CS10 Lab 3 (Lesson 1 to Lesson 40)
(5 points total 1 point for each question)
3 test runs for each question
(Functions and Strings)

1. Follow the Lab instructions and video on how to submit labs.
2. Follow the steps in the Lab instructions to separate each question with question number, short description of the question.
3. For each question you must provide output for 3 test runs (use the sample test run data already provided plus the rest yourself). For each question that do not have 3 test runs a zero will be given even if the program runs correctly.
4. You must use the data in the sample test runs that are given in the question. Provide your own data whenever there is no sample run data.
5. If the program does not run a zero will be given
6. Your test run must be exactly the same as the sample test run specifications provided. If the program runs but does not fulfill all the specifications stated in the question, a zero score will be given.
7. Use topics covered in class up till functions and strings(Lesson 1 to Lesson 40). Using global variables and topics or tools not in Lesson 1 to Lesson 40 will result in a zero for that question (no lists, tuples, sets, dictionary)

   ** all functions must include docstrings and type hints(annotations). See lesson 29 and 30 on docstrings and type hints. Programs with functions must have the statement at the end of the program before calling the main() function.

```
if __name__ == "__main__":
    main()
```

NO GLOBAL VARIABLES ALLOWED FOR THIS LAB, a zero will be given to any program with global variable. You are not allowed to use *break or continue* in your programs. If there is a break or continue command in your program a zero will be given for that question.

1. Write a program with functions (no user input required):
   a. Create a function texas() where:
      - birds = 5000
      - print the string "**Texas has 5000 birds**" (without the quotes ")

   b. create a function California() where:
      - birds = 8000
      - print the string "**California has 8000 birds**" (without the quotes ")

   c. create a function main() to call the two functions

Sample test runs:

```
#Test run 1
#Texas has 5000 birds
#California has 8000 birds

#Test run 2
#Texas has 5000 birds
#California has 8000 birds

#Test run 3
#Texas has 5000 birds
#California has 8000 birds
```

2. a. write a function *show_interest()* to take in 3 parameters with ***default arguments*** rate = 0.01, period = 10, principal = 10000.00 (no user input required):
   - interest = principal * rate * periods
   - print the following string

     The simple interest will be $1,000.00

   b. write the main() function to call *show_interest*:

sample test runs:

```
#Test run 1
#The simple interest will be $1,000.00

#Test run 2
#The simple interest will be $1,000.00

#Test run 3
#The simple interest will be $1,000.00
```

3. Write a program to find the area of a right-angled triangle using functions.

   a. Write a function getData() for user to input the base and the perpendicular height of a triangle.
   b. Write a function trigArea() to calculate the area of a triangle.
   c. Write a function displayData() to print the base, height, and the area of a triangle
      See sample test runs
   d. Write the main() function to call getData(), call trigArea() and call displayData().

Sample test runs(**blue user input**):

```
#Test run 1
##Enter the base of your triangle: 10
##Enter the height of your triangle: 5
##The base, height, and area are 10.00 , 5.00 , 25.00

#Test run 2
##Enter the base of your triangle: 2
##Enter the height of your triangle: 3
##The base, height, and area are 2.00 , 3.00 , 3.00

#Test run 3
(provide your own data)
```

4. The main() function template is provided below. You must use the main function template to write the functions that are called in the main() function to produce these outputs. Your test run output must be exactly the same as specified below or a zero will be given for this question.

Sample test runs (user input in blue)

Test run 1
Enter the monthly sales: **14550.00**
Enter the amount of advanced pay, or
enter 0 if no advanced pay was given.
Advanced pay: **1000.00**
The pay is $746.00

Test run 2
Enter the monthly sales: **9500**
Enter the amount of advanced pay, or
enter 0 if no advanced pay was given.
Advanced pay: **0**
The pay is $950.00

Test run 3
Enter the monthly sales: **12000.00**
Enter the amount of advanced pay, or
enter 0 if no advanced pay was given.
Advanced pay: **2000.00**
The pay is $-560.00
The salesperson must reimburse
the company.

Here is the template for the main() function, you must use this main() template, copy and paste into Python IDLE. Requirements for the `determine_comm_rate()` is found below the template. You are not allowed to change anything in the main function template (add, delete or change). Changing any single piece of code in the given main function template will result in a zero for this question. The main() function must be placed at the bottom after all other functions.

```
# This program calculates a salesperson's pay

def main():
    # Get the amount of sales from user
    sales = get_sales()

    # Get the amount of advanced pay from user.
    advanced_pay = get_advanced_pay()

    # Determine the commission rate.
    comm_rate = determine_comm_rate(sales)

    # Calculate the pay.
    pay = sales * comm_rate - advanced_pay

    # Display the amount of pay.
    print('The pay is $', format(pay, ',.2f'), sep='')

    # Determine whether the pay is negative.
    if pay < 0:
        print('The salesperson must reimburse')
        print('the company.')
```

Q4 continued…

Requirements for the function - **`determine_comm_rate()`**

	rate
Sales less than 10000.00	.10
Sales from 10000.00 to 14999.99	.12
Sales from 15000.00 to 17999.99	.14
Sales from 18000.00 to 21999.99	.16
Sales greater than 21999.99	.18

5. Write a program that asks the user to enter a single-digit number sequentially with nothing separating them. The single-digit number must be inputted as a string not an int or float. Then display the sum of all the single digit numbers in the single-digit string.

Sample test runs:

```
#Test run 1
##Enter a sequence of digits with nothing separating them: 4563
##The total of the digits in the string you entered is 18

#Test run 2
##Enter a sequence of digits with nothing separating them: 653214
##The total of the digits in the string you entered is 21

#Test run 3
(provide your own data)
```

The main() function template is provided for you below. Write the codes for the function string_total(). You are not allowed to change anything in the main function template (add, delete or change). Changing any single piece of code in the given main function template will result in a zero for this question. The main() function must be placed at the bottom after all other functions.

```
def main():
    # Get a string of numbers as input from the user.
    number_string = input('Enter a sequence of digits with nothing separating them: ')

    # Call string_total function, and store the total.
    total = string_total(number_string)

    # Display the total.
    print('The total of the digits in the string you entered is', total)
```