Python Programming Lab Manual

1. **If you run a 10 kilometer race in 43 minutes 30 seconds, what is your average time per mile? What is your average speed in miles per hour? (Hint: there are 1.61 kilometers in a mile)**

```
distKm = 10.0
distMi = (distKm * (1.0 / 1.61))
seconds = ((43.0 * 60.0) + 30.0)
hours = (43.5 / 60.0)

def timePerMile(distMi, seconds):
   print(seconds / distMi, 'seconds per mile')

def avgSpeed(distMi, hours):
   print( distMi / hours, 'miles per hour')
print("hello")

timePerMile(distMi, seconds)
avgSpeed(distMi, hours)
```

2. **The volume of a sphere with radius r is 5? (Use Sphere volume formula)**

```
# Define the value of pi
pi = 3.1415926535897931

# Define the radius of the sphere
r = 5.0

# Calculate the volume of the sphere using the formula
V = 4.0/3.0 * pi * r**3

# Print the calculated volume of the sphere
   print('The volume of the sphere is: ', V)
```

3. **Suppose the cover price of a book is $24.95, but bookstores get a 40% discount. Shipping costs $3 for the first copy and 75 cents for each additional copy. What is the total wholesale cost for 60 copies?**

```
bookPrice = 24.95
discount = .40
shippingPriceRest = .75
shippingPriceFirst = 3.00
totalUnits = 60

bookDiscountAmount = bookPrice * discount * totalUnits
shipping = shippingPriceRest * 59 + shippingPriceFirst

result = bookDiscountAmount + shipping


print('The total price for 60 books including shipping and discount is: ')
print('Total price of the books is: ' + str(bookDiscountAmount))
print('Total Shipping is: ' + str(shipping))
print('The Total price is: ' + str(result))
```

4. **A function object is a value you can assign to a variable or pass as an argument. For example, do_twice is a function that takes a function object as an argument and calls it twice:**

```
def do_twice(f):
f()
```

f()

Here's an example that uses do_twice to call a function named print_spam twice.

def print_spam():

print 'spam'

do_twice(print_spam)

a. Type this example into a script and test it.

b. Modify do_twice so that it takes two arguments, a function object and a value, and calls the function twice, passing the value as an argument.

c. Write a more general version of print_spam, called print_twice, that takes a string as a parameter and prints it twice.

d. Use the modified version of do_twice to call print_twice twice, passing 'spam'as an argument.

```
arg = 'spam'

def do_twice(f, arg):
    f(arg)
    f(arg)

def print_twice(arg):
    print(arg)
    print(arg)

do_twice(print_twice, arg)
print()

def do_four(f, arg):
    do_twice(f, arg)
    do_twice(f, arg)

      do_four(print_twice, arg)
```

5. Write a function that draws a grid like the following:

```
def my_func1():
    print("+", 4*'-', '+', 4*'-', '+', 4*'-', '+')

def my_func2():
    for x in range(4):
        print('|', 4*' ', '|', 4*' ', '|', 4*' ', '|')


def total():
    my_func1()
    my_func2()
    my_func1()
    my_func2()
    my_func1()
    my_func2()
    my_func1()


   total()
```

6. **Write a function called gcd that takes parameters a and b and returns their greatest common divisor.**

```
import math
print(" GCD of two number 0 and 0 is ", math.gcd(0, 0)) #math.gcd(a, b), a and b
are the two integer number
print(" GCD of two number 0 and 48 is ", math.gcd(0, 48))
a = 60 # assign the number to variable a
b = 48 # assign the number to variable b
print(" GCD of two number 60 and 48 is ", math.gcd(a, b)) # pass the variable a and
b to math.gcd() function.
print(" GCD of two number 48 and -12 is ", math.gcd(48, -12)) # pass the integer
number
print(" GCD of two number -24 and -18 is ", math.gcd(-24, -18))
      print(" GCD of two number -60 and 48 is ", math.gcd(-60, 48))
```

7. **Write a function called is_palindrome that takes a string argument and returns True if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string.**

```
def is_palindrome(s):
    if len(s) < 1:
        return True
    else:
        if s[0] == s[-1]:
            return is_palindrome(s[1:-1])
        else:
            return False
a=str(input("Enter string:"))
if(is_palindrome(a)==True):
    print("String is a palindrome!")
else:
          print("String isn't a palindrome!")
```

8. **Write a function called is_sorted that takes a list as a parameter and returns True if the list is sorted in ascending order and False otherwise.**

```
test_list = [1, 4, 5, 8, 10]

# printing original list
print ("Original list : " + str(test_list))

# using naive method to
# check sorted list
flag = 0
i = 1
while i < len(test_list):
    if(test_list[i] < test_list[i - 1]):
        flag = 1
    i += 1

# printing result
if (not flag) :
    print ("Yes, List is sorted.")
else :
          print ("No, List is not sorted.")
```

9. **Write a function called has_duplicates that takes a list and returns True if there is any element that appears more than once. It should not modify the original list.**

```
def has_duplicates(lst):
    seen = set()
    for element in lst:
        if element in seen:
            return True
        seen.add(element)
       return False
```

10. **Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order**

```
def has_duplicates(lst):
    seen = set()
    for element in lst:
        if element in seen:
            return True
        seen.add(element)
          return False
```

11. **The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", and the empty string.**

```
def add_specific_words(file_path):
    # Read words from the file
    with open(file_path, 'r') as file:
        words = file.read().splitlines()

    # Add "I", "a", and the empty string to the list
    words.extend(["I", "a", ""])

    return words

# Path to the words.txt file
file_path = 'words.txt'

# Add specific words and print the result
words_with_additions = add_specific_words(file_path)
print(words_with_additions)
```

12. **Write a python code to read a dictionary values from the user. Construct a function to invert its content. i.e., keys should be values and values should be keys.**

```
def invert_dict(d):
    return {v: k for k, v in d.items()}

def read_dict():
    n = int(input("Enter the number of items in the dictionary: "))
    d = {}
    for _ in range(n):
        key = input("Enter key: ")
        value = input("Enter value: ")
        d[key] = value
    return d
# Read dictionary from user
user_dict = read_dict()

# Invert the dictionary
inverted_dict = invert_dict(user_dict)

# Print the inverted dictionary
```

```
print("Inverted dictionary:", inverted_dict)
```

15. If there are 23 students in a class what are the chances that 2 of you have the same birthday? You can estimate this probability by generating random samples of 23 Birthdays and checking for matches.

```
import math
from decimal import Decimal
n=int(input('No of Students: '))
num=Decimal(math.factorial(365))/Decimal(math.factorial(365-n))
den=365**n
p=1-((num)/(den))
print(f'Probability of at least one matching birthday: {p:.4f}')
```

22. a. Write a function called draw_reactngle that takes a canvas and the Rectangle as arguments and draws a representation of rectangle on the canvas.

```
import turtle


def draw_rectangle(canvas, rectangle):
    # Initialize turtle screen
    screen = turtle.Screen()
    screen.title("Rectangle Drawing")


    # Initialize turtle for drawing
    t = turtle.Turtle()


    # Set turtle speed and pen size
    t.speed(1)  # Speed: 1 (slowest), 10 (fastest)
    t.pensize(2)


    # Move turtle to starting position of rectangle
    t.penup()
    t.goto(rectangle['x'], rectangle['y'])
    t.pendown()


    # Draw rectangle
    t.forward(rectangle['width'])
    t.left(90)
    t.forward(rectangle['height'])
    t.left(90)
    t.forward(rectangle['width'])
    t.left(90)
```

```
    t.forward(rectangle['height'])
    t.left(90)

    # Hide turtle and exit on click
    t.hideturtle()
    screen.exitonclick()

# Example usage:
canvas = turtle.Screen()
rectangle = {'x': -50, 'y': 50, 'width': 100, 'height': 80}
draw_rectangle(canvas, rectangle)
```

22. b. Add an attribute named color to your rectangle objects and modify draw_rectangle so that it uses color attribute as the fill color.

```
import turtle

def draw_rectangle(canvas, rectangle):
    # Initialize turtle screen
    screen = turtle.Screen()
    screen.title("Rectangle Drawing")

    # Initialize turtle for drawing
    t = turtle.Turtle()

    # Set turtle speed and pen size
    t.speed(1)  # Speed: 1 (slowest), 10 (fastest)
    t.pensize(2)

    # Move turtle to starting position of rectangle
    t.penup()
    t.goto(rectangle['x'], rectangle['y'])
    t.pendown()

    # Set fill color based on rectangle's color attribute
    t.fillcolor(rectangle['color'])
```

```
    t.begin_fill()

    # Draw rectangle
    for _ in range(2):
        t.forward(rectangle['width'])
        t.left(90)
        t.forward(rectangle['height'])
        t.left(90)

    t.end_fill()

    # Hide turtle and exit on click
    t.hideturtle()
    screen.exitonclick()

# Example usage:
canvas = turtle.Screen()

# Define rectangle with color attribute
rectangle = {'x': -50, 'y': 50, 'width': 100, 'height': 80, 'color': 'blue'}

draw_rectangle(canvas, rectangle)
```

22. c. Write a function called draw_pointthat takes a canvas and a point as a argument and draws a rep of point on canvas.

```
import turtle

def draw_point(canvas, point):
    # Initialize turtle screen
    screen = turtle.Screen()
    screen.title("Point Drawing")

    # Initialize turtle for drawing
    t = turtle.Turtle()
```

```
    # Set turtle speed and pen size
    t.speed(1)  # Speed: 1 (slowest), 10 (fastest)
    t.pensize(2)

    # Move turtle to the point's position
    t.penup()
    t.goto(point['x'], point['y'])
    t.pendown()

    # Draw a small dot to represent the point
    t.dot(5)  # Diameter of dot: 5 pixels

    # Hide turtle and exit on click
    t.hideturtle()
    screen.exitonclick()

# Example usage:
canvas = turtle.Screen()

# Define point with x and y coordinates
point = {'x': 50, 'y': 50}

draw_point(canvas, point)
```

D. Define a new class called Circle with appropriate attributes and instantiate a few Circle objects. Write a function called draw_circle that draws circle on the canvas.

```
import turtle

class Circle:
    def __init__(self, x, y, radius, color='black'):
        self.x = x
        self.y = y
        self.radius = radius
        self.color = color
```

```
def draw_circle(canvas, circle):
    # Initialize turtle for drawing
    t = turtle.Turtle()
    t.speed(1)  # Speed: 1 (slowest), 10 (fastest)
    t.pensize(2)

    # Move turtle to the circle's position
    t.penup()
    t.goto(circle.x, circle.y - circle.radius)
    t.pendown()

    # Set fill color based on circle's color attribute
    t.fillcolor(circle.color)
    t.begin_fill()

    # Draw circle
    t.circle(circle.radius)

    t.end_fill()

    # Hide turtle
    t.hideturtle()

# Example usage:
canvas = turtle.Screen()

# Define circle objects with x, y, radius, and color attributes
circle1 = Circle(x=0, y=0, radius=50, color='blue')
circle2 = Circle(x=-100, y=50, radius=30, color='red')
circle3 = Circle(x=100, y=-50, radius=40, color='green')

# Draw circles
draw_circle(canvas, circle1)
draw_circle(canvas, circle2)
draw_circle(canvas, circle3)
```

```
# Keep the screen open until clicked

canvas.exitonclick()
```

23. Write a Python program to demonstrate the usage of MRO in multiple levels of Inheritances.

```
class Human:
    def species(self):
        print("Hello I am a Human")
class Teacher(Human):
    def  job(self):
        print("Hello I am a Teacher")
    def work(self):
        print("I teach math")
class Doctor(Human):
    def job(self):
        print("Hello I am a doctor")
class humanity(Doctor,Teacher):
    pass
h=humanity()
h.job()
h.species()
```

24. Write a python code to read a phone number and email-id from user and validate it for correections.

```
p=input('Enter a phone number: ')
if p.isdigit() and len(p)==10:
        print('Phone number is valid.')
else:
        print('Phone number is invalid.')
m=input('Enter an email address: ')
if m.endswith('.com') and '@' in m :
        print('Email address is valid.')
else:
        print('Email address is invalid.')
```

25. Write a python code to merge two given file into third file.

```
f1=open('E:\\word\\txt1.txt','r')
f2=open('E:\\word\\txt2.txt','r')
s1=f1.read()
s2=f2.read()
s3=s1+"  "+s2
f3=open('E:\\word\\output.txt','w')
f3.write(s3)
print("Operation Success")
```

26. Write a Python code to open a give file and construct a function to check for given words present in it and display on found.

```
def chkwrds(s,chk):
    s=s.split()
    for i in s:
        if i in chk:
```

```
        print(f'Found word {i}')
f=open('E:\\word\\note.txt','r')
s=f.read()
l=["hi","how","what","where"]
chkwrds(s,l)
```