|  |  |
| --- | --- |
| **Task 3** | Importing Python modules and packages in python programming |

1. Given a string and a number k, find the k-th non-repeating character in the string. Consider a input string with set of characters. How to find the character by only doing only one traversal of input string?

**INPUT**

First line of input contains a string and second line contains the value k.

**OUTPUT**

For each input string, Print the kth non repeating character

**CONSTRAINTS**

1 ≤ K ≤ 109

**Sample Input and Output:**

Input : str = pythonprogramming, k = 3

Output : h

First non-repeating character is y,

second is t and third is h.

Input : str = pythonprogramming, k = 4

Output : a

Input : str = pythonprogramming, k = 6

Output : Less than k non-repeating

characters in input.

**Program:**

# Function to find k'th non repeating character

# in string

from collections import OrderedDict

def kthRepeating(input,k):

# OrderedDict returns a dictionary data structure having characters of input

# string as keys in the same order they were inserted and 0 as their default value

dict=OrderedDict.fromkeys(input,0)

# now traverse input string to calculate frequency of each character

for ch in input:

dict[ch]+=1

# now extract list of all keys whose value is 1 from dict Ordered Dictionary

nonRepeatDict = [key for (key,value) in dict.items() if value==1]

# now return (k-1)th character from above list

if len(nonRepeatDict) < k:

return 'Less than k non-repeating characters in input.'

else:

return nonRepeatDict[k-1]

# Driver function

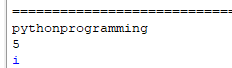
if \_\_name\_\_ == "\_\_main\_\_":

inp = str(input())

k = int(input())

print (kthRepeating(inp, k))

**output:**



1. It's a holiday season for all school students around the world! Unfortunately, Mahamba is busy preparing for International Olympiad in Informatics, which will be held in Tehran, Iran. He wants to wrie a program to Find the Sum of Cosine Series

**Sample Input and Output:**

First Line : Value of x in degrees

Second Line :Number of terms

Output:

Print the Sum of Cosine Series

TEST CASE 1

INPUT

65

10

OUTPUT

0.42

**Program:**

import math

def cosine(x,n):

cosx = 1

sign = -1

for i in range(2, n, 2):

pi=22/7

y=x\*(pi/180)

cosx = cosx + (sign\*(y\*\*i))/math.factorial(i)

sign = -sign

return cosx

x=int(input())

n=int(input())

print(round(cosine(x,n),2))

**Output:**



1. Create a modular Python application to analyze student performance data. The program must be **split into separate modules** to follow good programming practices:

* **Module 1** will handle **data loading** from a CSV file.
* **Module 2** will handle **statistical analysis** of the data.
* The **main program** will import these modules, perform calculations, and display the results in a formatted table.

Sample output:

Subject Mean Standard Deviation Mode

0 Math 86.000000 7.071068 85

1 Science 82.600000 10.200000 78

2 English 85.400000 8.089483 92

**students.csv**

Name,Age,Math,Science,English

Alice,20,85,78,92

Bob,21,90,88,84

Charlie,20,72,65,70

David,22,95,92,96

Eva,21,88,90,85

**dataloader.py**

import pandas as pd

def load\_data():

df = pd.read\_csv("students.csv")

return df

**statshelper.py**

import numpy as np

from scipy import stats

def subject\_mean(df, subject):

return np.mean(df[subject])

def subject\_std(df, subject):

return np.std(df[subject])

def subject\_mode(df, subject):

return stats.mode(df[subject], keepdims=True).mode[0]

**main.py**

# Step 1: Imports

from dataloader import load\_data

from statshelper import subject\_mean, subject\_std, subject\_mode

import pandas as pd

# Step 2: Load CSV data

df = load\_data()

# Step 3: Subjects to analyze

subjects = ["Math", "Science", "English"]

# Step 4: Collect results

results = []

for subj in subjects:

mean\_val = subject\_mean(df, subj)

std\_val = subject\_std(df, subj)

mode\_val = subject\_mode(df, subj)

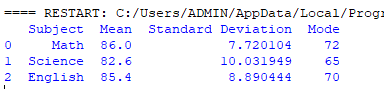
results.append([subj, mean\_val, std\_val, mode\_val])

# Step 5: Create results table

results\_df = pd.DataFrame(results, columns=["Subject", "Mean", "Standard Deviation", "Mode"])

print(results\_df)

output:



1. Create a program to generate sample sales data for different products, calculate total sales for each product, identify the top-selling product, and display the total sales amount along with the data.

**Sample output:**

Top Selling Product: Laptop

Total Sales: ₹ 1215587

**Sample Data:**

Product Units \_Sold Price\_per\_Unit Total\_Sales

0 Laptop 21 4633 97293

1 Headphones 32 4876 156032

2 Mouse 26 2711 70486

3 Keyboard 14 2789 39046

4 Laptop 12 4640 55680

import pandas as pd

import numpy as np

# Create sample sales data

data = pd.DataFrame({

'Product': ['Laptop', 'Headphones', 'Mouse', 'Keyboard'] \* 5,

'Units\_Sold': np.random.randint(5, 50, 20),

'Price\_per\_Unit': np.random.randint(1000, 5000, 20)

})

# Calculate total sales for each row

data['Total\_Sales'] = data['Units\_Sold'] \* data['Price\_per\_Unit']

# Find top-selling product

top\_product = data.groupby('Product')['Total\_Sales'].sum().idxmax()

# Calculate total sales

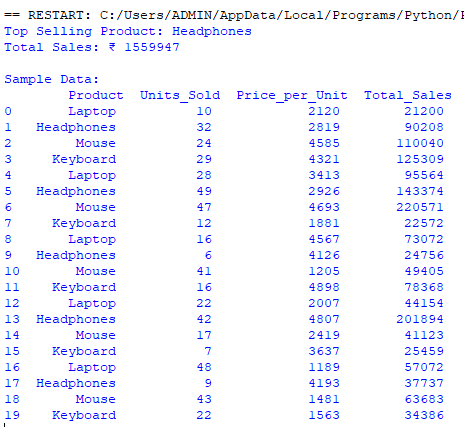
total\_sales = data['Total\_Sales'].sum()

# Show results

print("Top Selling Product:", top\_product)

print("Total Sales: ₹", total\_sales)

print("\nSample Data:\n", data)



1. To write a Python program using **packages** with modules for geometry (area, perimeter, hypotenuse) and text processing (uppercase, word count, palindrome) by organizing code into reusable components.

**Algorithm**

**Step 1 – Create the Package Structure**

1. Create a folder named my\_toolkit
2. Inside my\_toolkit, create:

* \_\_init\_\_.py
* geometry\_utils.py
* text\_utils.py

**Step 2 – Implement geometry\_utils.py Module**

1. area\_circle(radius) → Returns π × radius².
2. perimeter\_rectangle(length, width) → Returns 2 × (length + width).
3. pythagoras(a, b) → Returns √(a² + b²).

**Step 3 – Implement text\_utils.py Module**

1. to\_uppercase(s) → Returns the string in uppercase.
2. word\_count(s) → Counts the number of words.
3. is\_palindrome(s) → Returns True if the string reads the same forward and backward.

**Step 4 – Initialize Package with \_\_init\_\_.py**

1. Import functions from geometry\_utils and text\_utils so they are directly accessible from my\_toolkit.

**Step 5 – Create main.py Program**

1. Import all required functions from my\_toolkit.
2. Call geometry functions and display their results.
3. Call text functions and display their results.

**Step 6 – Execute the Program**

1. Run python main.py in the terminal.
2. Verify that outputs match expected results.

**Program**

1. Folder structure:

my\_toolkit/

\_\_init\_\_.py

geometry\_utils.py

text\_utils.py

main.py

**my\_toolkit/geometry\_utils.py**

import math

def area\_circle(radius):

return math.pi \* radius \*\* 2

def perimeter\_rectangle(length, width):

return 2 \* (length + width)

def pythagoras(a, b):

return math.sqrt(a\*\*2 + b\*\*2)

**my\_toolkit/text\_utils.py**

def to\_uppercase(s):

return s.upper()

def word\_count(s):

return len(s.split())

def is\_palindrome(s):

s\_clean = ''.join(s.lower().split())

return s\_clean == s\_clean[::-1]

**my\_toolkit/\_\_init\_\_.py**

from .geometry\_utils import area\_circle, perimeter\_rectangle, pythagoras

from .text\_utils import to\_uppercase, word\_count, is\_palindrome

**main.py**

from my\_toolkit import area\_circle, perimeter\_rectangle, pythagoras

from my\_toolkit import to\_uppercase, word\_count, is\_palindrome

# Geometry functions

print("Area of circle (r=5):", round(area\_circle(5), 2))

print("Perimeter of rectangle (4x6):", perimeter\_rectangle(4, 6))

print("Hypotenuse (3,4):", round(pythagoras(3, 4), 2))

# Text functions

print("Uppercase:", to\_uppercase("hello world"))

print("Word count:", word\_count("Python is fun"))

print("Is palindrome (level):", is\_palindrome("level"))

## **Sample Output**

Area of circle (r=5): 78.54

Perimeter of rectangle (4x6): 20

Hypotenuse (3,4): 5.0

Uppercase: HELLO WORLD

Word count: 3

Is palindrome (level): True