Experiment 1

Experiment: Write a program to Rule-based automation (email filtering).

Aim : Write a program for filtering emails based on keyword in the subject/sender.

Program :

```
emails = [
    {"sender": "boss@example.com", "subject": "Meeting remiander."},
    {"sender": "spam@abcd.com", "subject": "Buy Now"},
    {"sender": "friend@example.com", "subject": "Weekend Plans."},
]
def filter_email(email, keyword):
    filtered = [
        email for email in emails if keyword.lower() in email["subject"].lower()
    ]
    return filtered


keyword = "Buy"
print(filter_email(emails, keyword))
```

Input :

keyword = "Buy"

Output :

```
print(filter_email(emails, keyword))
```

```
[{'sender': 'spam@abcd.com', 'subject': 'Buy Now'}]
```

Result : Thus, the program has been successfully filtered the emails based on keyword and verified.

Experiment 2

Experiment : Write a program to Chatbot for FAQs (keyword matching).

Aim : Write a program to create a chatbot for FAQs (keyword).

Program :

```
faqs = {
    "hours": "We are open from 9AM to 6PM",
    "location": "Our office is located at 123 Main Street.",
    "contact": "You can reach us at contact@example.com",
}
def chatbot(query):
    for key in faqs:
        if key in query.lower():
            return faqs[key]
    return "Sorry, I don't understand the question"
query = "What are your hours ?"
res = chatbot(query)
print(res)
```

Input :

What are your hours ?

Output :

```
query = "What are your hours ?"
res = chatbot(query)
print(res)
```

```
We are open from 9AM to 6PM
```

Result : Thus, the program has been successfully created a chatbot for FAQs and verified.

Experiment 3

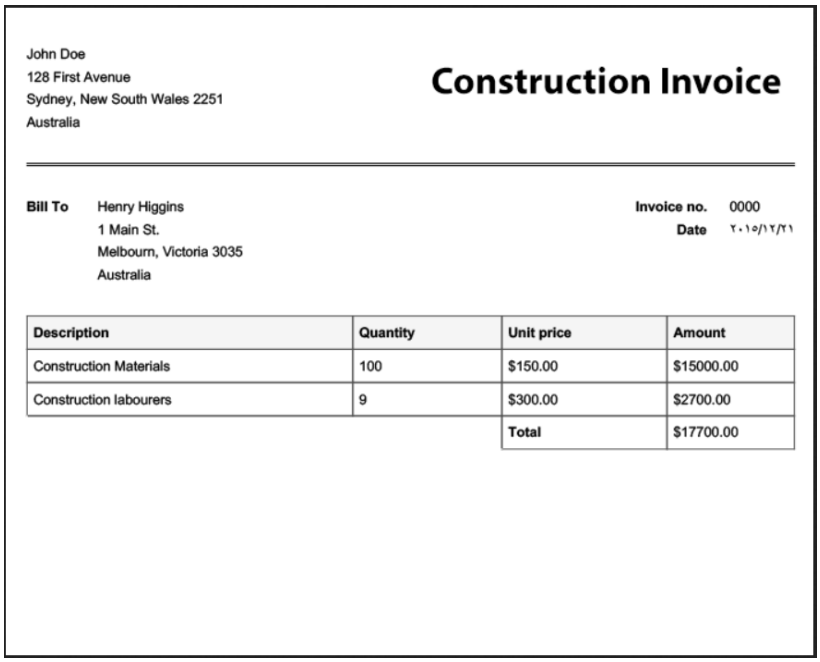
Experiment : Write a program to Automated data entry with OCR(tesseract example).

Aim : Write a program to automated data entry with OCR(Tesseract).

Program :

```
from PIL import Image

import pytesseract as pt

def ocr_text(filename):

    img = Image.open(filename)

    text = pt.image_to_string(img)

    return img, text

filename = "iv2.png"

img, res_text = ocr_text(filename)

print(res_text)
```

Input:



John Doe
128 First Avenue
Sydney, New South Wales 2251
Australia

Construction Invoice

Bill To Henry Higgins
1 Main St.
Melbourn, Victoria 3035
Australia

Invoice no. 0000
Date ٢٠١٥/١٢/٢١

Description	Quantity	Unit price	Amount
Construction Materials	100	$150.00	$15000.00
Construction labourers	9	$300.00	$2700.00
		Total	$17700.00

Output :

In [6]:
```
filename = "iv2.png"
img, res_text = ocr_text(filename)
print(res_text)
```

Out[6]:
```
John Doe
128 First Avenue

‘Sydney, New South Wales 2251
Australia

Construction Invoice

Bill To Henry Higgins

Invoice no. 0000

1 Main St. Date rsvenxyry
Melbourn, Victoria 3035
Australia
Description Quantity Unit price ‘Amount
Construction Materials 100 $150.00 $1500.00
Construction labourers: 9 $300.00 $2700.00
Total $1700.00
```

Result : Thus, the program has been successfully automated data entry with OCR and verified.

Experiment 4

Experiment : Write a program to Email automation with NLP (simple keyword categorization).

Aim : Write a program to perform email automation with NLP.

Program :

```
emails = [
    {"subject": "URGENT: Project deadline", "body": "Please finish ASAP"},
    {"subject": "Hello friend", "body": "How are you?"},
    {"subject": "Win a free prize!", "body": "Click here!"}
]
def categorize_email(email):
    subject = email["subject"].lower()
    if "urgent" in subject or "asap" in subject:
        return "important"
    elif "win" in subject or "prize" in subject:
        return "spam"
    else:
        return "normal"
email = {"subject": "Win a free prize!", "body": "Click here!"}
print(categorize_email(email))
```

Input :

{"subject": "Win a free prize!", "body": "Click here!"}

Output :

```
email = {"subject": "Win a free prize!", "body": "Click here!"}
print(categorize_email(email))
```

```
spam
```

Result : Thus, the program has been successfully email automation with NLP and verified.

Experiment 5

Experiment : Write a program to Web scraping and data extraction bot (beautiful soup).

Aim : Write a program to perform web scraping and data extraction.

Program :

```
import requests

from bs4 import BeautifulSoup as BS

def extract_heading(url):

    res = requests.get(url)

    soup = BS(res.text, "html.parser")

    headings = [h.get_text() for h in soup.find_all(["h1", "h2", "h3"])]

    return headings


url = " https://www.geeksforgeeks.org/"

headings = extract_heading(url)

print(headings)
```

Input :

https://www.geeksforgeeks.org/

Output :

```
url = " https://www.geeksforgeeks.org/"
headings = extract_heading(url)
print(headings)

['Explore', 'Courses', 'DSA', 'AI ML & Data Science', 'Web Development', 'Languages', 'CS Subjects ', 'Databases', 'DevOps', 'Tutorials', 'Free Courses', 'GfG School ', 'Must Explore']
```

Result : Thus, the program has been successfully performed scraping , data extraction and verified.

Experiment 6

Experiment: Invoice Processing Automation(Simple parsing).

Aim : Write a program to extract invoice number and amount from a text.

Program :

```
invoice_text = """Invoice No: 12345

                  Date: 2025-05-01

                  Total amount: $ 2500

                  Thank you for your business """

def extract_invoice_details(text):
    lines = text.split("\n")
    details = {}
    for line in lines:
        if "Invoice No" in line:
            details['invoice_no'] = line.split(":")[1].strip()
        elif "Total amount" in line:
            details['total_amount'] = line.split(":")[1].strip()
    return details

invoice_details = extract_invoice_details(invoice_text)

print(invoice_details)
```

Input :

Invoice No: 12345 Date: 2025-05-01 Total amount: $ 2500 Thank you for your business

Output :

```
{'invoice_no': '12345', 'total_amount': '$ 2500'}
```

Result : Thus, the program has been successfully extracted invoice no. and amount from a text and verified.

Experiment 7

Experiment: Simple image recognition for sorting.

Aim : Write a program to recognize image using pretrained model placeholder.

Program :

```
from PIL import Image

def classify_image(image_path):

    category = "cat"

    return category

image_path = 'cat-img.jpg'

category = classify_image(image_path)

print(category)
```

Input :

Output :

```
cat
```

Result : Thus, the program has been successfully recognized the image and verified.

Experiment 8

Experiment: Task reminder automation.

Aim : Write a program that automate the task reminder.

Program :

```
import datetime

tasks = [{"task": "Submit report", "due_date": "2025-09-05"},{"task": "Pay bills",
"due_date": "2025-09-04"},{"task": "File taxes", "due_date": "2025-09-20"}]

def check_reminders(tasks):

    today = datetime.datetime.now().date()

    reminders = []

    for task in tasks:

        due_date_str = task["due_date"]

        due_date = datetime.datetime.strptime(due_date_str, "%Y-%m-%d").date()

        time_diff = due_date - today

        if datetime.timedelta(days=0) <= time_diff <= datetime.timedelta(days=3):

            reminders.append(f"Reminder on: {task['task']} is due on {due_date_str}")

    return reminders

reminders_list = check_reminders(tasks)

print(reminders_list)
```

Input :

```
[{"task": "Submit report", "due_date": "2025-09-05"},{"task": "Pay bills", "due_date":
"2025-09-04"},{"task": "File taxes", "due_date": "2025-09-20"}]
```

Output :

```
['Reminder on: Pay bills is due on 2025-09-04']
```

Result : Thus, the program has been successfully automates the task reminder and verified.

Experiment 9

Experiment: Smart attendance system (face recognition using OpenCV).

Aim : Write a program to detect and mark attendance using recognition.

Program :

```
def marke_attendance(img_path):
    attendance_marked = ['Student1', 'Student2']
    return attendance_marked
img = 'class_photo.jpg'
attendance  =  marke_attendance(img)
print(attendance)
```

Input :

Output :

```
['Student1', 'Student2']
```

Result : Thus, the program has been successfully detect and mark attendance and verified.

Experiment 10

Experiment: Social media sentiment analysis.

Aim : Write a program that analyze sentiment of a tweet.

Program :

```
from textblob import TextBlob

def analyze_sentiment(text):
    analysis = TextBlob(text)
    if analysis.sentiment.polarity > 0:
        return 'Positive'
    elif analysis.sentiment.polarity == 0:
        return 'Neutral'
    else:
        return 'Negative'
text = "I do not like this pen"
sentiment = analyze_sentiment(text)
print(sentiment)
```

Input :

I do not like this pen

Output :

```
Neutral
```

Result : Thus, the program has been successfully analyze sentiment of a tweet and verified.