1. Define a function isEven(number) that takes in a number as an argument and returns True if it is an even number.

**Examples**

>>> isEven(0)

True

>>> isEven(1)

False

>>> isEven(-2)

True

def isEven(x):

if x % 2 == 0:

return True

else:

return False

1. Write a function isIsosceles(x, y, z) that accepts the 3 sides of a triangle as inputs. The function should return True if it is an isosceles triangle. An isosceles triangle has 2 equal sides. An equilateral triangle is a special case of isosceles triangle.

Examples

>>> isIsosceles(2, 4, 3)

False

>>> isIsosceles(3, 3, 3)

True

>>> isIsosceles(2, 3, 2)

True

>>> isIsosceles(-2, 3, -2)

False

>>> isIsosceles(0, 0, 2)

False

A:

def isIsosceles(x, y, z):

if x <= 0 or y <= 0 or z <= 0:

return False

if x==y or y==z or z==x:

return True

else:

return False

1. Write a function isScalene(x, y, z) that accepts the 3 sides of a triangle as inputs. The function should return True if it is a scalene triangle. A scalene triangle has no equal sides.

Examples

>>> isScalene(2, 4, 3)

True

>>> isScalene(3, 3, 3)

False

>>> isScalene(2, 2, 3)

False

A:

def isScalene(x, y, z):

if x != y and y != z and z != x:

return True

else:

return False

1. Define a function to determine the standard achieved by a participant taking a physical fitness test.

The standard is determined based on the individual and total scores for 3 stations.

|  |  |  |  |
| --- | --- | --- | --- |
| Gold | Silver | Pass | Fail |
| Min. of 4 points for each station, and min. total of 13 | Min. of 3 points for each station, and min. total of 10 | Min. of 2 points for each station, and min. total of 7 | Less than 2 points for any station or total<7 |

Examples

>>> Fitness(4,5,4)

'Gold'

>>> Fitness(4,4,4)

'Silver'

>>> Fitness(1,5,5)

'Fail'

>>> Fitness(2,2,5)

'Pass'

A:

def Fitness(a, b, c):

if a >= 4 and b >= 4 and c >= 4 and (a+b+c) >= 13:

return 'Gold'

elif a >= 3 and b >= 3 and c >= 3 and (a+b+c) >= 10:

return 'Silver'

elif a >= 2 and b >= 2 and c >= 2 and (a+b+c) >= 7:

return 'Pass'

else:

return 'Fail'

1. Define a function isPrime(number) that takes in a number as argument and return True if the number is a prime number.

Hint: A number, x is a prime number if it is only divisible by 1 and x itself.

By definition, 1 is not a prime number.

Examples

>>> isPrime(97)

True

>>> isPrime(1)

False

>>> isPrime(-2)

False

A:

# Hint: Step through the range between (2, number-1),

# and determine if the number is divisible using the modulus operator.

def isPrime(x):

if x > 1:

for i in range(2,x):

if x % i == 0:

return False

return True

else:

return False

1. Write a function using 'if/elif/else' conditionals to compute the BMI of a person, and return the risk associated with cardiovascular diseases.

BMI = weight(kg)/( height(m)\*height(m) )

|  |  |
| --- | --- |
| BMI | Risk |
| 27.5 and above | High Risk |
| 23 - 27.4 | Moderate Risk |
| 18.5 - 22.9 | Low Risk |
| Below 18.5 | Risk of nutritional deficiency diseases |

Examples

>>> HealthScreen(85, 1.75)

'Your BMI is 27.8 (High Risk).'

>>> HealthScreen(68, 1.65)

'Your BMI is 25.0 (Moderate Risk).'

>>> HealthScreen(60, 1.63)

'Your BMI is 22.6 (Low Risk).'

>>> HealthScreen(40,1.58)

'Your BMI is 16.0 (Risk of nutritional deficiency diseases).'

A:

def HealthScreen(weight, height):

BMI = weight/(height\*height)

BMI = round(BMI,1)

if BMI >= 27.5:

return 'Your BMI is '+str(BMI)+ ' (High Risk).'

elif BMI >= 23 and BMI <= 27.4:

return 'Your BMI is '+str(BMI)+ ' (Moderate Risk).'

elif BMI >= 18.5 and BMI <= 22.9:

return 'Your BMI is '+str(BMI)+ ' (Low Risk).'

else:

return 'Your BMI is '+str(BMI)+ ' (Risk of nutritional deficiency diseases).'

1. Define a function that takes in 3 values and determine if they can form the sides of an triangle.

Hint: The sum of the lengths of any two sides of a triangle is

greater than the length of the third side.

Examples

>>> isTriangle(3,4,5)

True

>>> isTriangle(1,3,1)

False

A:

def isTriangle(x,y,z):

if (x+y<=z) or (x+z<=y) or (y+z<=x):

return False

else:

return True

1. For a quadratic equation in the form of ax2 + bx + c, the discriminant, D is b2-4ac.  
   Write a function that return the following output depending on the discriminant.  
   D > 0: 2 real roots.  
   D = 0: 1 real root.  
   D < 0: 2 complex roots.

Examples

>>> quadratic(1, 2, 3)

'This equation has 2 complex roots.'

>>> quadratic(1, 3, 2)

'This equation has 2 real roots.'

>>> quadratic(1, 4, 4)

'This equation has 1 real root.'

A:

def quadratic(a, b, c):

d = b\*\*2 - 4\*a\*c

if d > 0:

return 'This equation has 2 real roots.'

elif d < 0:

return 'This equation has 2 complex roots.'

else:

return 'This equation has 1 real root.'

1. Write a function that converts the time to 24hr format.

Examples

>>> time24hr('12:34am')

'0034hr'

>>> time24hr('12:15pm')

'1215hr'

A:

def time24hr(tstr):

tstr.replace("\'","")

tstr.replace(":","")

if tstr[-2:] == "am" and tstr[:2] == "12":

return "00" + tstr[3:-2] + "hr"

elif tstr[-2:] == "am":

return tstr[:-2] + "hr"

elif tstr[-2:] == "pm" and tstr[:2] == "12":

return tstr[:2] + tstr[3:-2] + "hr"

else:

temp = tstr[:1]

val = int(temp) + 12

return str(val) + tstr[2:-2] + "hr"

1. Write a function that determines if a given year is a leap year.  
   A leap year is divisible by 4, but not by 100, unless it is also divisible by 400.

Examples

>>> LeapYear(2012)

True

>>> LeapYear(2010)

False

A:

def LeapYear(yr):

return (yr % 4 == 0 and yr % 100 != 0) or yr % 400 == 0