Program-1

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import re

def madlibs():

text = '''Today I went to the zoo. I saw a(n) ADJECTIVE NOUN jumping up and down in its tree.

He VERB ADVERB through the large tunnel that led to its NOUN.

I got some peanuts and passed them through the cage to a gigantic ADJECTIVE NOUN towering above my head.

Feeding that animal made me hungry. I went to get a ADJECTIVE scoop of ice cream.

It VERB ADVERB down my arm and onto my NOUN.'''

print("Original Text:\n", text)

placeholders = re.findall(r'\b(ADJECTIVE|NOUN|VERB|ADVERB)\b', text)

for placeholder in placeholders:

word = input(f"Enter a {placeholder.lower()}: ")

text = text.replace(placeholder, word, 1)

print("\nModified Text:\n", text)

with open('madlibs\_output.txt', 'w') as file:

file.write(text)

print("\nModified text saved to 'madlibs\_output.txt'")

madlibs()

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Original Text:

Today I went to the zoo. I saw a(n) ADJECTIVE NOUN jumping up and down in its tree.

He VERB ADVERB through the large tunnel that led to its NOUN.

I got some peanuts and passed them through the cage to a gigantic ADJECTIVE NOUN towering above my head.

Feeding that animal made me hungry. I went to get a ADJECTIVE scoop of ice cream.

It VERB ADVERB down my arm and onto my NOUN.

Enter a adjective: AAA

Enter a noun: BBB

Enter a verb: CCC

Enter a adverb: DDD

Enter a noun: EEE

Enter a adjective: FFF

Enter a noun: GGG

Enter a adjective: HHH

Enter a verb: III

Enter a adverb: JJJ

Enter a noun: KKK

Modified Text:

Today I went to the zoo. I saw a(n) AAA BBB jumping up and down in its tree.

He CCC DDD through the large tunnel that led to its EEE.

I got some peanuts and passed them through the cage to a gigantic FFF GGG towering above my head.

Feeding that animal made me hungry. I went to get a HHH scoop of ice cream.

It III JJJ down my arm and onto my KKK.

Modified text saved to 'madlibs\_output.txt'

Program-2

print("NAME:CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import re

import os

def search(folder,element):

try:

for file in os.listdir(folder):

if file.endswith(".txt"):

file=os.path.join(folder,file)

try:

with open(file,"r") as f:

for linenum, line in enumerate(f,1):

print(f"File: {file},linne {linenum}: {line.strip()}")

except Exception as e:

print(f"Error reading {file}:{e}")

except FileNotFoundError:

print("Folder not found")

except Exception as e:

print(f"an unexpected error {e}")

folder=input("Enter any folder name:")

element=input("enter any expression to search:")

search(folder,element)

Output:

NAME:CHETAN U

USN:1AY24AI025

SECTION:M

Enter any folder name: python1

enter any expression to search: example.txt

File: python1\example.txt,linne 1: 2+5=7

File: python1\example.txt,linne 2: 3\*8=24

Program-3

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import os

import shutil

def selective(source, extensions, destination):

source = os.path.abspath(source)

destination = os.path.abspath(destination)

print(f"Looking in '{source}' for files with extensions: {', '.join(extensions)}")

for foldername, subfolders, filenames in os.walk(source):

for filename in filenames:

\_, ext = os.path.splitext(filename)

if ext.lower() in extensions:

source\_file = os.path.join(foldername, filename)

print(f"Copying '{source\_file}' to '{destination}'")

shutil.copy(source\_file, destination)

extensions = ['.php', '.py']

source = 'randomFolder'

destination = 'selectiveFolder'

selective(source, extensions, destination)

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Looking in 'D:\Py programs\python' for files with extensions: .php, .py

Copying 'D:\Py programs\python\characterpic.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\chess.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\coinflip.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\collatz.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\commacode.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\conway.py' to 'D:\Py programs\selectiveFolder'

Copying 'D:\Py programs\python\fantasy.py' to 'D:\Py programs\selectiveFolder'

Program-4

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import os

import pathlib

def find(folder, size\_limit=100 \* 1024 \* 1024):

for dirpath, dirnames, filenames in os.walk(folder):

for filename in filenames:

file\_path = os.path.join(dirpath, filename)

try:

file\_size = os.path.getsize(file\_path)

if file\_size > size\_limit:

print(f"{file\_path} - {file\_size / (1024 \* 1024):.2f} MB")

except OSError as e:

print(f"Error accessing {file\_path}: {e}")

find(pathlib.Path.home())

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

C:\Users\Aneesh\AppData\Local\canva-updater\installer.exe - 167.37 MB

C:\Users\Aneesh\AppData\Local\Google\Chrome\User Data\component\_crx\_cache\35b755f253cbcca00af6ef7d8ed89aa87531d9b42f02d913c806c68b7440fbf9 - 106.15 MB

C:\Users\Aneesh\AppData\Local\Packages\5319275A.WhatsAppDesktop\_cv1g1gvanyjgm\LocalState\messages.db - 314.63 MB

C:\Users\Aneesh\AppData\Local\Packages\5319275A.WhatsAppDesktop\_cv1g1gvanyjgm\LocalState\shared\transfers\2025\_19\VID-20250518-WA0031.mp4 - 119.75 MB

C:\Users\Aneesh\AppData\Local\Packages\5319275A.WhatsAppDesktop\_cv1g1gvanyjgm\LocalState\shared\transfers\2025\_19\VID-20250518-WA0034.mp4 - 164.06 MB

Error accessing C:\Users\Aneesh\CrossDevice\Aneesh\storage\Android\media\com.whatsapp\WhatsApp\Media\WhatsApp Documents\Sent\DOC-20240625-WA0063.: [WinError 2] The system cannot find the file specified: 'C:\\Users\\Aneesh\\CrossDevice\\Aneesh\\storage\\Android\\media\\com.whatsapp\\WhatsApp\\Media\\WhatsApp Documents\\Sent\\DOC-20240625-WA0063.'

Program-5

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import os

import re

def fillgap(folder, prefix='spam', extension='.txt'):

# Regex to match files like spam001.txt

pattern = re.compile(rf'^{prefix}(\d+){re.escape(extension)}$')

files = []

for f in os.listdir(folder):

match = pattern.match(f)

if match:

number = int(match.group(1))

files.append((number, f))

if not files:

print("No matching files found.")

return

# Sort files based on number

files.sort()

# Get number width (e.g., 3 for spam001)

number\_width = len(re.search(r'\d+', files[0][1]).group())

expected\_num = 1

for actual\_num, filename in files:

if actual\_num != expected\_num:

# Rename file to close the gap

new\_name = f"{prefix}{str(expected\_num).zfill(number\_width)}{extension}"

old\_path = os.path.join(folder, filename)

new\_path = os.path.join(folder, new\_name)

os.rename(old\_path, new\_path)

print(f"Renamed {filename} → {new\_name}")

expected\_num += 1

print("Gaps filled successfully.")

# Example usage

fillgap(os.getcwd(), 'spam', '.txt')

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Renamed spam003.txt → spam002.txt

Gaps filled successfully.

Program-6

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import random

guess = ''

while guess not in ('heads'.'tails'):

print("Guess the coin toss! Enter heads ot tails: ")

guess = input()

break

toss = random.randint(0,1)

if toss==guess:

print("You got it")

else:

print("NOPE!!! Guess again! ")

guess = input()

if toss==guess:

print("You got it")

else:

print("Nope, you did not get this")

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Guess the coin toss! Enter heads ot tails:

heads

NOPE!!! Guess again!

tails

Nope, you did not get this

Program-7

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

import math

class Point:

def \_\_init\_\_(self, x, y): self.x, self.y = x, y

class Circle:

def \_\_init\_\_(self, center, radius): self.center, self.radius = center, radius

class Rectangle:

def \_\_init\_\_(self, corner, width, height): self.corner, self.width, self.height = corner, width, height

def point\_in\_circle(circle, point):

dx = point.x - circle.center.x

dy = point.y - circle.center.y

return math.hypot(dx, dy) <= circle.radius

def rect\_in\_circle(circle, rect):

return all(point\_in\_circle(circle, Point(rect.corner.x + dx, rect.corner.y + dy))

for dx in (0, rect.width) for dy in (0, rect.height))

def rect\_circle\_overlap(circle, rect):

return any(point\_in\_circle(circle, Point(rect.corner.x + dx, rect.corner.y + dy))

for dx in (0, rect.width) for dy in (0, rect.height))

c = Circle(Point(150, 100), 75)

r = Rectangle(Point(100, 50), 100, 50)

print(point\_in\_circle(c, Point(150, 100)))

print(point\_in\_circle(c, Point(230, 100)))

print(rect\_in\_circle(c, r))

print(rect\_circle\_overlap(c, r))

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

True

False

True

True

Program-8

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

class Time:

def \_\_init\_\_(self, hour=0, minute=0, second=0):

self.hour = hour

self.minute = minute

self.second = second

def time\_to\_seconds(t):

return t.hour \* 3600 + t.minute \* 60 + t.second

def seconds\_to\_time(seconds):

hour, rem = divmod(int(seconds), 3600)

minute, second = divmod(rem, 60)

return Time(hour, minute, second)

def multiply\_time(t, factor):

return seconds\_to\_time(time\_to\_seconds(t) \* factor)

def average\_pace(finish\_time, distance):

return multiply\_time(finish\_time, 1 / distance)

finish = Time(1, 23, 45)

distance = 3.1

pace = average\_pace(finish, distance)

print(f"Average pace per mile: {pace.hour:02d}:{pace.minute:02d}:{pace.second:02d}")

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Average pace per mile: 00:27:00

Program-9

print("NAME:CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

from datetime import datetime, timedelta

def show\_day\_of\_week():

today = datetime.today()

print("Today is:", today.strftime("%A"))

def birthday\_info():

bday = input("Enter your birthday (YYYY-MM-DD): ")

bday = datetime.strptime(bday, "%Y-%m-%d")

now = datetime.now()

age = now.year - bday.year - ((now.month, now.day) < (bday.month, bday.day))

next\_bday = bday.replace(year=now.year)

if next\_bday < now:

next\_bday = next\_bday.replace(year=now.year + 1)

delta = next\_bday - now

seconds = int(delta.total\_seconds())

print(f"You are {age} years old.")

print(f"Time until next birthday: {delta.days} days, {seconds//3600%24} hours, {seconds//60%60} mins, {seconds%60} secs.")

def double\_day():

b1 = datetime.strptime(input("First birthday (YYYY-MM-DD): "), "%Y-%m-%d")

b2 = datetime.strptime(input("Second birthday (YYYY-MM-DD): "), "%Y-%m-%d")

older, younger = sorted([b1, b2])

double = older + 2 \* (younger - older)

print("Double Day:", double.date())

def n\_times\_day():

b1 = datetime.strptime(input("First birthday (YYYY-MM-DD): "), "%Y-%m-%d")

b2 = datetime.strptime(input("Second birthday (YYYY-MM-DD): "), "%Y-%m-%d")

n = float(input("Enter n (e.g. 2 for double): "))

older, younger = sorted([b1, b2])

target = older + timedelta(seconds=(younger - older).total\_seconds() / (n - 1))

print(f"{n} times older day:", target.date())

def main():

while True:

print("\n1. Day of week\n2. Birthday info\n3. Double Day\n4. N-times Day\n5. Exit")

choice = input("Choose (1-5): ")

if choice == '1': show\_day\_of\_week()

elif choice == '2': birthday\_info()

elif choice == '3': double\_day()

elif choice == '4': n\_times\_day()

elif choice == '5': break

else: print("Try again.")

main()

Output:

NAME:CHETAN U

USN:1AY24AI025

SECTION:M

1. Day of week

2. Birthday info

3. Double Day

4. N-times Day

5. Exit

Choose (1-5): 1

Today is: Tuesday

1. Day of week

2. Birthday info

3. Double Day

4. N-times Day

5. Exit

Choose (1-5): 5

Program-10

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

class Time:

def \_\_init\_\_(self, hour=0, minute=0, second=0):

self.seconds = hour \* 3600 + minute \* 60 + second

def \_\_str\_\_(self):

minutes, seconds = divmod(self.seconds, 60)

hours, minutes = divmod(minutes, 60)

return f'{hours:02d}:{minutes:02d}:{seconds:02d}'

def increment(self, seconds):

self.seconds += seconds

return self

def is\_\_after(self, other):

return self.seconds > other.seconds

def \_\_add\_\_(self, other):

if isinstance(other, Time):

return Time(0, 0, self.seconds + other.seconds)

else:

return Time(0, 0, self.seconds + other)

def int\_\_to\_\_time(seconds):

return Time(0, 0, seconds)

def main():

start\_\_time = Time(9, 45, 0)

print(f"Start time: {start\_\_time}")

end\_\_time = Time(start\_\_time.seconds)

end\_\_time.increment(1337)

print(f"End time: {end\_\_time}")

print(f"Is end time after start time? {end\_\_time.is\_\_after(start\_\_time)}")

print(f"Start time + 1000 seconds: {start\_\_time + 1000}")

print(f"Start time + 1 hour: {start\_\_time + 3600}")

main()

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Start time: 09:45:00

End time: 35100:22:17

Is end time after start time? True

Start time + 1000 seconds: 10:01:40

Start time + 1 hour: 10:45:00

Program-11

print("NAME: CHETAN U")

print("USN:1AY24AI025")

print("SECTION:M")

class Kangaroo:

def \_\_init\_\_(self, pouch\_contents=None):

if pouch\_contents is None:

self.pouch\_contents = []

else:

self.pouch\_contents = list(pouch\_contents)

def put\_in\_pouch(self, item):

self.pouch\_contents.append(item)

def \_\_str\_\_(self):

contents = ', '.join(str(item) for item in self.pouch\_contents)

return f"Kangaroo with pouch contents: [{contents}]"

kanga = Kangaroo()

roo = Kangaroo()

kanga.put\_in\_pouch("wallet")

kanga.put\_in\_pouch("keys")

kanga.put\_in\_pouch(roo)

print(kanga)

Output:

NAME: CHETAN U

USN:1AY24AI025

SECTION:M

Kangaroo with pouch contents: [wallet, keys, Kangaroo with pouch contents: []]