Calcutta Institute of Engineering and Management

Department of Information Technology

IT Workshop (Python) - PCC-CS393

NAME:	DIBBYARUP DUTTA
ROLL NO.:	16500221005
ASSIGNMENT:	WEEK 2
DATE OF ASSIGNMENT:	30th July, 2022
DATE OF SUBMISSION:	

Signature of the Faculty with Date

Question-1

Write a Python function to find if a number is even or odd. Implement the function in a python program.

CODE:

```python
def evenOrOdd(number):
    if (number&1) == 1:
        print("Number is Odd.")
    else:
        print("Number is Even")

num = int(input("Enter a number: "))
evenOrOdd(num)
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python even_odd.py
Enter a number: 13
Number is Odd.
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python even_odd.py
Enter a number: 24
Number is Even
```

Question-2

Write a Python function to find if which is the greater number of two numbers. Implement the function in a python program.

CODE:

```python
def findGreaterNumber(number1, number2):
    if number1 > number2:
        print(f"{number1} is greater.")
    else:
        print(f"{number2} is greater.")

num1, num2 = input("Enter two numbers: ").split()
findGreaterNumber(int(num1), int(num2))
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python greater_number.py
Enter two numbers: 47 113
113 is greater.
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python greater_number.py
Enter two numbers: 12 11
12 is greater.
```

Question-3

Write a Python function to prove Pascal's principal. a/b = c/d.

CODE:

```
def isPascalFraction(a, b, c, d):
    if a*d == c*b:
        print("Pascal's Fractions")
    else:
        print("Not Pascal's Fractions")

frac_1, frac_2 = input("Enter two fractions: ").split()
a, b = list(map(int, frac_1.split("/")))
c, d = list(map(int, frac_2.split("/")))
isPascalFraction(a, b, c, d)
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python pascals_principle.py
Enter two fractions: 1/2 2/4
Pascal's Fractions
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python pascals_principle.py
Enter two fractions: 3/5 2/3
Not Pascal's Fractions
```

Question-4

Write a Python function to find if the given year is leap year or not. Implement the function.

CODE:

```python
def isLeapYear(year):
    if year%4 == 0:
        if year%100 == 0:
            if year%400 == 0:
                print("It is a Leap Year.")
            else:
                print("It is not a Leap Year.")
        else:
            print("It is a Leap Year.")
    else:
        print("It is not a Leap Year.")

year = int(input("Enter the year: "))
isLeapYear(year)
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python leap_year.py
Enter the year: 2016
It is a Leap Year.
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python leap_year.py
Enter the year: 1900
It is not a Leap Year.
```

Question-5

Write a Python function to find if the roots of a quadratic equation are imaginary, real or equal.

CODE:

```python
from cmath import sqrt

def quadraticRoots(a, b, c):
    d = sqrt((b ** 2) - 4 * a * c)
    r_1 = (- b + d) / (2 * a)
    r_2 = (- b - d) / (2 * a)
    print(f"Roots are {r_1}, {r_2}")

a, b, c = list(map(int, input("Enter a, b and c: ").split()))
quadraticRoots(a, b, c)
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python roots_of_quadratic_eq.py
Enter a, b and c: 2 -11 5
Roots are (5+0j), (0.5+0j)
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python roots_of_quadratic_eq.py
Enter a, b and c: 2 -8 16
Roots are (2+2j), (2-2j)
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python roots_of_quadratic_eq.py
Enter a, b and c: 1 -4 5
Roots are (2+1j), (2-1j)
```

Bonus Question-1

In deep.py, implement a program that prompts the user for the answer to the Great Question of Life,the Universe and Everything, outputting Yes if the user inputs 42 or (case-insensitively) forty-two or forty two . Otherwise output No.

CODE:

```
def isCorrect(answer):

    if answer in ['42', 'forty-two', 'forty two']:
        print("Yes")
    else:
        print("No")

answer = input("What is answer to the Life, the Universe and Everything? ")
isCorrect(answer.lower())
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python deep.py
What is answer to the Life, the Universe and Everything? 42
Yes
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python deep.py
What is answer to the Life, the Universe and Everything? forty-two
Yes
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python deep.py
What is answer to the Life, the Universe and Everything? Forty Two
Yes
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python deep.py
What is answer to the Life, the Universe and Everything? peace
No
```

Bonus Question-2

Kramer visits a bank that promises to give $100 to anyone who isn’t greeted with a “hello.” Kramer is instead greeted with a “hey,” which he insists isn’t a “hello,” and so he asks for $100. The bank’s manager proposes a compromise: “You got a greeting that starts with an ‘h,’ how does $20 sound?” Kramer accepts.

In a file called bank.py , implement a program that prompts the user for a greeting. If the greeting starts with “hello”, output $0 . If the greeting starts with an “h” (but not “hello”), output $20. Otherwise, output $100 . Ignore any leading whitespace in the user’s greeting, and treat the user’s greeting case-insensitively.

CODE:

```python
def moneyGiveaway(greeting):
    if greeting == "hello":
        print("$0")
    elif greeting[0] == 'h':
        print("$20")
    else:
        print("$100")

greeting = input("Enter the greeting by the bank: ")
moneyGiveaway(greeting.replace(" ","").lower())
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python bank.py
Enter the greeting by the bank: Hello
$0
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python bank.py
Enter the greeting by the bank: hey
$20
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python bank.py
Enter the greeting by the bank: welcome
$100
```

Bonus Question-3

Python already supports math, whereby you can write code to add, subtract, multiply, or divide values and even variables. But let’s write a program that enables users to do math, even without knowing Python.

In a file called interpreter.py, implement a program that prompts the user for an arithmetic expression and then calculates and outputs the result as a floating-point value formatted to one decimal place. Assume that the user’s input will be formatted as x y z, with one space between x and y and one space between y and z, wherein:

- x is an integer
- y is + , - , * , or /
- z is an integer.

For instance, if the user inputs 1 + 1, your program should output 2.0. Assume that, if y is /, then z will not be 0.
Note that, just as python itself is an interpreter for Python, so will your interpreter.py be an interpreter for math!

CODE:

```
def operation(x, y, z):
    if y == '+':
        r = x + z
        print("{:.1f}".format(r))
    elif y == '-':
        r = x - z
        print("{:.1f}",format(r))
    elif y == '*':
        r = x * z
        print("{:.1f}".format(r))
    elif y == '/':
        if z != 0:
            r = x / z
            print("{:.1f}".format(r))
        else:
            print("Can't divide by Zero.")
    else:
        print("Invalid Input.")

x, y, z = input("Enter the math problem: ").split()
operation(float(x), y, float(z))
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python interpreter.py
Enter the math problem: 45 / 3
15.0
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python interpreter.py
Enter the math problem: 15 / 0
Can’t Divide by Zero.
```

Bonus Question-4

Suppose that you're in a country where it's customary to eat breakfast between 7:00 and 8:00, lunch between 12:00 and 13:00, and dinner between 18:00 and 19:00. Wouldn't it be nice if you had a program that could tell you what to eat when?

In meal.py, implement a program that prompts the user for a time and outputs whether it's breakfast time, lunch time, or dinner time. If it's not time for a meal, don't output anything at all. Assume that the user's input will be formatted in 24-hour time as #:## or ##:##. And assume that each meal's time range is inclusive. For instance, whether it's 7:00, 7:01, 7:59, or 8:00, or anytime in between, it's time for breakfast. Structure your program per the below, wherein convert is a function (that can be called by main) that converts time, a str in 24-hour format, to the corresponding number of hours as a float.
For instance, given a time like "7:30" (i.e., 7 hours and 30 minutes), convert should return 7.5 (i.e., 7.5 hours).

```
def main():
...
def convert(time):
...
if __name__ == "__main__":
main()
```

CODE:

```python
def convert(time):
    h, m = [float(i) for i in time.split(":")]
    m /= 60
    return h + m

def main():
    time_str = input("Enter time (HH:MM): ")
    time = convert(time_str)

    if 7 <= time <= 8:
        print("Breakfast time.")
    elif 12 <= time <= 13:
        print("Lunch time.")
    elif 18 <= time <= 19:
        print("Dinner time.")
main()
```

OUTPUT:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python meal.py
Enter time (HH:MM):18:47
Dinner time.
PS F:\COM.PROG.DOCS\CODE FILES\Lang\Python\Week 2> python meal.py
Enter time (HH:MM):13:07
```