Question

.Build a Python program utilizing data structures, conditional statements, and functions to solve a data-related task.

LIST:list are ordered collections of items, which can be of any data type.They are mutable,means you can add,remove or modify elements after the list is created.Lists are created using square brackets[]and can contain elements seperated by commas.

LIST:

```
# creating a list with different data types
first_lst=["string",True,69]
```

```
print(first_lst)
```

```
['string', True, 69]
```

```
# Creating another list
lst2=['sam','Yusuf','Ukasha','Umar','Sultan','jerry']
```

```
print(lst2)
```

```
['sam', 'Yusuf', 'Ukasha', 'Umar', 'Sultan', 'jerry']
```

```
# Add an element in a list
lst2[0]='pablo'
```

```
lst2
```

```
['pablo', 'Yusuf', 'Ukasha', 'Umar', 'Sultan', 'jerry']
```

```
# Remove an element in a list
lst2.remove('pablo')
```

```
lst2
```

```
['Yusuf', 'Ukasha', 'Umar', 'Sultan', 'jerry']
```

TUPLES: Tuples are similar to a lists but are immutable,means their elements cannot be changed after creation. They are created using parentheses()and can contain elements separated by commas.Tuples are often used for storing fixed collections of related data.

TUPLES:

```
# Creating a tuple
squares_t=(1,9,4,9,16,25,36,49,64,81,100)
```

```
squares_t (1,9,4,9,16,25,36,49,64,81,100)
type('squares')
```

```
    str
```

```
type(squares_t)
```

```
    tuple
```

```
print(squares_t)
```

```
    (1, 9, 4, 9, 16, 25, 36, 49, 64, 81, 100)
```

```
b_my_tuple=('my_tuple')
```

```
b_my_tuple
```

```
    'my_tuple'
```

```
print('my_tuple')
```

```
    my_tuple
```

SET: Sets are unordered collections of unique elements.They don't allow duplicate elements,and the order of elements is not guaranted.Sets are created using curly brackets{}or the set() function.

SETS:

```
# creating a script that store some values and assign it to a variable my_set
my_set={2,4,6,8}
```

Displaying the values of the sets

```
# Displaying the values of sets
print(my_set)
```

```
    {8, 2, 4, 6}
```

```
new_sets={1,2,4,5,6,1,2,3,5,4,6,9,8,0,2,1,3,6,9,1,4,5,8,4,7,2,3,6,7,5,8,9,2,6,2,5,}
```

```
new_sets
```

```
    {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

```
an_set={'Nickname','Intigers','Destination'}
```

```
an_set
```

```
    {'Destination', 'Intigers', 'Nickname'}
```

DICTIONARIES: Dictionaries are unordered collections of key-value pairs.Each element in a dictionary consists of a key and it's corresponding value,separated by a colon:. Dictionaries are created using curly brackets{}and key-value Par separated by commmas.

EXAMPLES:

```
# Creating a dictionaries script that store some values and assign it to a variable first_dict.
first_dict={'name:''jabir:','age:',20,'city:''Nework'}
```

Displaying the Dict

```
# print the value of the dictionaries
print(first_dict)
```

```
    {'name:jabir:', 20, 'age:', 'city:Nework'}
```

```
first_dict
```

```
    {20, 'age:', 'city:Nework', 'name:jabir:'}
```

Introduce conditional statements and functions through practical examples and coding exercises. Conditional statements:- Conditional statementsin Python allow you to execute different blocks of code based on whether a certain condition is true or false.The most commonly used conditional statements in Python are if,elif(short for"else if"),and else.Here's simple example:-

Conditional statements:- In this example,if the value of x is greater than 0,the message "x is positive" will be printed.If the value of x is equal to 0, the message "x is zero" will be printed.Otherwise, if none of the above conditions are met,the message "x is negative" will be printed.

```
# Conditional statements
x=4
```

```
if x < 0:
  print("x is positive")
```

```
print("x is lower")
```

```
    x is lower
```

```
age=20
if age >18:
  print("eligible to vote")
```

```
    eligible to vote
```

```
  print("not eligible to vote")
```

```
    not eligible to vote
```

Functions: Functions in Python allow you to encapsulate reusable blocks of code in to named blocks,making your code modular and easier to understand.Here is an example of defining and using a function. Function: In this example, greet is a function that takes one parameter and prints a greeting message using that name. When you call the function("Ukasha"),it will print "hello, Ukasha!".

```
# Define a function called greet
def greet(Ukasha):
    print("Hello,Ukasha")


# Calling the function
greet("Ukasha")
```

```
    Hello,Ukasha
```

Coding Exercises: Now let's combine conditional statements and functions in a coding exercise.Write a python function called check_even_odd that takes an intigers as input and prints whether the number is even or odd.Here is a template that get me stated.

```
# Testing the function
check_even_odd(20)
check_even_odd(9)
```

```
    20 is even
    num,is odd
    9 is even
    num,is odd
```