**EXERCICE 1 :**

def iscoel(a, b, c):

if (a == 1 and b == 1 and c== 20):

print("faux")

return(0)

elif (a == 1 and b == 20 and c== 1):

print("faux")

return(0)

elif (a == 20 and b == 1 and c ==1):

print("faux")

return(0)

if ((a==b) or (a==c)):

print("vrai")

return(1)

if ((b==a) or (b==c)):

print("vrai")

return(1)

if((c==a) or (c==b)):

print("vrai")

return(1)

else:

print("faux")

return(0)

iscoel(2, 2, 5)

iscoel(1, 20, 1)

iscoel(2, 9, 4)

iscoel(5, 8, 1)

iscoel(9, 4, 1)

**EXERCICE 2 :**

import math

def aire\_ordonne(a,b,c):

# debut de verification

u3 = max(a,b,c)

if (u3 == a):

u1 = min(b,c)

if (u1 == b):

u2 = c

else:

u2 = b

elif(u3 == b):

u1 = min(a,c)

if (u1 == a):

u2 = c

else:

u2 = c

elif (u3 == c):

u1 = min(a,b)

if (u1 == a):

u2 = b

else:

u2 = a

# fin de verification

d = (pow(u1,2)-pow(u2,2)+pow(u3,2))/2

f = pow(d,2)

e = (pow(u1,2)\*pow(u3,2))-f

A = (math.sqrt (e)) / 2

print(A)

aire\_ordonne(4,2,3)

aire\_ordonne(4,3,3)

aire\_ordonne(4,4,4)

aire\_ordonne(3,4,5)

aire\_ordonne(13,14,15)

aire\_ordonne(1,1,1)

**EXERCICE 3 :**

def definit\_triangle(a,b,c):

if (a>0 and b>0 and c>0):

y = (a+b+c)/2

if (a<y and b<y and c<y):

return(True)

else:

return(False)

else:

return(False)

print(definit\_triangle(1,1,20))

print(definit\_triangle(4,2,3))

print(definit\_triangle(4,4,4))

print(definit\_triangle(3,9,4))

print(definit\_triangle(6,4,2))

**EXERCICE 4 :**

import math

# def of perimetre function

def perim(a, b, c):

return (a + b +c)

# def of valid triangle

def definit\_triangle(a, b, c):

# verif si positif

if (a < 0 or b < 0 or c < 0):

return (False)

x = (a + b + c) / 2

if (a >= x or b >= x or c >= x):

return (False)

return (True)

# def air of triangle comme exo3

def air\_ordonne(a, b, c):

# select min and max value for u1 u2 u3

# u1 : min value

# u3 : max value

# u2 : mid value

# start verif

u3 = max(a,b,c)

if (u3 == a):

u1 = min(b,c)

if (u1 == b):

u2 = c

else:

u2 = b

elif (u3 == b):

u1 = min(a,c)

if (u1 == a):

u2 = c

else:

u2 = c

elif (u3 == c):

u1 = min(a,b)

if (u1 == a):

u2 = b

else:

u2 = a

# end of verif

x = math.pow(u1,2) \* math.pow(u3,2)

y = (math.pow(u1,2) - math.pow(u2,2) + math.pow(u3,2)) / 2

z = x - math.pow(y,2)

if (z < 0):

return (-1)

final = (math.sqrt(z)) / 2

return (final)

def check\_duplicat(i, j, k):

if (i <= j and j <= k):

return (True)

return (False)

# function

def nb\_triangles\_speciaux(n,p):

if (n < 0 or p < 0):

return ("error :> nombre n or p negative")

if (n > p):

return ("error :> please specify n > p when you call the function (n,p)")

i = n

ret = 0

while (i <= p):

j = n

while (j <= p):

k = n

while (k <= p):

if (definit\_triangle(i, j, k)):

if (air\_ordonne(i, j, k) == perim(i, j, k)):

if (check\_duplicat(i, j, k)):

print ("(", i, ",", j, ",", k, ")")

ret += 1

k += 1

j += 1

i += 1

return (ret)

def simulation\_ex04():

print (" -------------------------- Ex04 ---------------------------------")

print (" ")

print ("test : nb\_triangles\_speciaux(1,20)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(1,20))

print (" ")

print (" ")

print ("test : nb\_triangles\_speciaux(1,50)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(1,50))

print (" ")

print (" ")

print ("test : nb\_triangles\_speciaux(8,20)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(8,20))

print (" ")

print (" ")

print ("test : nb\_triangles\_speciaux(8,50)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(8,50))

print (" ")

print (" ")

print ("test : nb\_triangles\_speciaux(-5,-6)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(-9,-6))

print (" ")

print (" ")

print ("test : nb\_triangles\_speciaux(-5,60)")

print ("ensemble des combinaisons")

print ("result : ", nb\_triangles\_speciaux(-9,60))

print (" ")

simulation\_ex04()