

Assignment – 2

Name: Anupam Kumar Khamrai

Student Id: 241001271118

Department & Section: MCA 1B

Subject: Python

Semester: 1st

Year: 2024

1. To determine the profit or loss on sale.

Source code:-

```
cost_price, selling_price = eval(input("Enter cost price and selling price separated by a comma: "))
```

```
if selling_price > cost_price:
```

```
    profit = selling_price - cost_price
```

```
    print("Profit:", profit)
```

```
elif selling_price < cost_price:
```

```
    loss = cost_price - selling_price
```

```
    print("Loss:", loss)
```

```
else:
```

```
    print("No profit, no loss.")
```

Output:-

```
Microsoft Windows [Version 10.0.19045.5073]
(c) Microsoft Corporation. All rights reserved.

C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/1.py
Enter cost price and selling price separated by a comma: 150,250
Profit: 100

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

2. To determine greater between two numbers.

Source code:-

```
num1, num2 = eval(input("Enter two numbers separated by a comma: "))  
  
if num1 > num2:  
    print("Greater number:", num1)  
  
elif num2 > num1:  
    print("Greater number:", num2)  
  
else:  
    print("Both numbers are equal.")
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/2.py  
Enter two numbers separated by a comma: 4,8  
Greater number: 8  
  
C:\Users\khamr\Desktop\Python\Theory\2nd>
```

3. To determine the absolute difference between a given number and 123 and if it is greater than 123 then print the triple of the given number.

Source code:-

```
number = eval(input("Enter a number: "))  
difference = abs(number - 123)  
print("Absolute Difference:", difference)  
if difference > 123:  
    print("Triple of the number:", number * 3)
```

Output:-

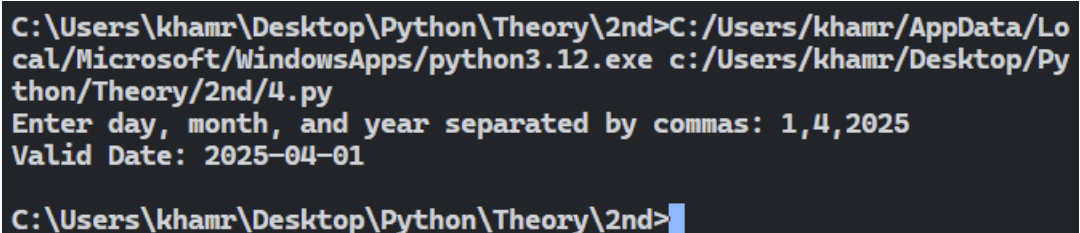
```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/3.py  
Enter a number: 458  
Absolute Difference: 335  
Triple of the number: 1374  
  
C:\Users\khamr\Desktop\Python\Theory\2nd>
```

4. To validate a given date.

Source code:-

```
day, month, year = eval(input("Enter day, month, and year separated by commas: "))
import datetime
try:
    date = datetime.date(year, month, day)
    print("Valid Date:", date)
except ValueError:
    print("Invalid Date")
```

Output:-



```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/4.py
Enter day, month, and year separated by commas: 1,4,2025
Valid Date: 2025-04-01

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

5. To determine whether a quadrilateral is a square, rhombus, parallelogram, square or irregular on the basis of only the lengths of all the sides and one internal angle.

Source code:-

```
side1, side2, side3, side4, angle = eval(input("Enter four sides and one internal angle separated by commas: "))
```

```
if side1 == side2 == side3 == side4:
```

```
    if angle == 90:
```

```
        print("Square")
```

```
    else:
```

```
        print("Rhombus")
```

```
elif side1 == side3 and side2 == side4:
```

```
    if angle == 90:
```

```
        print("Rectangle")
```

```
    else:
```

```
        print("Parallelogram")
```

```
else:
```

```
    print("Irregular Quadrilateral")
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/5.py
Enter four sides and one internal angle separated by commas: 4,4,4,4,90
Square

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

6. To print a currency conversion table from Pounds, Dollar, Euro to equivalent Indian Rupees.

Source code:-

```
pound_to_inr = 101.58
dollar_to_inr = 74.32
euro_to_inr = 87.67
amount = eval(input("Enter the amount in Pounds, Dollar, and Euro separated by commas: "))
pounds, dollars, euros = amount
print("INR equivalent of Pounds:", pounds * pound_to_inr)
print("INR equivalent of Dollars:", dollars * dollar_to_inr)
print("INR equivalent of Euros:", euros * euro_to_inr)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/6.py
Enter the amount in Pounds, Dollar, and Euro separated by commas:
1,4,5
INR equivalent of Pounds: 101.58
INR equivalent of Dollars: 297.28
INR equivalent of Euros: 438.35

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

7. To print the number of days in each month of a given year.

Source code:-

```
year = eval(input("Enter the year: "))

months = [("January", 31), ("February", 29 if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0) else 28), ("March", 31), ("April", 30), ("May", 31), ("June", 30), ("July", 31), ("August", 31), ("September", 30), ("October", 31), ("November", 30), ("December", 31)]

for month, days in months:

    print(f"{month}: {days} days")
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/7.py
Enter the year: 2029
January: 31 days
February: 28 days
March: 31 days
April: 30 days
May: 31 days
June: 30 days
July: 31 days
August: 31 days
September: 30 days
October: 31 days
November: 30 days
December: 31 days

C:\Users\khamr\Desktop\Python\Theory\2nd>
```


8. To calculate commission of a salesman when the calculation of commission is based on the rules given below:

(a) Nil, when the sales <10000 in region A.

(b) 6.5% of the sales if the sales <15000 in region B and <16000 in region A.

(c) 8.5% plus Rs.1500, if the sales ≥ 15000 but <25000 in region B and ≥ 16000 but <35000, in region A.

(d) 11% of the sales plus Rs.4500 for all regions for all other cases.

Source code:-

```
sales_a, sales_b = eval(input("Enter sales in region A and region B separated by a comma: "))
if sales_a < 10000:
    commission = 0
    print("Commission: Nil (No commission)")
elif sales_b < 15000 and sales_a < 16000:
    commission = 0.065 * (sales_a + sales_b)
    print("Commission:", commission)
elif 15000 <= sales_b < 25000 and 16000 <= sales_a < 35000:
    commission = 0.085 * (sales_a + sales_b) + 1500
    print("Commission:", commission)
else:
    commission = 0.11 * (sales_a + sales_b) + 4500
    print("Commission:", commission)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/8.py
Enter sales in region A and region B separated by a comma: 40000,20000
Commission: 11100.0

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

9. To categorize a triangle on the basis of its three given angles. The triangle may be in an 'invalid triangle', 'Equiangular', 'right-angle', 'acute angled' or 'obtuse angled'.

Source code:-

```
angle1, angle2, angle3 = eval(input("Enter the three angles of the triangle separated by commas: "))
```

```
if angle1 + angle2 + angle3 != 180:
```

```
    print("Invalid triangle")
```

```
elif angle1 == angle2 == angle3 == 60:
```

```
    print("Equiangular triangle")
```

```
elif 90 in [angle1, angle2, angle3]:
```

```
    print("Right-angle triangle")
```

```
elif all(angle < 90 for angle in [angle1, angle2, angle3]):
```

```
    print("Acute-angled triangle")
```

```
else:
```

```
    print("Obtuse-angled triangle")
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/9.py
Enter the three angles of the triangle separated by commas: 30,60,90
Right-angle triangle

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

10. To accept the lengths of three sides of a triangle to check whether the given lengths can be valid lengths of three sides of a triangle and to categorize the triangle as 'Equilateral' or 'Isosceles' or 'Scalene' one.

Source code:-

```
side1, side2, side3 = eval(input("Enter the three sides of the triangle separated by commas: "))
```

```
if side1 + side2 > side3 and side1 + side3 > side2 and side2 + side3 > side1:
```

```
    if side1 == side2 == side3:
```

```
        print("Equilateral triangle")
```

```
    elif side1 == side2 or side1 == side3 or side2 == side3:
```

```
        print("Isosceles triangle")
```

```
    else:
```

```
        print("Scalene triangle")
```

```
else:
```

```
    print("Invalid triangle sides")
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/10.py
Enter the three sides of the triangle separated by commas: 7,3,5
Scalene triangle

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

11. To compute the festival bonus of the employees on the basis of the basic pay and the designation as per the following rules:

Basic Pay	Designation	Percentage of basic pay payable
<40,000	Manager	12% of the basic pay subject to minimum of Rs. 2500
>=40,000	Manager	16% of the basic pay subject to maximum of Rs. 7500
<20000	Officer	14% of the basic pay subject to a minimum of Rs.2500 and a maximum of Rs.5000.
For all others cases	Whatever	8.9% of basic pay

Source code:-

```
basic_pay, designation = input("Enter basic pay and designation (Manager or Officer)
separated by a comma: ").split(",")

basic_pay = float(basic_pay)

designation = designation.strip()

if designation == "Manager":

    if basic_pay < 40000:

        bonus = max(0.12 * basic_pay, 2500)

    else:

        bonus = min(0.16 * basic_pay, 7500)

elif designation == "Officer" and basic_pay < 20000:

    bonus = max(0.14 * basic_pay, 2500)

    bonus = min(bonus, 5000)

else:

    bonus = 0.089 * basic_pay

print("Festival Bonus:", bonus)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/11.py
Enter basic pay and designation (Manager or Officer) separated by a comma: 25000,Officer
Festival Bonus: 2225.0

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

12. For the first 75 calls, the charge is fixed and it is equal to Rs. 75; for the next 75 calls, the charge is calculated @Rs 0.75 per call; for the next 90 calls, the charge is Rs.0.65 per call and for the rest, if any, the rate is Rs.0.55 per call. It is required to determine the monthly bill of a subscriber.

Source code:-

```
calls = eval(input("Enter the total number of calls: "))
```

```
bill = 0
```

```
if calls <= 75:
```

```
    bill = 75
```

```
elif calls <= 150:
```

```
    bill = 75 + (calls - 75) * 0.75
```

```
elif calls <= 240:
```

```
    bill = 75 + 75 * 0.75 + (calls - 150) * 0.65
```

```
else:
```

```
    bill = 75 + 75 * 0.75 + 90 * 0.65 + (calls - 240) * 0.55
```

```
print("Monthly Bill:", bill)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/12.py
Enter the total number of calls: 83
Monthly Bill: 81.0

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

13. In a certain country, the difference between two consecutive gas meter readings gives the amount of gas consumed in cubic feet. It is then multiplied by a factor 1.475 to convert it into the number of therms used by the consumer, where therm is the unit of billing. The meter readings are collected at the end of each month. The following rate chart is then used by the gas company to calculate the bill amount of each consumer:

No. of therms used	Rate per therm
≤ 125	Rs.7.75
>125 but ≤ 250	Rs. 9.75 plus a surcharge of 1.25% over the calculated charge;
>250	Rs. 13.00 plus a surcharge of 2.5% over the calculated charge.

A meter rent of Rs.25 is also added to the gas charges of each consumer to determine the gas bill. If the gas meters display 8-digit readings, then develop a script in Python is required to determine the monthly gas bill of the consumers.

Source code:-

```
previous_reading, current_reading = eval(input("Enter previous and current meter readings separated by a comma: "))
units_consumed = (current_reading - previous_reading)
if units_consumed < 0:
    units_consumed = (10**8 - previous_reading) + current_reading
therms_used = int(units_consumed * 1.475)
meter_rent = 25
if therms_used <= 125:
    rate_per_therm = 7.75
    surcharge = 0
elif therms_used <= 250:
    rate_per_therm = 9.75
    surcharge = 0.0125 * rate_per_therm
else:
    rate_per_therm = 13.00
    surcharge = 0.025 * rate_per_therm
total_cost = therms_used * (rate_per_therm + surcharge) + meter_rent
print("Gas Bill:", total_cost)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/13.py
Enter previous and current meter readings separated by a comma: 9999999,100
Gas Bill: 1486.0375

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

14. In a certain area, the parking charge for cars is calculated according to the following rules: For the first 8.5 hours or part thereof, the rate is fixed and it is equal to Rs. 55; for the next every 2 hours or part thereof up to a maximum of 23 hours, the rate Rs. 13.75; beyond the 23 hours limit, the charge Rs. 5.50 for every minute. Develop a Python script to calculate the parking charge for the users of the area.

Source code:-

```
total_hours = float(input("Enter the total number of hours the car was parked: "))
charge = 0
if total_hours <= 8.5:
    charge = 55
else:
    charge = 55
    remaining_hours = total_hours - 8.5
    additional_blocks = (remaining_hours + 1.99) // 2
    charge += additional_blocks * 13.75
    if total_hours > 23:
        extra_time_in_minutes = (total_hours - 23) * 60
        charge += extra_time_in_minutes * 5.50
print("The total parking charge is Rs.", charge)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/14.py
Enter the total number of hours the car was parked: 40
The total parking charge is Rs. 5885.0

C:\Users\khamr\Desktop\Python\Theory\2nd>
```


15. Develop a Python script to determine the minimum number of 100 rupee notes, 200 rupee notes, 500 rupee notes and 2000 rupee notes required to dispense a given sum of money.

Source code:-

```
amount = eval(input("Enter the amount of money: "))
notes_2000 = notes_500 = notes_200 = notes_100 = 0
if amount >= 2000:
    notes_2000 = amount // 2000
    amount = amount % 2000
if amount >= 500:
    notes_500 = amount // 500
    amount = amount % 500
if amount >= 200:
    notes_200 = amount // 200
    amount = amount % 200
if amount >= 100:
    notes_100 = amount // 100
    amount = amount % 100
print("2000 rupee notes:", notes_2000)
print("500 rupee notes:", notes_500)
print("200 rupee notes:", notes_200)
print("100 rupee notes:", notes_100)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/15.py
Enter the amount of money: 15000
2000 rupee notes: 7
500 rupee notes: 2
200 rupee notes: 0
100 rupee notes: 0

C:\Users\khamr\Desktop\Python\Theory\2nd>
```

16. It is required to print the name of the starting day of any given year.

Source code:-

```
import datetime  
  
year = eval(input("Enter the year: "))  
  
starting_day = datetime.datetime(year, 1, 1).strftime("%A")  
  
print("The starting day of the year", year, "is:", starting_day)
```

Output:-

```
C:\Users\khamr\Desktop\Python\Theory\2nd>C:/Users/khamr/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/khamr/Desktop/Python/Theory/2nd/16.py  
Enter the year: 2029  
The starting day of the year 2029 is: Monday  
  
C:\Users\khamr\Desktop\Python\Theory\2nd>
```