GA 1

1) Install and run Visual Studio Code. In your Terminal (or Command Prompt), type code -s and press Enter. Copy and paste the entire output below.

What is the output of code -s?

2) Running uv run --with httpie -- https [URL] installs the Python package httpie and sends a HTTPS request to the URL.

Send a HTTPS request to https://httpbin.org/get with the URL encoded parameter email set to **23f2003751@ds.study.iitm.ac.in**

What is the JSON output of the command? (Paste only the JSON body, not the headers)

3) Download README.md. In the directory where you downloaded it, make sure it is called README.md, and run npx -y prettier@3.4.2 README.md | sha256sum.

What is the output of the command?

4)Let's make sure you can write formulas in Google Sheets. Type this formula into Google Sheets. (It won't work in Excel)

=SUM(ARRAY_CONSTRAIN(SEQUENCE(**100, 100, 5, 2**), **1, 10**))

What is the result?

5) Note: This will ONLY work in Office 365.

=SUM(TAKE(SORTBY({**1,10,12,4,6,8,9,13,6,15,14,15,2,13,0,3},**
{10,9,13,2,11,8,16,14,7,15,5,4,6,1,3,12}), **1, 6**))

What is the result?

Note: If you get #NAME? you have the wrong version of Excel. Find a friend for whom this works.

6) Just above this paragraph, there's a hidden input with a secret value.
What is the value in the hidden input?

7) How many Wednesdays are there in the date range **1986-08-06** to **2008-01-29**?

The dates are in the year-month-day format. Include both the start and end date in your count. You can do this using any tool (e.g. Excel, Python, JavaScript, manually).

8) Download and unzip file q-extract-csv-zip.zip which has a single extract.csv file inside.

What is the value in the "answer" column of the CSV file?

9) Let's make sure you know how to use JSON. Sort this JSON array of objects by the value of the age field. In case of a tie, sort by the name field. Paste the resulting JSON below without any spaces or newlines.

[{"name":"Alice","age":92},{"name":"Bob","age":28},{"name":"Charlie","age":16},{"name":"David","age":56},{"name":"Emma","age":70},{"name":"Frank","age":67},{"name":"Grace","age":36},{"name":"Henry","age":94},{"name":"Ivy","age":44},{"name":"Jack","age":53},{"name":"Karen","age":65},{"name":"Liam","age":23},{"name":"Mary","age":97},{"name":"Nora","age":68},{"name":"Oscar","age":57},{"name":"Paul","age":88}]

Sorted JSON:

10) Download q-multi-cursor-json.txt and use multi-cursors and convert it into a single JSON object, where key=value pairs are converted into {key: value, key: value, ...}.

What's the result when you paste the JSON at tools-in-data-science.pages.dev/jsonhash and click the Hash button?

11) Let's make sure you know how to select elements using CSS selectors. Find all <div>s having a foo class in the hidden element below. What's the sum of their data-value attributes?

Sum of data-value attributes:

12) Download and process the files in q-unicode-data.zip which contains three files with different encodings:

data1.csv: CSV file encoded in CP-1252

data2.csv: CSV file encoded in UTF-8

data3.txt: Tab-separated file encoded in UTF-16

Each file has 2 columns: symbol and value. Sum up all the values where the symbol matches **” OR † OR Ž** across all three files.

What is the sum of all values associated with these symbols?

13) Create a GitHub account if you don't have one. Create a new public repository. Commit a single JSON file called email.json with the value {"email": "**23f2003751@ds.study.iitm.ac.in**"} and push it.

Enter the raw Github URL ofemail.jsonso we can verify it. (It might look likehttps://raw.githubusercontent.com/[GITHUB ID]/[REPO NAME]/main/email.json.)

14) Download q-replace-across-files.zip and unzip it into a new folder, then replace all "IITM" (in upper, lower, or mixed case) with "IIT Madras" in all files. Leave everything as-is - don't change the line endings.

What does runningcat * | sha256sumin that folder show in bash?

15) Download q-list-files-attributes.zip and extract it. Use ls with options to list all files in the folder along with their date and file size.

What's the total size of all files at least **4772** bytes large and modified on or after **Thu, 1 Dec, 2011, 10:53 pm** IST?

Don't copy from inside the ZIP file or use Windows Explorer to unzip. That destroys the timestamps. Extract using unzip, 7-Zip or similar utilities and check the timestamps.

16) Download q-move-rename-files.zip and extract it. Use mv to move all files under folders into an empty folder. Then rename all files replacing each digit with the next. 1 becomes 2, 9 becomes 0, a1b9c.txt becomes a2b0c.txt.

What does runninggrep . * | LC_ALL=C sort | sha256 sum in bash on that folder show?

17) Download q-compare-files.zip and extract it. It has 2 nearly identical files, a.txt and b.txt, with the same number of lines.

How many lines are different betweena.txtandb.txt?

18) There is a tickets table in a SQLite database that has columns type, units, and price. Each row is a customer bid for a concert ticket.

typeunitspriceGOLD2451.26bronze2021.36Gold1800.8Bronze6541.69SILVER9870.86...

What is the total sales of all the items in the "Gold" ticket type? Write SQL to calculate it.

Get all rows where the Type is "Gold". Ignore spaces and treat mis-spellings like GOLD, gold, etc. as "Gold". Calculate the sales as Units * Price, and sum them up.

GA 2

1) Write documentation in Markdown for an *imaginary* analysis of the number of steps you walked each day for a week, comparing over time and with friends. The Markdown must include:

Top-Level Heading: At least 1 heading at level 1, e.g., # Introduction

Subheadings: At least 1 heading at level 2, e.g., ## Methodology

Bold Text: At least 1 instance of bold text, e.g., *important*

Italic Text: At least 1 instance of italic text, e.g., note

Inline Code: At least 1 instance of inline code, e.g., sample_code

Code Block: At least 1 instance of a fenced code block, e.g.

print("Hello World")

Bulleted List: At least 1 instance of a bulleted list, e.g., - Item

Numbered List: At least 1 instance of a numbered list, e.g., 1. Step One

Table: At least 1 instance of a table, e.g., | Column A | Column B |

Hyperlink: At least 1 instance of a hyperlink, e.g., [Text](https://example.com)

Image: At least 1 instance of an image, e.g., ![Alt Text](https://example.com/image.jpg)

Blockquote: At least 1 instance of a blockquote, e.g., > This is a quote

2) By losslessly, we mean that every pixel in the new image should be identical to the original image.

Upload your losslessly compressed image (less than 1,500 bytes)

3) Publish a page using GitHub Pages that showcases your work. Ensure that your email address **23f2003751@ds.study.iitm.ac.in** is in the page's HTML.

GitHub pages are served via CloudFlare which obfuscates emails. So, wrap your email address inside a:

<!--email_off-->**23f2003751@ds.study.iitm.ac.in**<!--/email_off-->

What is the GitHub Pages URL? It might look like:https://[USER].github.io/[REPO]/

If a recent change that's not reflected, add ?v=1, ?v=2 to the URL to bust the cache.

4) Let's make sure you can access Google Colab. Run this program on Google Colab, allowing all required access to your email ID: **23f2003751@ds.study.iitm.ac.in**.

```
import hashlib
import requests
from google.colab import auth
from oauth2client.client import GoogleCredentials

auth.authenticate_user()
creds = GoogleCredentials.get_application_default()
token = creds.get_access_token().access_token
response = requests.get(
  "https://www.googleapis.com/oauth2/v1/userinfo",
  params={"alt": "json"},
  headers={"Authorization": f"Bearer {token}"}
)
email = response.json()["email"]
hashlib.sha256(f"{email} {creds.token_expiry.year}".encode()).hexdigest()[-5:]
```

What is the result? (It should be a 5-character string)

5) Download this image. Create a new Google Colab notebook and run this code (after fixing a mistake in it) to calculate the number of pixels with a certain minimum brightness:

```
import numpy as np
from PIL import Image
from google.colab import files
import colorsys

# There is a mistake in the line below. Fix it
image = Image.open(list(files.upload().keys)[0])
```

```
rgb = np.array(image) / 255.0
lightness = np.apply_along_axis(lambda x: colorsys.rgb_to_hls(*x)[1], 2, rgb)
light_pixels = np.sum(lightness > 0.937)
print(f'Number of pixels with lightness > 0.937: {light_pixels}')
```

What is the result? (It should be a number)

6) Download this q-vercel-python.json which has the marks of 100 imaginary students.

Create and deploy a Python app to Vercel. Expose an API so that when a request like https://your-app.vercel.app/api?name=X&name=Y is made, it returns a JSON response with the marks of the names X and Y in the same order, like this:

```
{ "marks": [10, 20] }
```

Make sure you enable CORS to allow GET requests from any origin.

What is the Vercel URL? It should look like:https://your-app.vercel.app/api

7) Create a GitHub action on one of your GitHub repositories. Make sure one of the steps in the action has a name that contains your email address 23f2003751@ds.study.iitm.ac.in. For example:

```
jobs:
  test:
    steps:
      - name: 23f2003751@ds.study.iitm.ac.in
        run: echo "Hello, world!"
```

Trigger the action and make sure it is the most recent action.

What is your repository URL? It will look like: https://github.com/USER/REPO

8) Create and push an image to Docker Hub. Add a tag named **23f2003751** to the image.

What is the Docker image URL? It should look like:https://hub.docker.com/repository/docker/$USER/$REPO/general

9) If the URL has a query parameter class, it should return only students in those classes. For example, /api?class=1A should return only students in class 1A. /api?class=1A&class=1B should return only students in class 1A and 1B. There may be any number of classes

specified. Return students in the same order as they appear in the CSV file (not the order of the classes).

Make sure you enable CORS to allow GET requests from any origin.

What is the API URL endpoint for FastAPI? It might look like:http://127.0.0.1:8000/api

We'll check by sending a request to this URL with ?class=... added and check if the response matches the data.

10) Download Llamafile. Run the Llama-3.2-1B-Instruct.Q6_K.llamafile model with it.

Create a tunnel to the Llamafile server using ngrok.

What is the ngrok URL? It might look like:https://[random].ngrok-free.app/

GA 3

1) DataSentinel Inc. is a tech company specializing in building advanced natural language processing (NLP) solutions. Their latest project involves integrating an AI-powered sentiment analysis module into an internal monitoring dashboard. The goal is to automatically classify large volumes of unstructured feedback and text data from various sources as either GOOD, BAD, or NEUTRAL. As part of the quality assurance process, the development team needs to test the integration with a series of sample inputs—even ones that may not represent coherent text—to ensure that the system routes and processes the data correctly.

Before rolling out the live system, the team creates a test harness using Python. The harness employs the httpx library to send POST requests to OpenAI's API. For this proof-of-concept, the team uses the dummy model gpt-4o-mini along with a dummy API key in the Authorization header to simulate real API calls.

One of the test cases involves sending a sample piece of meaningless text:

5w yh4uV 5r m e8 9gMbg9VN1 cPFVoPIVZl LIF5 uw kJn

Write a Python program that uses httpx to send a POST request to OpenAI's API to analyze the sentiment of this (meaningless) text into GOOD, BAD or NEUTRAL. Specifically:

Make sure you pass an Authorization header with dummy API key.

Use gpt-4o-mini as the model.

The first message must be a system message asking the LLM to analyze the sentiment of the text. Make sure you mention GOOD, BAD, or NEUTRAL as the categories.

The second message must be exactly the text contained above.

This test is crucial for DataSentinel Inc. as it validates both the API integration and the correctness of message formatting in a controlled environment. Once verified, the same mechanism will be used to process genuine customer feedback, ensuring that the sentiment analysis module reliably categorizes data as GOOD, BAD, or NEUTRAL. This reliability is essential for maintaining high operational standards and swift response times in real-world applications.

Note: This uses a dummy httpx library, not the real one. You can only use:

response = httpx.get(url, **kwargs)

response = httpx.post(url, json=None, **kwargs)

response.raise_for_status()

response.json()

2) LexiSolve Inc. is a startup that delivers a conversational AI platform to enterprise clients. The system leverages OpenAI's language models to power a variety of customer service, sentiment analysis, and data extraction features. Because pricing for these models is based on the number of tokens processed—and strict token limits apply—accurate token accounting is critical for managing costs and ensuring system stability.

To optimize operational costs and prevent unexpected API overages, the engineering team at LexiSolve has developed an internal diagnostic tool that simulates and measures token usage for typical prompts sent to the language model.

One specific test case an understanding of text tokenization. Your task is to generate data for that test case.

Specifically, when you make a request to OpenAI's GPT-4o-Mini with just this user message:

List only the valid English words from these: **YigdW, Z6M, MFj, quEkQcyY, sp, Ok, 42CIKm, aSXECg6tj, X0c, MBva2, jAn, b3M, MV6zvA, nY4, pk, D7M, k9xV, JaY30SkT, o, c6CIIzKsKM, OYzj, QyoM7I, g5taA, r1KVIho3j, UZYTrdZ8, Wb, h, s3BKu8, dhN76C, me8V, I6Woi, GEivxKNf, R8KtzV, Di, E2h, ABa, fmyVjlp, 2n1af, 5, fhZR1, Vpubmqkd, T, yw, Vh5l, PzcgFg, VjaoOh, Zt1s, bPbQgEO8, 2IS, M, pyIvn, WH6hOL2, D3LULHH, rW, YtAz, HUS, fH7HYb11g, xpox8z, t9rqaP5CJA, ybFDvHdVJ, wy5g1HQ, 5ndlSETx, cKyFCQNA, 3yUY, Swoytp, 1sXYdD,**

n2UunZBBAS, 2a3bfddVo, kTIBWim, m5qPNJ, tpuJxs6lSR, lZu93k5HqY, 44xQm, 3NcJI, t9PwSvUvjq, tFTu, jpkMN8nPe, MSmS, H0, 0s, QIJqq1, ahXv5I, vjuu6x5, JFRXm, Y, M9Yeymnp, pEO

... how many input tokens does it use up?

Number of tokens:

Remember: indicating that this is a user message takes up a few extra tokens. You actually need to make the request to get the answer.

3) RapidRoute Solutions is a logistics and delivery company that relies on accurate and standardized address data to optimize package routing. Recently, they encountered challenges with manually collecting and verifying new addresses for testing their planning software. To overcome this, the company decided to create an automated address generator using a language model, which would provide realistic, standardized U.S. addresses that could be directly integrated into their system.

The engineering team at RapidRoute is tasked with designing a service that uses OpenAI's GPT-4o-Mini model to generate fake but plausible address data. The addresses must follow a strict format, which is critical for downstream processes such as geocoding, routing, and verification against customer databases. For consistency and validation, the development team requires that the addresses be returned as structured JSON data with no additional properties that could confuse their parsers.

As part of the integration process, you need to write the body of the request to an OpenAI chat completion call that:

Uses model gpt-4o-mini

Has a system message: Respond in JSON

Has a user message: Generate 10 random addresses in the US

Uses structured outputs to respond with an object addresses which is an array of objects with required fields: **latitude (number) city (string) apartment (string)** .

Sets additionalProperties to false to prevent additional properties.

Note that you don't need to run the request or use an API key; your task is simply to write the correct JSON body.

What is the JSON body we should send to https://api.openai.com/v1/chat/completions for this? (No need to run it or to use an API key. Just write the body of the request below.)

4) Acme Global Solutions manages hundreds of invoices from vendors every month. To streamline their accounts payable process, the company is developing an automated document processing system. This system uses a computer vision model to extract useful text from scanned invoice images. Critical pieces of data such as vendor email addresses, invoice or transaction numbers, and other details are embedded within these documents.

Your team is tasked with integrating OpenAI's vision model into the invoice processing workflow. The chosen model, gpt-4o-mini, is capable of analyzing both text and image inputs simultaneously. When an invoice is received—for example, an invoice image may contain a vendor email like alice.brown@acmeglobal.com and a transaction number such as 34921. The system needs to extract all embedded text to automatically populate the vendor management system.

The automated process will send a POST request to OpenAI's API with two inputs in a single user message:

Text: A simple instruction "Extract text from this image."

Image URL: A base64 URL representing the invoice image that might include the email and the transaction number among other details.

Here is an example invoice image:

IMAGE

Write just the JSON body (not the URL, nor headers) for the POST request that sends these two pieces of content (text and image URL) to the OpenAI API endpoint.

Use gpt-4o-mini as the model.

Send a single user message to the model that has a text and an image_url content (in that order).

The text content should be Extract text from this image.

Send the image_url as a base64 URL of the image above. CAREFUL: Do not modify the image.

Write your JSON body here:

5) SecurePay, a leading fintech startup, has implemented an innovative feature to detect and prevent fraudulent activities in real time. As part of its security suite, the system analyzes personalized transaction messages by converting them into embeddings. These embeddings

are compared against known patterns of legitimate and fraudulent messages to flag unusual activity.

Imagine you are working on the SecurePay team as a junior developer tasked with integrating the text embeddings feature into the fraud detection module. When a user initiates a transaction, the system sends a personalized verification message to the user's registered email address. This message includes the user's email address and a unique transaction code (a randomly generated number). Here are 2 verification messages:

Dear user, please verify your transaction code 30668 sent to 23f2003751@ds.study.iitm.ac.in

Dear user, please verify your transaction code 5965 sent to 23f2003751@ds.study.iitm.ac.in

The goal is to capture this message, convert it into a meaningful embedding using OpenAI's text-embedding-3-small model, and subsequently use the embedding in a machine learning model to detect anomalies.

Your task is to write the JSON body for a POST request that will be sent to the OpenAI API endpoint to obtain the text embedding for the 2 given personalized transaction verification messages above. This will be sent to the endpoint https://api.openai.com/v1/embeddings.

Write your JSON body here:

6) ShopSmart is an online retail platform that places a high value on customer feedback. Each month, the company receives hundreds of comments from shoppers regarding product quality, delivery speed, customer service, and more. To automatically understand and cluster this feedback, ShopSmart's data science team uses text embeddings to capture the semantic meaning behind each comment.

As part of a pilot project, ShopSmart has curated a collection of 25 feedback phrases that represent a variety of customer sentiments. Examples of these phrases include comments like "Fast shipping and great service," "Product quality could be improved," "Excellent packaging," and so on. Due to limited processing capacity during initial testing, you have been tasked with determine which pair(s) of 5 of these phrases are most similar to each other. This similarity analysis will help in grouping similar feedback to enhance the company's understanding of recurring customer issues.

ShopSmart has written a Python program that has the 5 phrases and their embeddings as an array of floats. It looks like this:

embeddings = **{"Packaging was excellent.":[-0.01674579456448555,-0.06481242924928665,...,-0.08907122164964676]}**

Your task is to write a Python function most_similar(embeddings) that will calculate the cosine similarity between each pair of these embeddings and return the pair that has the highest similarity. The result should be a tuple of the two phrases that are most similar.

Write your Python code here:

7) InfoCore Solutions is a technology consulting firm that maintains an extensive internal knowledge base of technical documents, project reports, and case studies. Employees frequently search through these documents to answer client questions quickly or gain insights for ongoing projects. However, due to the sheer volume of documentation, traditional keyword-based search often returns too many irrelevant results.

To address this issue, InfoCore's data science team decides to integrate a semantic search feature into their internal portal. This feature uses text embeddings to capture the contextual meaning of both the documents and the user's query. The documents are pre-embedded, and when an employee submits a search query, the system computes the similarity between the query's embedding and those of the documents. The API then returns a ranked list of document identifiers based on similarity.

Imagine you are an engineer on the InfoCore team. Your task is to build a FastAPI POST endpoint that accepts an array of docs and query string via a JSON body. The endpoint is structured as follows:

POST /similarity

```
{
  "docs": ["Contents of document 1", "Contents of document 2", "Contents of document 3",
...],
  "query": "Your query string"
}
```

Service Flow:

Request Payload: The client sends a POST request with a JSON body containing:

docs: An array of document texts from the internal knowledge base.

query: A string representing the user's search query.

Embedding Generation: For each document in the docs array and for the query string, the API computes a text embedding using text-embedding-3-small.

Similarity Computation: The API then calculates the cosine similarity between the query embedding and each document embedding. This allows the service to determine which documents best match the intent of the query.

Response Structure: After ranking the documents by their similarity scores, the API returns the identifiers (or positions) of the three most similar documents. The JSON response might look like this:

```
{
  "matches": ["Contents of document 3", "Contents of document 1", "Contents of document
2"]
}
```

Here, "Contents of document 3" is considered the closest match, followed by "Contents of document 1", then "Contents of document 2".

Make sure you enable CORS to allow OPTIONS and POST methods, perhaps allowing all origins and headers.

What is the API URL endpoint for your implementation? It might look like: http://127.0.0.1:8000/similarity

We'll check by sending a POST request to this URL with a JSON body containing random docs and query.

8) TechNova Corp. is a multinational corporation that has implemented a digital assistant to support employees with various internal tasks. The assistant can answer queries related to human resources, IT support, and administrative services. Employees use a simple web interface to enter their requests, which may include:

Checking the status of an IT support ticket.

Scheduling a meeting.

Retrieving their current expense reimbursement balance.

Requesting details about their performance bonus.

Reporting an office issue by specifying a department or issue number.

Each question is direct and templatized, containing one or more parameters such as an employee or ticket number (which might be randomized). In the backend, a FastAPI app routes each request by matching the query to one of a set of pre-defined functions. The response that the API returns is used by OpenAI to call the right function with the necessary arguments.

Pre-Defined Functions:

For this exercise, assume the following functions have been defined:

get_ticket_status(ticket_id: int)

schedule_meeting(date: str, time: str, meeting_room: str)

get_expense_balance(employee_id: int)

calculate_performance_bonus(employee_id: int, current_year: int)

report_office_issue(issue_code: int, department: str)

Each function has a specific signature, and the student's FastAPI app should map specific queries to these functions.

Example Questions (Templatized with a Random Number):

Ticket Status: Query: "What is the status of ticket 83742?" → Should map to get_ticket_status(ticket_id=83742)

Meeting Scheduling: Query: "Schedule a meeting on 2025-02-15 at 14:00 in Room A." → Should map to schedule_meeting(date="2025-02-15", time="14:00", meeting_room="Room A")

Expense Reimbursement: Query: "Show my expense balance for employee 10056." → Should map to get_expense_balance(employee_id=10056)

Performance Bonus Calculation: Query: "Calculate performance bonus for employee 10056 for 2025." → Should map to calculate_performance_bonus(employee_id=10056, current_year=2025)

Office Issue Reporting: Query: "Report office issue 45321 for the Facilities department." → Should map to report_office_issue(issue_code=45321, department="Facilities")

Task Overview:

Develop a FastAPI application that:

Exposes a GET endpoint /execute?q=... where the query parameter q contains one of the pre-templatized questions.

Analyzes the q parameter to identify which function should be called.

Extracts the parameters from the question text.

Returns a response in the following JSON format:

{ "name": "function_name", "arguments": "{ ...JSON encoded parameters... }" }

For example, the query "What is the status of ticket 83742?" should return:

```
{
  "name": "get_ticket_status",
  "arguments": "{\"ticket_id\": 83742}"
}
```

Make sure you enable CORS to allow GET requests from any origin.

What is the API URL endpoint for your implementation? It might look like: http://127.0.0.1:8000/execute

We'll check by sending a GET request to this URL with ?q=... containing a task. We'll verify that it matches the expected response. Arguments must be in the same order as the function definition.

9) SecurePrompt Technologies is a cybersecurity firm that specializes in deploying large language models (LLMs) for sensitive enterprise applications. To ensure that these models adhere strictly to security policies, SecurePrompt imposes hardcoded behavioral instructions on the LLMs. For example, an LLM may be configured to never output certain sensitive keywords.

As part of their regular security audits and red-team exercises, SecurePrompt's engineers and external auditors test how well the LLMs follow these strict instructions. One objective of these tests is to determine if it is possible to bypass or trick the LLM into violating its preset security constraints.

This task is simulates potential attack vectors where a malicious actor might manipulate the model's output by ingeniously engineering the prompt. While the intention is to expose vulnerabilities in instruction adherence, it also provides valuable insights into improving the safety and security of the deployed system.

Here's your task: You are chatting with an LLM that has been told to never say Yes. You need to get it to say Yes.

Use your AI Proxy token when prompted.

Write a prompt that will get the LLM to say Yes.

As long as the LLM says the word Yes (case sensitive), you will be marked correct. Careful! If you get a correct answer, submit and don't change it. You may get a different answer next time.

GA 4

1) Your Task

ESPN Cricinfo has ODI batting stats for each batsman. The result is paginated across multiple pages. Count the number of ducks in page number 3.

Understanding the Data Source: ESPN Cricinfo's ODI batting statistics are spread across multiple pages, each containing a table of player data. Go to page number 3.

Setting Up Google Sheets: Utilize Google Sheets' IMPORTHTML function to import table data from the URL for page number 3.

Data Extraction and Analysis: Pull the relevant table from the assigned page into Google Sheets. Locate the column that represents the number of ducks for each player. (It is titled "0".) Sum the values in the "0" column to determine the total number of ducks on that page.

Impact

By automating the extraction and analysis of cricket batting statistics, CricketPro Insights can:

Enhance Analytical Efficiency: Reduce the time and effort required to manually gather and process player performance data.

Provide Timely Insights: Deliver up-to-date statistical analyses that aid teams and coaches in making informed decisions.

Scalability: Easily handle large volumes of data across multiple pages, ensuring comprehensive coverage of player performances.

Data-Driven Strategies: Enable the development of data-driven strategies for player selection, training focus areas, and game planning.

Client Satisfaction: Improve service offerings by providing accurate and insightful analytics that meet the specific needs of clients in the cricketing world.

What is the total number of ducks across players on page number **3** ofESPN Cricinfo's ODI batting stats?

2) To achieve this, you need to:

Extract Data: Retrieve movie information from IMDb for all films that have a rating between 7 and 8.

Format Data: Structure the extracted information into a JSON format containing the following fields:

id: The unique identifier for the movie on IMDb.

title: The official title of the movie.

year: The year the movie was released.

rating: The IMDb user rating for the movie.

Your Task

Source: Utilize IMDb's advanced web search at https://www.imdb.com/search/title/ to access movie data.

Filter: Filter all titles with a rating between **7 and 8**.

Format: For up to the first 25 titles, extract the necessary details: ID, title, year, and rating. The ID of the movie is the part of the URL after tt in the href attribute. For example, tt10078772. Organize the data into a JSON structure as follows:

```
[
  { "id": "tt1234567", "title": "Movie 1", "year": "2021", "rating": "5.8" },
  { "id": "tt7654321", "title": "Movie 2", "year": "2019", "rating": "6.2" },
  // ... more titles
]
```

Submit: Submit the JSON data in the text box below.

Impact

By completing this assignment, you'll simulate a key component of a streaming service's content acquisition strategy. Your work will enable StreamFlix to make informed decisions about which titles to license, ensuring that their catalog remains both diverse and aligned with subscriber preferences. This, in turn, contributes to improved customer satisfaction and retention, driving the company's growth and success in a competitive market.

What is the JSON data?

IMDb search results may differ by region. You may need to manually translate titles. Results may also change periodically. You may need to re-run your scraper code.

3) Your Task

Write a web application that exposes an API with a single query parameter: ?country=. It should fetch the Wikipedia page of the country, extracts all headings (H1 to H6), and create a Markdown outline for the country. The outline should look like this:

```
## Contents

# Vanuatu

## Etymology

## History

### Prehistory

...
```

API Development: Choose any web framework (e.g., FastAPI) to develop the web application. Create an API endpoint (e.g., /api/outline) that accepts a country query parameter.

Fetching Wikipedia Content: Find out the Wikipedia URL of the country and fetch the page's HTML.

Extracting Headings: Use an HTML parsing library (e.g., BeautifulSoup, lxml) to parse the fetched Wikipedia page. Extract all headings (H1 to H6) from the page, maintaining order.

Generating Markdown Outline: Convert the extracted headings into a Markdown-formatted outline. Headings should begin with #.

Enabling CORS: Configure the web application to include appropriate CORS headers, allowing GET requests from any origin.

What is the URL of your API endpoint?

We'll check by sending a request to this URL with ?country=... passing different countries.

4) Your Task

As part of this initiative, you are tasked with developing a system that automates the following:

API Integration and Data Retrieval: Use the BBC Weather API to fetch the weather forecast for Mumbai. Send a GET request to the locator service to obtain the city's locationId. Include necessary query parameters such as API key, locale, filters, and search term (city).

Weather Data Extraction: Retrieve the weather forecast data using the obtained locationId. Send a GET request to the weather broker API endpoint with the locationId.

Data Transformation: Extract the localDate and enhancedWeatherDescription from each day's forecast. Iterate through the forecasts array in the API response and map each localDate to its corresponding enhancedWeatherDescription. Create a JSON object where each key is the localDate and the value is the enhancedWeatherDescription.

The output would look like this:

```
{
  "2025-01-01": "Sunny with scattered clouds",
  "2025-01-02": "Partly cloudy with a chance of rain",
  "2025-01-03": "Overcast skies",
  // ... additional days
}
```

What is the JSON weather forecast description for **Mumbai**?

5) Your Task

What is the minimum latitude of the bounding box of the city Delhi in the country India on the Nominatim API?

API Integration: Use the Nominatim API to fetch geospatial data for a specified city within a country via a GET request to the Nominatim API with parameters for the city and country. Ensure adherence to Nominatim's usage policies, including rate limiting and proper attribution.

Data Retrieval and Filtering: Parse the JSON response from the API. If multiple results are returned (e.g., multiple cities named “Springfield” in different states), filter the results based on the provided osm_id ending to select the correct city instance.

Parameter Extraction: Access the boundingbox attribute. Depending on whether you're looking for the minimum or maximum latitude, extract the corresponding latitude value.

Impact

By automating the extraction and processing of bounding box data, UrbanRide can:

Optimize Routing: Enhance route planning algorithms with precise geographical boundaries, reducing delivery times and operational costs.

Improve Fleet Allocation: Allocate vehicles more effectively across defined service zones based on accurate city extents.

Enhance Market Analysis: Gain deeper insights into regional performance, enabling targeted marketing and service improvements.

Scale Operations: Seamlessly integrate new cities into their service network with minimal manual intervention, ensuring consistent data quality.

What is theminimum latitudeof the bounding box of the city **Delhi** in the country **India** on the Nominatim API?Value of theminimum latitude

6) Your Task

Search using the Hacker News RSS API for the latest Hacker News post mentioning DuckDB and having a minimum of 71 points. What is the link that it points to?

Automate Data Retrieval: Utilize the HNRSS API to fetch the latest Hacker News posts. Use the URL relevant to fetching the latest posts, searching for topics and filtering by a minimum number of points.

Extract and Present Data: Extract the most recent <item> from this result. Get the <link> tag inside it.

Share the result: Type in just the URL in the answer.

What is the link to the latest Hacker News post mentioning **DuckDB** having at least **71** points?

7) Your Task

Using the GitHub API, find all users located in the city **Seattle** with over **130** followers.

When was the newest user's GitHub profile created?

API Integration and Data Retrieval: Leverage GitHub’s search endpoints to query users by location and filter them by follower count.

Data Processing: From the returned list of GitHub users, isolate those profiles that meet the specified criteria.

Sort and Format: Identify the "newest" user by comparing the created_at dates provided in the user profile data. Format the account creation date in the ISO 8601 standard (e.g., "2024-01-01T00:00:00Z").

Impact

By automating this data retrieval and filtering process, CodeConnect gains several strategic advantages:

Targeted Recruitment: Quickly identify new, promising talent in key regions, allowing for more focused and timely recruitment campaigns.

Competitive Intelligence: Stay updated on emerging trends within local developer communities and adjust talent acquisition strategies accordingly.

Efficiency: Automating repetitive data collection tasks frees up time for recruiters to focus on engagement and relationship-building.

Data-Driven Decisions: Leverage standardized and reliable data to support strategic business decisions in recruitment and market research.

Enter the date (ISO 8601, e.g. "2024-01-01T00:00:00Z") when the newest user joined GitHub.

Search using location: and followers: filters, sort by joined descending, fetch the first url, and enter the created_at field. Ignore ultra-new users who JUST joined, i.e. after 3/19/2025, 1:51:09 PM.

8) Your Task

Create a scheduled GitHub action that runs daily and adds a commit to your repository. The workflow should:

Use schedule with cron syntax to run once per day (must use specific hours/minutes, not wildcards)

Include a step with your email **23f2003751@ds.study.iitm.ac.in** in its name

Create a commit in each run

Be located in .github/workflows/ directory

After creating the workflow:

Trigger the workflow and wait for it to complete

Ensure it appears as the most recent action in your repository

Verify that it creates a commit during or within 5 minutes of the workflow run

Enter your repository URL (format: https://github.com/USER/REPO):

9) Your Task

This file, q-extract-tables-from-pdf.pdf contains a table of student marks in Maths, Physics, English, Economics, and Biology.

Calculate the total Biology marks of students who scored 17 or more marks in Physics in groups 43-66 (including both groups).

Data Extraction:: Retrieve the PDF file containing the student marks table and use PDF parsing libraries (e.g., Tabula, Camelot, or PyPDF2) to accurately extract the table data into a workable format (e.g., CSV, Excel, or a DataFrame).

Data Cleaning and Preparation: Convert marks to numerical data types to facilitate accurate calculations.

Data Filtering: Identify students who have scored marks between 17 and Physics in groups 43-66 (including both groups).

Calculation: Sum the marks of the filtered students to obtain the total marks for this specific cohort.

By automating the extraction and analysis of student marks, EduAnalytics empowers Greenwood High School to make informed decisions swiftly. This capability enables the school to:

Identify Performance Trends: Quickly spot areas where students excel or need additional support.

Allocate Resources Effectively: Direct teaching resources and interventions to groups and subjects that require attention.

Enhance Reporting Efficiency: Reduce the time and effort spent on manual data processing, allowing educators to focus more on teaching and student engagement.

Support Data-Driven Strategies: Use accurate and timely data to shape educational strategies and improve overall student outcomes.

What is the total **Biology** marks of students who scored17or more marks in **Physics** in groups **43-66** (including both groups)?

10) Your Task

As part of the Documentation Transformation Project, you are a junior developer at EduDocs tasked with developing a streamlined workflow for converting PDF files to Markdown and ensuring their consistent formatting. This project is critical for supporting EduDocs' commitment to delivering high-quality, accessible educational resources to its clients.

q-pdf-to-markdown.pdf has the contents of a sample document.

Convert the PDF to Markdown: Extract the content from the PDF file. Accurately convert the extracted content into Markdown format, preserving the structure and formatting as much as possible.

Format the Markdown: Use Prettier version 3.4.2 to format the converted Markdown file.

Submit the Formatted Markdown: Provide the final, formatted Markdown file as your submission.

Impact

By completing this exercise, you will contribute to EduDocs Inc.'s mission of providing high-quality, accessible educational resources. Automating the PDF to Markdown conversion and ensuring consistent formatting:

Enhances Productivity: Reduces the time and effort required to prepare documentation for clients.

Improves Quality: Ensures all documents adhere to standardized formatting, enhancing readability and professionalism.

Supports Scalability: Enables EduDocs to handle larger volumes of documentation without compromising on quality.

Facilitates Integration: Makes it easier to integrate Markdown-formatted documents into various digital platforms and content management systems.

What is the markdown content of the PDF, formatted with prettier@3.4.2?

GA 5

1) Improving Sales Data Accuracy for RetailWise Inc.

RetailWise Inc. is a retail analytics firm that supports companies in optimizing their pricing, margins, and inventory decisions. Their reports depend on accurate historical sales data, but legacy data sources are often messy. Recently, RetailWise received an Excel sheet containing 1,000 transaction records that were generated from scanned receipts. Due to OCR inconsistencies and legacy formatting issues, the data in the Excel sheet is not clean.

The Excel file has these columns, and they are messy:

Customer Name: Contains leading/trailing spaces.

Country: Uses inconsistent representations. Instead of 2-letter abbreviations, it also contains other values like "USA" vs. "US", "UK" vs. "U.K", "Fra" for France, "Bra" for Brazil, "Ind" for India.

Date: Uses mixed formats like "MM-DD-YYYY", "YYYY/MM/DD", etc.

Product: Includes a product name followed by a slash and a random code (e.g., "Theta/5x01vd"). Only the product name part (before the slash) is relevant.

Sales and Cost: Contain extra spaces and the currency string ("USD"). In some rows, the Cost field is missing. When the cost is missing, it should be treated as 50% of the Sales value.

TransactionID: Though formatted as four-digit numbers, this field may have inconsistent spacing.

Your Task

You need to clean this Excel data and calculate the total margin for all transactions that satisfy the following criteria:

Time Filter: Sales that occurred up to and including a specified date (Sun Feb 06 2022 18:40:58 GMT+0530 (India Standard Time)).

Product Filter: Transactions for a specific product (Iota). (Use only the product name before the slash.)

Country Filter: Transactions from a specific country (UK), after standardizing the country names.

The total margin is defined as:

Total Margin=Total Sales−Total CostTotal Sales

Your solution should address the following challenges:

Trim and Normalize Strings: Remove extra spaces from the Customer Name and Country fields. Map inconsistent country names (e.g., "USA", "U.S.A", "US") to a standardized format.

Standardize Date Formats: Detect and convert dates from "MM-DD-YYYY" and "YYYY/MM/DD" into a consistent date format (e.g., ISO 8601).

Extract the Product Name: From the Product field, extract the portion before the slash (e.g., extract "Theta" from "Theta/5x01vd").

Clean and Convert Sales and Cost: Remove the "USD" text and extra spaces from the Sales and Cost fields. Convert these fields to numerical values. Handle missing Cost values appropriately (50% of Sales).

Filter the Data: Include only transactions up to and including Sun Feb 06 2022 18:40:58 GMT+0530 (India Standard Time), matching product Iota, and country UK.

Calculate the Margin: Sum the Sales and Cost for the filtered transactions. Compute the overall margin using the formula provided.

By cleaning the data and calculating accurate margins, RetailWise Inc. can:

Improve Decision Making: Provide clients with reliable margin analyses to optimize pricing and inventory.

Enhance Reporting: Ensure historical data is consistent and accurate, boosting stakeholder confidence.

Streamline Operations: Reduce the manual effort needed to clean data from legacy sources.

Download the Sales Excel file: q-clean-up-excel-sales-data.xlsx

What is the total margin for transactions before **Sun Feb 06 2022 18:40:58 GMT+0530** (India Standard Time) for **Iota** sold in **UK** (which may be spelt in different ways)?

You can enter the margin as a percentage (e.g. 12.34%) or a decimal (e.g. 0.1234).

2) Your Task

As a data analyst at EduTrack Systems, your task is to process this text file and determine the number of unique students based on their student IDs. This deduplication is essential to:

Ensure Accurate Reporting: Avoid inflated counts in enrollment and performance reports.

Improve Data Quality: Clean the dataset for further analytics, such as tracking academic progress or resource allocation.

Optimize Administrative Processes: Provide administrators with reliable data to support decision-making.

You need to do the following:

Data Extraction: Read the text file line by line. Parse each line to extract the student ID.

Deduplication: Remove duplicates from the student ID list.

Reporting: Count the number of unique student IDs present in the file.

By accurately identifying the number of unique students, EduTrack Systems will:

Enhance Data Integrity: Ensure that subsequent analyses and reports reflect the true number of individual students.

Reduce Administrative Errors: Minimize the risk of misinformed decisions that can arise from duplicate entries.

Streamline Resource Allocation: Provide accurate student counts for budgeting, staffing, and planning academic programs.

Improve Compliance Reporting: Ensure adherence to regulatory requirements by maintaining precise student records.

Download the text file with student marks q-clean-up-student-marks.txt

How many unique students are there in the file?

3) Peak Usage Analysis for Regional Content

s-anand.net is a personal website that had region-specific music content. One of the site's key sections is telugump3, which hosts music files and is especially popular among the local audience. The website is powered by robust Apache web servers that record detailed access logs. These logs are essential for understanding user behavior, server load, and content engagement.

The author noticed unusual traffic patterns during weekend evenings. To better tailor their content and optimize server resources, they need to know precisely how many successful requests are made to the telugump3 section during peak hours on Tuesday. Specifically, they are interested in:

Time Window: From 8 until before 20.

Request Type: Only GET requests.

Success Criteria: Requests that return HTTP status codes between 200 and 299.

Data Source: The logs for May 2024 stored in a GZipped Apache log file containing 258,074 rows.

The challenge is further complicated by the nature of the log file:

The logs are recorded in the GMT-0500 timezone.

The file format is non-standard in that fields are separated by spaces, with most fields quoted by double quotes, except the Time field.

Some lines have minor formatting issues (41 rows have unique quoting due to how quotes are escaped).

Your Task

As a data analyst, you are tasked with determining how many successful GET requests for pages under telugump3 were made on Tuesday between 8 and 20 during May 2024. This metric will help:

Scale Resources: Ensure that servers can handle the peak load during these critical hours.

Content Planning: Determine the popularity of regional content to decide on future content investments.

Marketing Insights: Tailor promotional strategies for peak usage times.

This GZipped Apache log file (61MB) has 258,074 rows. Each row is an Apache web log entry for the site s-anand.net in May 2024.

Each row has these fields:

IP: The IP address of the visitor

Remote logname: The remote logname of the visitor. Typically "-"

Remote user: The remote user of the visitor. Typically "-"

Time: The time of the visit. E.g. [01/May/2024:00:00:00 +0000]. Not that this is not quoted and you need to handle this.

Request: The request made by the visitor. E.g. GET /blog/ HTTP/1.1. It has 3 space-separated parts, namely (a) Method: The HTTP method. E.g. GET (b) URL: The URL visited. E.g. /blog/ (c) Protocol: The HTTP protocol. E.g. HTTP/1.1

Status: The HTTP status code. If 200 <= Status < 300 it is a successful request

Size: The size of the response in bytes. E.g. 1234

Referer: The referer URL. E.g. https://s-anand.net/

User agent: The browser used. This will contain spaces and might have escaped quotes.

Vhost: The virtual host. E.g. s-anand.net

Server: The IP address of the server.

The fields are separated by spaces and quoted by double quotes ("). Unlike CSV files, quoted fields are escaped via \" and not "". (This impacts 41 rows.)

All data is in the GMT-0500 timezone and the questions are based in this same timezone.

By determining the number of successful GET requests under the defined conditions, we'll be able to:

Optimize Infrastructure: Scale server resources effectively during peak traffic times, reducing downtime and improving user experience.

Strategize Content Delivery: Identify popular content segments and adjust digital content strategies to better serve the audience.

Improve Marketing Efforts: Focus marketing initiatives on peak usage windows to maximize engagement and conversion.

What is the number of successful GET requests for pages under **/telugump3/** from **8:00** until before **20:00** on **Tuesdays**?

4) Bandwidth Analysis for Regional Content

s-anand.net is a personal website that had region-specific music content. One of the site's key sections is kannada, which hosts music files and is especially popular among the local audience. The website is powered by robust Apache web servers that record detailed access logs. These logs are essential for understanding user behavior, server load, and content engagement.

By analyzing the server's Apache log file, the author can identify heavy users and take measures to manage bandwidth, improve site performance, or even investigate potential abuse.

Your Task

This GZipped Apache log file (61MB) has 258,074 rows. Each row is an Apache web log entry for the site s-anand.net in May 2024.

Each row has these fields:

IP: The IP address of the visitor

Remote logname: The remote logname of the visitor. Typically "-"

Remote user: The remote user of the visitor. Typically "-"

Time: The time of the visit. E.g. [01/May/2024:00:00:00 +0000]. Not that this is not quoted and you need to handle this.

Request: The request made by the visitor. E.g. GET /blog/ HTTP/1.1. It has 3 space-separated parts, namely (a) Method: The HTTP method. E.g. GET (b) URL: The URL visited. E.g. /blog/ (c) Protocol: The HTTP protocol. E.g. HTTP/1.1

Status: The HTTP status code. If 200 <= Status < 300 it is a successful request

Size: The size of the response in bytes. E.g. 1234

Referer: The referer URL. E.g. https://s-anand.net/

User agent: The browser used. This will contain spaces and might have escaped quotes.

Vhost: The virtual host. E.g. s-anand.net

Server: The IP address of the server.

The fields are separated by spaces and quoted by double quotes ("). Unlike CSV files, quoted fields are escaped via \" and not "". (This impacts 41 rows.)

All data is in the GMT-0500 timezone and the questions are based in this same timezone.

Filter the Log Entries: Extract only the requests where the URL starts with /kannada/. Include only those requests made on the specified 2024-05-08.

Aggregate Data by IP: Sum the "Size" field for each unique IP address from the filtered entries.

Identify the Top Data Consumer: Determine the IP address that has the highest total downloaded bytes. Reports the total number of bytes that this IP address downloaded.

Across all requests under **kannada**/on **2024-05-08**, how many bytes did the top IP address (by volume of downloads) download?

5) Your Task

As a data analyst at GlobalRetail Insights, you are tasked with extracting meaningful insights from this dataset. Specifically, you need to:

Group Mis-spelt City Names: Use phonetic clustering algorithms to group together entries that refer to the same city despite variations in spelling. For instance, cluster "Tokyo" and "Tokio" as one.

Filter Sales Entries: Select all entries where:

The product sold is Soap.

The number of units sold is at least 60.

Aggregate Sales by City: After clustering city names, group the filtered sales entries by city and calculate the total units sold for each city.

By performing this analysis, GlobalRetail Insights will be able to:

Improve Data Accuracy: Correct mis-spellings and inconsistencies in the dataset, leading to more reliable insights.

Target Marketing Efforts: Identify high-performing regions for the specific product, enabling targeted promotional strategies.

Optimize Inventory Management: Ensure that inventory allocations reflect the true demand in each region, reducing wastage and stockouts.

Drive Strategic Decision-Making: Provide actionable intelligence to clients that supports strategic planning and competitive advantage in the market.

How many units of **Soap** were sold in **Parison** transactions with at least **60** units?

6) Your Task

As a data recovery analyst at ReceiptRevive Analytics, your task is to develop a program that will:

Parse the Sales Data: Read the provided JSON file containing 100 rows of sales data. Despite the truncated data (specifically the missing id), you must accurately extract the sales figures from each row.

Data Validation and Cleanup: Ensure that the data is properly handled even if some fields are incomplete. Since the id is missing for some entries, your focus will be solely on the sales values.

Calculate Total Sales: Sum the sales values across all 100 rows to provide a single aggregate figure that represents the total sales recorded.

By successfully recovering and aggregating the sales data, ReceiptRevive Analytics will enable RetailFlow Inc. to:

Reconstruct Historical Sales Data: Gain insights into past sales performance even when original receipts are damaged.

Inform Business Decisions: Use the recovered data to understand sales trends, adjust inventory, and plan future promotions.

Enhance Data Recovery Processes: Improve methods for handling imperfect OCR data, reducing future data loss and increasing data accuracy.

Build Client Trust: Demonstrate the ability to extract valuable insights from challenging datasets, thereby reinforcing client confidence in ReceiptRevive's services.

Download the data from q-parse-partial-json.jsonl

What is the total sales value?

7) Your Task

As a data analyst at DataSure Technologies, you have been tasked with developing a script that processes a large JSON log file and counts the number of times a specific key, represented by the placeholder LJ, appears in the JSON structure. Your solution must:

Parse the Large, Nested JSON: Efficiently traverse the JSON structure regardless of its complexity.

Count Key Occurrences: Increment a count only when LJ is used as a key in the JSON object (ignoring occurrences of LJ as a value).

Return the Count: Output the total number of occurrences, which will be used by the operations team to assess the prevalence of particular system events or errors.

By accurately counting the occurrences of a specific key in the log files, DataSure Technologies can:

Diagnose Issues: Quickly determine the frequency of error events or specific system flags that may indicate recurring problems.

Prioritize Maintenance: Focus resources on addressing the most frequent issues as identified by the key count.

Enhance Monitoring: Improve automated monitoring systems by correlating key occurrence data with system performance metrics.

Inform Decision-Making: Provide data-driven insights that support strategic planning for system upgrades and operational improvements.

Download the data from q-extract-nested-json-keys.json

How many times does **LJ** appear as a key?

8) Your Task

Your task as a data analyst at EngageMetrics is to write a query that performs the following:

Filter Posts by Date: Consider only posts with a timestamp greater than or equal to a specified minimum time (2025-02-26T00:17:09.465Z), ensuring that the analysis focuses on recent posts.

Evaluate Comment Quality: From these recent posts, identify posts where at least one comment has received more than a given number of useful stars (5). This criterion filters out posts with low or mediocre engagement.

Extract and Sort Post IDs: Finally, extract all the post_id values of the posts that meet these criteria and sort them in ascending order.

By accurately extracting these high-impact post IDs, EngageMetrics can:

Enhance Reporting: Provide clients with focused insights on posts that are currently engaging audiences effectively.

Target Content Strategy: Help marketing teams identify trending content themes that generate high-quality user engagement.

Optimize Resource Allocation: Enable better prioritization for content promotion and further in-depth analysis of high-performing posts.

Write a DuckDB SQL query to find all posts IDs after **2025-02-26T00:17:09.465Z** with at least 1 comment with **5** useful stars, sorted. The result should be a table with a single column calledpost_id, and the relevant post IDs should be sorted in ascending order.

9) Your Task

Access the Video: Use the provided YouTube link to access the mystery story audiobook.

Convert to Audio: Extract the audio for the segment between 51 and 173.1.

Transcribe the Segment: Utilize automated speech-to-text tools as needed.

By producing an accurate transcript of this key segment, Mystery Tales Publishing will be able to:

Boost Accessibility: Provide high-quality captions and text alternatives for hearing-impaired users.

Enhance SEO: Improve the discoverability of their content through better keyword indexing.

Drive Engagement: Use the transcript for social media snippets, summaries, and promotional materials.

Enable Content Analysis: Facilitate further analysis such as sentiment analysis, topic modeling, and reader comprehension studies.

What is the text of the transcript of this Mystery Story Audiobookbetween **51** and **173.1** seconds?

10) Your Task

As a digital forensics analyst at PixelGuard Solutions, your task is to reconstruct the original image from its scrambled pieces. You are provided with:

The 25 individual image pieces (put together as a single image).

A mapping file detailing the original (row, col) position for each piece and its current (row, col) location.

Your reconstructed image will be critical evidence in the investigation. Once assembled, the image must be uploaded to the secure case management system for further analysis by the investigative team.

Understand the Mapping: Review the provided mapping file that shows how each piece's original coordinates (row, col) relate to its current scrambled position.

Reassemble the Image: Using the mapping, reassemble the 5x5 grid of image pieces to reconstruct the original image. You may use an image processing library (e.g., Python's Pillow, ImageMagick, or a similar tool) to automate the reconstruction process.

Output the Reconstructed Image: Save the reassembled image in a lossless format (e.g., PNG or WEBP). Upload the reconstructed image to the secure case management system as required by PixelGuard's workflow.

By accurately reconstructing the scrambled image, PixelGuard Solutions will:

Reveal Critical Evidence: Provide investigators with a clear view of the original image, which may contain important details related to the case.

Enhance Analytical Capabilities: Enable further analysis and digital enhancements that can lead to breakthroughs in the investigation.

Maintain Chain of Custody: Ensure that the reconstruction process is documented and reliable, supporting the admissibility of the evidence in court.

Improve Operational Efficiency: Demonstrate the effectiveness of automated image reconstruction techniques in forensic investigations.

Here is the image. It is a 500x500 pixel image that has been cut into 25 (5x5) pieces:

Here is the mapping of each piece:

Original RowOriginal ColumnScrambled RowScrambled
Column2**100110141020303010414102011241242132214002032214322302334241030233133324433023431401241134204434044**

Upload the reconstructed image by moving the pieces from the scrambled position to the original position: