In [1]:
```
#1. Create a function that takes a boolean variable flag and returns it as a s
#Examples
#bool_to_string(True) → "True"
#bool_to_string(False) → "False"

def bool_to_string(flag):
    if flag==True:
        return "True"
    elif flag==False:
        return "False"
    else:
        return None
```

In [2]:
```
bool_to_string(True)
```

Out[2]: 'True'

In [3]:
```
bool_to_string(False)
```

Out[3]: 'False'

In [6]:
```
#2 Create a function that returns True when num1 is equal to num2; otherwise r
#Examples
#is_same_num(4, 8) → False

#is_same_num(2, 2) →  True

#is_same_num(2, "2") → False


def f1(num1,num2):
    if num1==num2:
        return True
    else:
        return False
```

In [7]:
```
f1(1,2)
```

Out[7]: False

In [8]:
```
f1(1,1)
```

Out[8]: True

In [12]:
```
#3 Create a function that takes a number as its only argument and returns True
#otherwise return False.
#Examples
#less_than_or_equal_to_zero(5) → False
#less_than_or_equal_to_zero(0) → True
#less_than_or_equal_to_zero(-2) → True

def f1(num):
    return num<=0
```

In [13]:
```
f1(0)
```

Out[13]: True

In [14]:
```
#4 Emmy has written a function that returns a greeting to users. However, she'
#and would like to greet him slightly differently. She added a special case in
#but she made a mistake.Can you help her?
#Can you help her?
#greeting("Matt") → "Hello, Matt!"
#greeting("Helen") → "Hello, Helen!"
#greeting("Mubashir") → "Hello, my Love!"

def greeting(name):
    if name=="Mubashir" or name=="mubashir":
        print("Hello, my Love!")
    else:
        print("Hello, {}!".format(name))
```

In [15]:
```
greeting("Matt")
```

```
Hello, Matt!
```

In [16]:
```
#5 Create a function that takes two arguments. Both arguments are integers, a
# Return True if one of them is 10 or if their sum is 10.
#Examples
#makes10(9, 10) → True
#makes10(9, 9) → False
#makes10(1, 9) → True

def makes10(a,b):
    if a==10 or b==10 or a+b==10:
        return True
    else:
        return False
```

In [17]:
```
makes10(9,10)
```

Out[17]: True

In [18]:
```
#6 Create a function that takes three arguments prob, prize, pay and returns T
#To illustrate:profitable_gamble(0.2, 50, 9)
#... should yield True, since the net profit is 1 (0.2 * 50 - 9), and 1 > 0.

def profitable_gamble(prob,prize,pay):
    if prob*prize>pay:
        return True
    else:
        return False
```

In [19]:
```
profitable_gamble(0.2,50,9)
```

Out[19]: True

In [22]:
```
#7 Create a function that takes an integer and returns True if it's divisible
#Examples
#divisible(1) → False
#divisible(1000) → True
#divisible(100) → True

def divisible(x):
    if x%100==0:
        return True
    else:
        return False
```

In [23]:
```
divisible(1)
```

Out[23]: False

In [24]:
```
#8 Create a function that takes two strings as arguments and return either Tru
#of characters in the first string is equal to the total number of characters
#Examples
#comp("AB", "CD") → True
#comp("ABC", "DE") → False
#comp("hello", "edabit") → False

def hello(x,y):
    if len(x)==len(y):
        return True
    else:
        return False
```

In [26]:
```
hello("SAI","RAM")
```

Out[26]: True

In [28]:
```
#9 Given a string, return True if its length is even or False if the length is
#odd_or_even("apples") → True
# The word "apples" has 6 characters.
# 6 is an even number, so the program outputs True.
#odd_or_even("pears") → False
# "pears" has 5 letters, and 5 is odd. # Therefore the program outputs False.
#odd_or_even("cherry") → True

def odd_or_even(s):
    if len(s)%2==0:
        return True
    else:
        return False
```

In [29]:
```
odd_or_even("apples")
```

Out[29]:
```
True
```

In [30]:
```
#10 Create a function that takes a string; we'll say that the front is the fir
#If the string length is less than three characters, the front is whatever is
#Return a new string, which is three copies of the front.
#Examples
#front3("Python") → "PytPytPyt"
#front3("Cucumber") → "CucCucCuc"
#front3("bioshock") → "biobiobio"

def front3(s):
    if len(s)<=3:
        front=s
    else:
        front=s[:3]
    return front*3
```

In [31]:
```
front3("Python")
```

Out[31]:
```
'PytPytPyt'
```

In [32]:
```
#11 Darts is a target game played by throwing feathered darts at a circular bo
#Our darts game is the simplest of all games. The score of a single turn is ca
#the middle. You need to create a function that takes the dart location as two
#returns a score based on the distance from the middle, aka Bullseye (x=0, y=0
#● Bullseye and inner circle scores = 10 points
#● Middle ring scores = 5 points
#● Outer ring scores = 1 point
#● Outside the target = 0 points

def darts(x,y):
    r=((x**2)+(y**2))**2
    if r<=1:
        p=10
    elif r>1 and r<=5:
        p=5
    elif r>5 and r<=10:
        p=1
    else:
        p=0
    return p
```

In [33]:
```
darts(1,1)
```

Out[33]: 5

In [34]:
```
#12  Write a function that returns True if a year is a leap, otherwise return
#lasts 366 days, instead of 365 in a typical year. That extra day is added to
#creating February 29.A leap year occurs every four years, and will take place
#The exception to this is a year at the beginning of a century (for example, 1
#divisible by 4 to be a leap year.

def leap_year(y):
    if y%4==0:
        return True
    else:
        return False
```

In [35]:
```
leap_year(1999)
```

Out[35]: False

In [38]:
```
#13 Create a function that takes an array of hurdle heights and a jumper's jum
#or not the hurdler can clear all the hurdles.A hurdler can clear a hurdle if
#the hurdle height.
#Examples
#hurdle_jump([1, 2, 3, 4, 5], 5) ➞ True
#hurdle_jump([5, 5, 3, 4, 5], 3) ➞ False
#hurdle_jump([5, 4, 5, 6], 10) ➞ True
#hurdle_jump([1, 2, 1], 1) ➞ False

def hurdle_jump(hurdle_heights,jump_height):
    for x in hurdle_heights:
        if x>jump_height:
            return "False"

    return "True"
```

In [39]:
```
hurdle_jump([1,2,3,4,5],5)
```

Out[39]: 'True'

In [40]:
```
#14 Create a function that takes a number (from 1 to 12) and returns its corre
#For example, if you're given 3 as input, your function should return "March",

def month_name(num):
    year={1:"January",2:"February",3:"March",4:"April",5:"May",6:"June",7:"Jul
          10:"October",11:"November",12:"December"}
    for x in year.keys():
        if x==num:
            month=year[x]
        else:
            continue
    return month
```

In [41]:
```
month_name(5)
```

Out[41]: 'May'

In [46]:
```
#16 A bartender is writing a simple program to determine whether he should ser
#He only serves drinks to people 18 and older and when he's not on break. Give
#is in session, create a function which returns whether he should serve drinks

def drinks(age):
    if age>=18:
        return "should_serve_drinks"
    else:
        return "should_serve_drinks"
```

In [47]:
```
drinks(18)
```

Out[47]: 'should_serve_drinks'

In [49]:
```
#15 Create a function that takes in a two-dimensional list and returns the num

def count_identical(lst):
    l=0
    for x in lst:
        if len(x)==1:
            l+=1
        else:
            a=x[0];b=0
            for y in range(1,len(x)):
                if a==x[y]:
                    b+=1
            if b>=1:
                l+=1
    return l
```

In [50]:
```
count_identical([[1, 2], [2, 3],[3, 4],[4, 4]])
```

Out[50]: 1

In [53]:
```
#18 Write a function that accepts base (decimal), height (decimal) and shape (
#as input and calculates the area of that shape.

def area_shape(base=1,height=1,shape="parallelogram"):
    s=shape.lower()
    if s=="triangle":
        area=0.5*base*height
    elif s=="parallelogram":
        area=base*height
    return area
```

In [54]:
```
area_shape(2, 3, "triangle")
```

Out[54]: 3.0

In [55]:
```
#19 Create a function that takes a string (a random name). If the last charact
#otherwise return False.

def s(name):
    if name[-1]=="n":
        return "True"
    else:
        return "False"
```

In [56]:
```
s("name")
```

Out[56]: 'False'

In [57]:
```
#20 Create a function that takes in a word and determines whether or not it is

def is_plural(s):
    if s.endswith("s"):
        return True
    else:
        return False
```

In [58]:
```
is_plural("dancers")
```

Out[58]:
```
True
```

In [65]:
```
#21 Create a function that takes a number as an argument and returns "even" fo

def f1(num):
    if num%2==0:
        return "even"
    else:
        return "odd"
```

In [66]:
```
f1(10)
```

Out[66]:
```
'even'
```

In [ ]:
```

```