulet(b)` qui saisit les caractéristiques d'un Boulet dans `b`. Affichez l'invite :

Angle et vitesse du boulet?



Écrivez une procédure `afficherBoulet(b)` qui affiche les caractéristiques (angle et vitesse) d'un Boulet `b`.



Validez vos procédures avec la solution.

**Solution Python**

```
def saisirBoulet(b):
    """ Saisie les caractéristiques d'un Boulet

    :param b: un Boulet
    """
    print("Angle et vitesse du boulet? ", sep="", end="")
    b.alpha = float(input())
    b.v0 = float(input())
```

```
def afficherBoulet(b):
    """ Affiche les données d'un Boulet

    :param b: un Boulet
    """
    print("=> angle et vitesse du boulet ", b.alpha, " ", b.v0, sep="")
```

**1.2 Mouvement d'un projectile**

Un projectile de position  $p(t) = \begin{pmatrix} x \\ y \end{pmatrix}$  de vitesse  $v(t) = \begin{pmatrix} v_x(t) \\ v_y(t) \end{pmatrix}$  soumis à une accélération  $a(t) = \begin{pmatrix} a_x(t) \\ a_y(t) \end{pmatrix}$  pendant un temps  $\delta t$  a pour nouvelle vitesse :

$$v(t + \delta t) = v(t) + a \cdot \delta t$$

et nouvelle position :

$$p(t + \delta t) = p(t) + v(t + \delta t) \cdot \delta t$$



Écrivez une procédure `bougerProjectile(x,y,vx,vy,ax,ay,dt)` qui calcule la **nouvelle** position dans  $(x,y)$  et la **nouvelle** vitesse dans  $(vx,vy)$  d'un projectile d'accélération  $(ax,ay)$  à l'instant  $dt$  suivant. Tous les paramètres sont des réels.



Écrivez une fonction `distanceSimulee(b,dt)` qui calcule et renvoie la distance du tir d'un projectile lancé avec un Boulet  $b$  soumis à la force de pesanteur (constante `GTERRE`) de valeur 9.81, le delta temps étant  $dt$  (réel).



### Aide méthodologique

Placez le projectile en  $(0,0)$  de vitesse initiale  $(v_0 \cos \alpha, v_0 \sin \alpha)$ , puis bougez le projectile avec l'accélération  $(0, -g)$  jusqu'à ce que  $y \leq 0$ , enfin retournez  $x$ .

### Outil Python

Les fonctions sinus `sin(x)` et cosinus `cos(x)` sont définies dans la bibliothèque `math`.



Validez votre procédure et votre fonction avec la solution.

### Solution Python @[pgprojectile.py]

```

def bougerProjectile(x, y, vx, vy, ax, ay, dt):
    """ Bouge un projectile

    :param x: position x
    :param y: position y
    :param vx: vitesse x
    :param vy: vitesse y
    :param ax: accélération x
    :param ay: accélération y
    :param dt: delta temps
    :return: le tuple (x,y,vx,vy)
    """
    vx += ax * dt
    vy += ay * dt
    x += vx * dt
    y += vy * dt
    return (x, y, vx, vy)

def distanceSimulee(b, dt):
    """ Distance simulée d'un tir

    :param b: un Boulet
    :param dt: delta temps
    :return: la distance d'un tir de vitesse v0 d'angle alpha de delta dt
    """
    x = 0.0
    y = 0.0
  
```

```

vx = b.v0 * math.cos(b.alpha)
vy = b.v0 * math.sin(b.alpha)

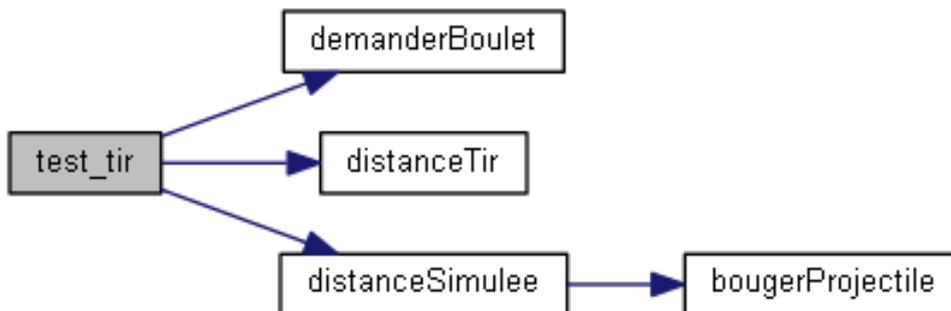
btir = True
while btir:
    x, y, vx, vy = bougerProjectile(x, y, vx, vy, 0.0, -GTERRE, dt)
    btir = (y > 0.0)
return x

```

### 1.3 Test de tir



Écrivez une procédure `test_tir` qui saisit les caractéristiques de tir d'un boulet (angle et vitesse) puis calcule et affiche la distance exacte et les distances simulées à 0.1 et 0.01 du tir.



Testez. Exemple d'exécution :

```

angle et vitesse du boulet? 0.6 50
==> Distance exacte = 237.5227028542
==> Distance simulee 0.1 = 235.2206502465
==> Distance simulee 0.01 = 237.2839892837

```



Validez votre procédure avec la solution.

**Solution Python** @[pgprojectile.py]

```

def PGProjectile():
    b = Boulet()
    saisirBoulet(b)
    print("==> Distance exacte = ", distanceTir(b), sep="")
    print("==> Distance simulee 0.1 = ", distanceSimulee(b, 0.1), sep="")
    print("==> Distance simulee 0.01 = ", distanceSimulee(b, 0.01), sep="")

```



On désire saisir l'angle en degrés.  
Que faut-il modifier dans le programme principal ?

### **Solution simple**

On écrit une fonction `degVersRad(alpha)` qui donne l'équivalent en radians d'un angle `alpha` en degrés puis, après la saisie (en degrés), on calcule son équivalent en radians avant de lancer les calculs.

## **2 Références générales**

Comprend [Boudreault-CC1 :c2 :td6] ■