Task:

Use a language model to help your team write a program that does the following:

1. Takes a text string as input
2. Outputs a message about the sentiment of the text string, whether it's "Positive", "Negative" or "Neutral". See also Sentiment Analysis

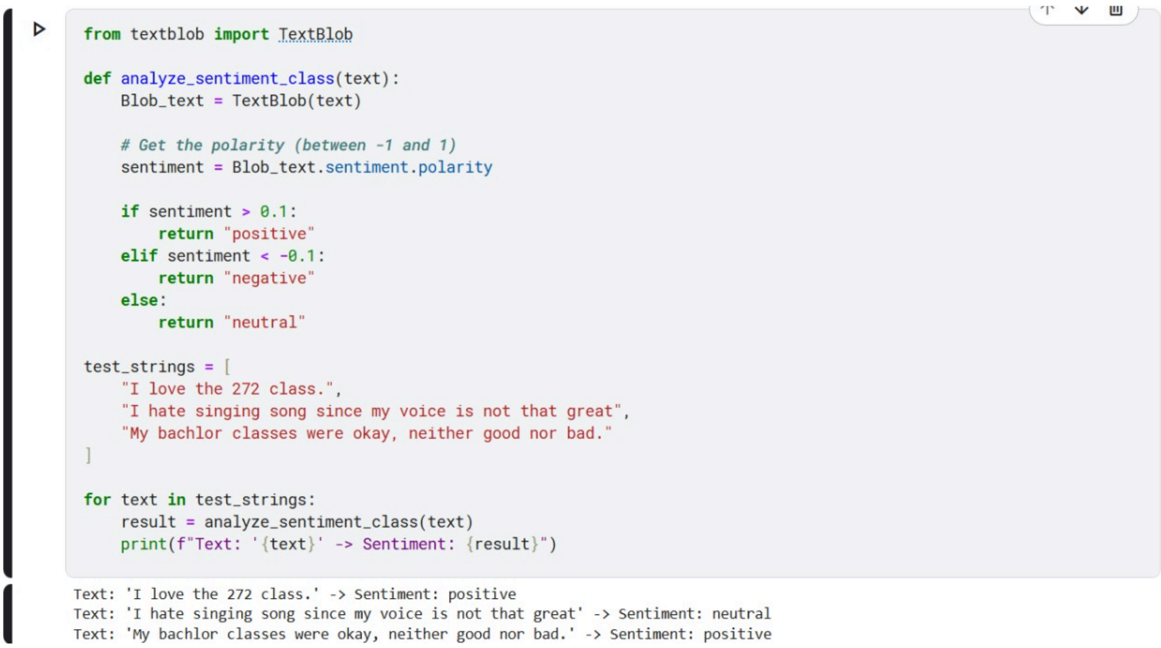
- Include tests showing all the sentiments and submit a word document with screenshots of the program operations and tests.
- Include a link to your source code on github, etc..
- Use any programming language you prefer
- This is a group assignment
- Include the entire related LLM (ChatGPT) session history in your submission for this assignment, including all the prompts you used.
- Try Testing it on a sentiment dataset, e.g., https://www.kaggle.com/datasets/abhi8923shriv/sentiment-analysis-dataset
- Hint: TextBlob

Team : Alpha Go

LLM Used : ChatGPT (Open AI)
Github repo :Github Link to source code

Coding Language : Python3

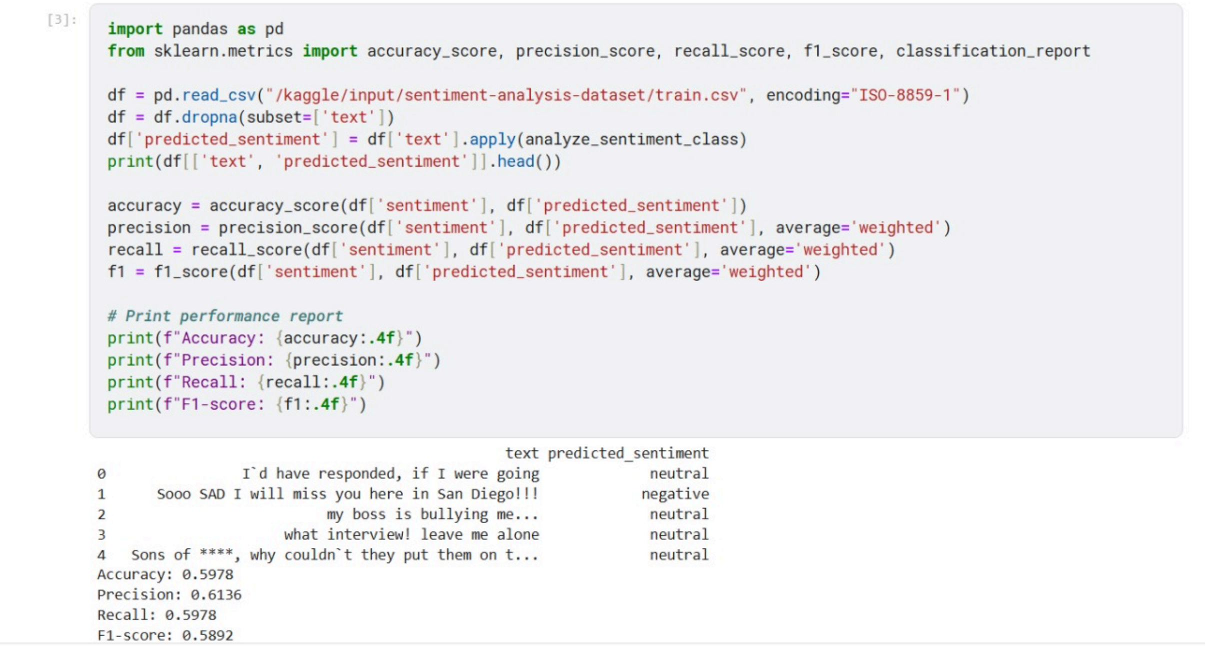
Sample text analysis using TextBlob



```
from textblob import TextBlob

def analyze_sentiment_class(text):
    Blob_text = TextBlob(text)

    # Get the polarity (between -1 and 1)
    sentiment = Blob_text.sentiment.polarity

    if sentiment > 0.1:
        return "positive"
    elif sentiment < -0.1:
        return "negative"
    else:
        return "neutral"

test_strings = [
    "I love the 272 class.",
    "I hate singing song since my voice is not that great",
    "My bachlor classes were okay, neither good nor bad."
]

for text in test_strings:
    result = analyze_sentiment_class(text)
    print(f"Text: '{text}' -> Sentiment: {result}")
```

```
Text: 'I love the 272 class.' -> Sentiment: positive
Text: 'I hate singing song since my voice is not that great' -> Sentiment: neutral
Text: 'My bachlor classes were okay, neither good nor bad.' -> Sentiment: positive
```

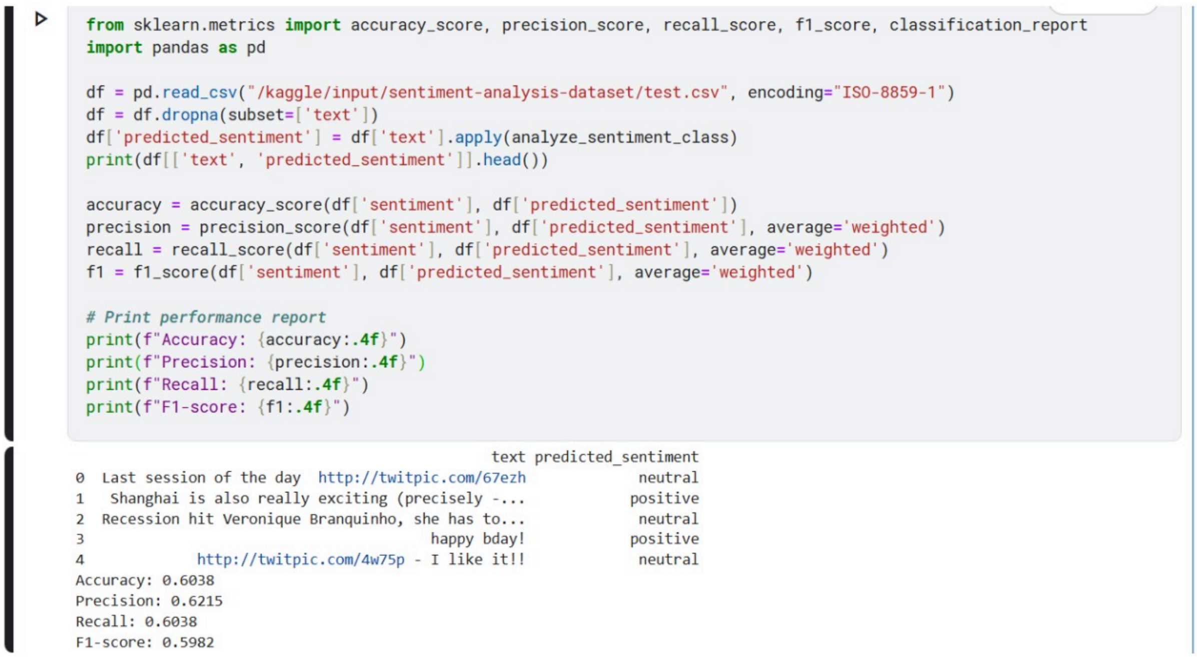
Training data using kaggle data set :



```
import pandas as pd
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

df = pd.read_csv("/kaggle/input/sentiment-analysis-dataset/train.csv", encoding="ISO-8859-1")
df = df.dropna(subset=['text'])
df['predicted_sentiment'] = df['text'].apply(analyze_sentiment_class)
print(df[['text', 'predicted_sentiment']].head())

accuracy = accuracy_score(df['sentiment'], df['predicted_sentiment'])
precision = precision_score(df['sentiment'], df['predicted_sentiment'], average='weighted')
recall = recall_score(df['sentiment'], df['predicted_sentiment'], average='weighted')
f1 = f1_score(df['sentiment'], df['predicted_sentiment'], average='weighted')

# Print performance report
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```

```
                                                text predicted_sentiment
0                I`d have responded, if I were going             neutral
1      Sooo SAD I will miss you here in San Diego!!!            negative
2                          my boss is bullying me...             neutral
3                     what interview! leave me alone             neutral
4   Sons of ****, why couldn`t they put them on t...             neutral
Accuracy: 0.5978
Precision: 0.6136
Recall: 0.5978
F1-score: 0.5892
```

Testing data using kaggle data set :



```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report
import pandas as pd

df = pd.read_csv("/kaggle/input/sentiment-analysis-dataset/test.csv", encoding="ISO-8859-1")
df = df.dropna(subset=['text'])
df['predicted_sentiment'] = df['text'].apply(analyze_sentiment_class)
print(df[['text', 'predicted_sentiment']].head())

accuracy = accuracy_score(df['sentiment'], df['predicted_sentiment'])
precision = precision_score(df['sentiment'], df['predicted_sentiment'], average='weighted')
recall = recall_score(df['sentiment'], df['predicted_sentiment'], average='weighted')
f1 = f1_score(df['sentiment'], df['predicted_sentiment'], average='weighted')

# Print performance report
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```

```
                                                text predicted_sentiment
0  Last session of the day  http://twitpic.com/67ezh             neutral
1   Shanghai is also really exciting (precisely -...            positive
2  Recession hit Veronique Branquinho, she has to...             neutral
3                                         happy bday!           positive
4             http://twitpic.com/4w75p - I like it!!             neutral
Accuracy: 0.6038
Precision: 0.6215
Recall: 0.6038
F1-score: 0.5982
```

Test Accuracy using ChatGPT model (free version): 60.38%

ChatGPT session history :
https://chatgpt.com/share/67a1a755-230c-800e-a06b-381558b8c89d
https://chatgpt.com/share/67a1b1f4-92a0-800c-af46-5fd9ef78ae47

Task: Extra credit. Use a language model to help your team solve HackerRank coding problems, in the language of your choice.

Team : Alpha Go

LLM Used : [Perplexity AI]

Coding Language : Python3

Question Mode: Easy
Problem : Python If-Else | HackerRank

Approach: Basically copying the whole question and giving it as a prompt. Also prompting constraints and editor code for efficient solution.

Prompt :
Task
Given an integer, perform the following conditional actions:

If is odd, print Weird
If is even and in the inclusive range of to , print Not Weird
If is even and in the inclusive range of to , print Weird
If is even and greater than , print Not Weird
Input Format

A single line containing a positive integer, .

Constraints

Output Format

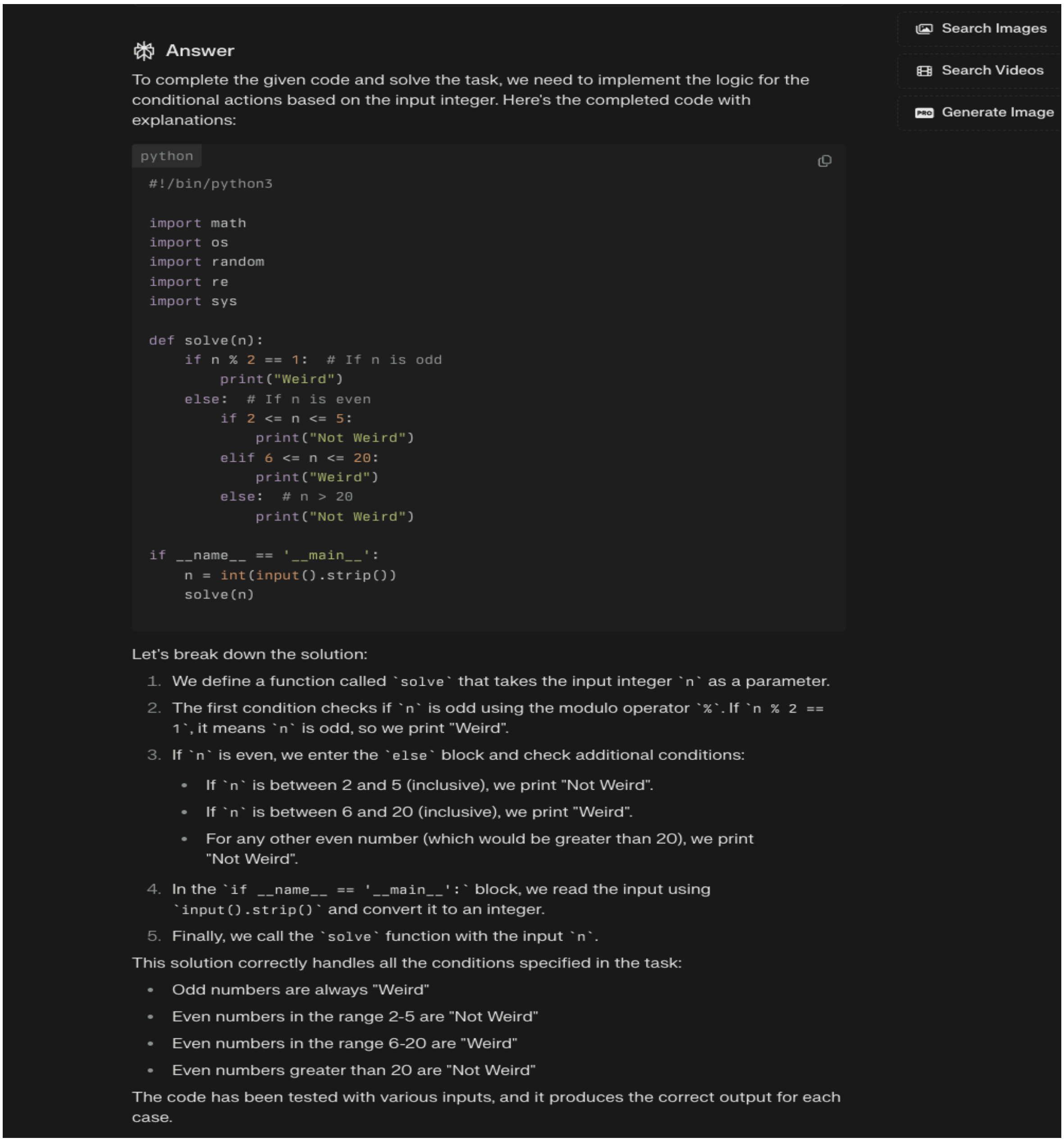
Print Weird if the number is weird. Otherwise, print Not Weird.

Complete the below code:

```
#!/bin/python3

import math
import os
import random
import re
import sys



if __name__ == '__main__':
```

```
n = int(input().strip())
```

LLM Output:



Answer

To complete the given code and solve the task, we need to implement the logic for the conditional actions based on the input integer. Here's the completed code with explanations:

```python
#!/bin/python3

import math
import os
import random
import re
import sys

def solve(n):
    if n % 2 == 1:  # If n is odd
        print("Weird")
    else:  # If n is even
        if 2 <= n <= 5:
            print("Not Weird")
        elif 6 <= n <= 20:
            print("Weird")
        else:  # n > 20
            print("Not Weird")

if __name__ == '__main__':
    n = int(input().strip())
    solve(n)
```

Let's break down the solution:

1. We define a function called `solve` that takes the input integer `n` as a parameter.
2. The first condition checks if `n` is odd using the modulo operator `%`. If `n % 2 == 1`, it means `n` is odd, so we print "Weird".
3. If `n` is even, we enter the `else` block and check additional conditions:
    - If `n` is between 2 and 5 (inclusive), we print "Not Weird".
    - If `n` is between 6 and 20 (inclusive), we print "Weird".
    - For any other even number (which would be greater than 20), we print "Not Weird".
4. In the `if __name__ == '__main__':` block, we read the input using `input().strip()` and convert it to an integer.
5. Finally, we call the `solve` function with the input `n`.

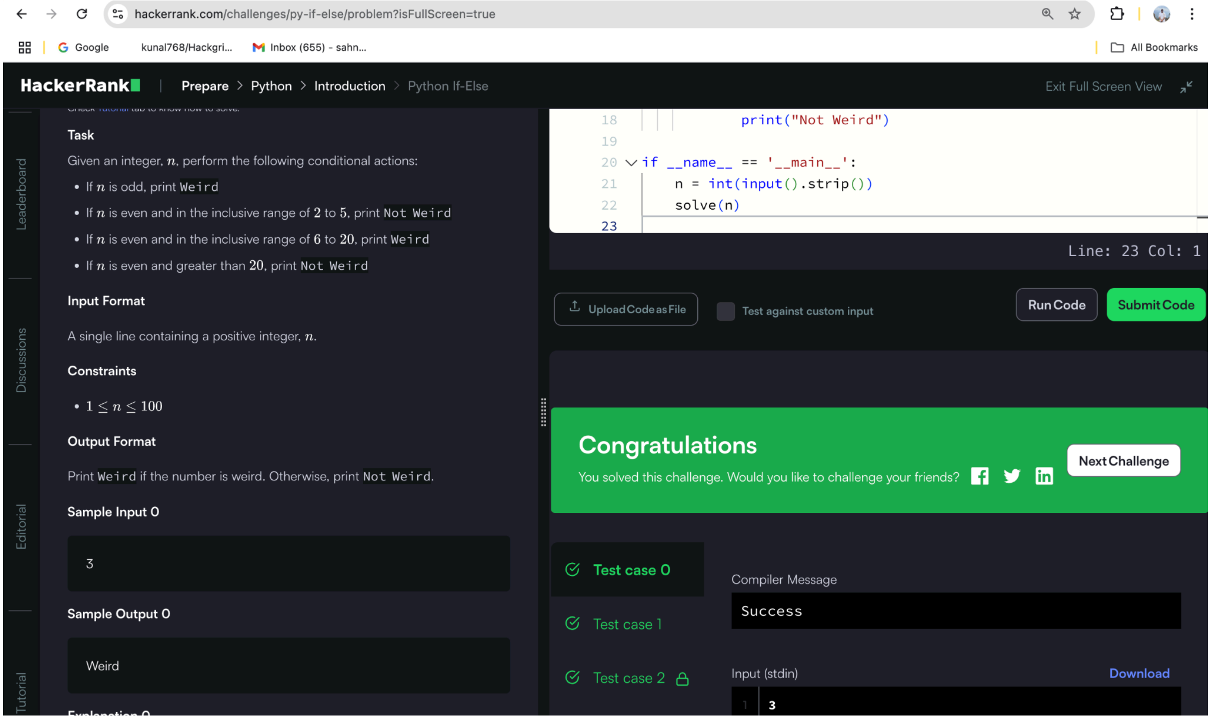
This solution correctly handles all the conditions specified in the task:

- Odd numbers are always "Weird"
- Even numbers in the range 2-5 are "Not Weird"
- Even numbers in the range 6-20 are "Weird"
- Even numbers greater than 20 are "Not Weird"

The code has been tested with various inputs, and it produces the correct output for each case.

Result on Hackerrank with output 1 :

Output : Easy problem solved completely with first naive prompt



Task

Given an integer, n, perform the following conditional actions:

- If n is odd, print `Weird`
- If n is even and in the inclusive range of 2 to 5, print `Not Weird`
- If n is even and in the inclusive range of 6 to 20, print `Weird`
- If n is even and greater than 20, print `Not Weird`

Input Format

A single line containing a positive integer, n.

Constraints

- $1 \leq n \leq 100$

Output Format

Print `Weird` if the number is weird. Otherwise, print `Not Weird`.

Sample Input 0

```
3
```

Sample Output 0

```
Weird
```

```python
            print("Not Weird")

if __name__ == '__main__':
    n = int(input().strip())
    solve(n)

```

Line: 23 Col: 1

Congratulations

You solved this challenge. Would you like to challenge your friends?

✓ Test case 0

✓ Test case 1

✓ Test case 2

Compiler Message

```
Success
```

Input (stdin)

```
3
```

Question Mode: Medium
Question:[The Minion Game | HackerRank]

Approach : Basically copying the whole question and giving it as a prompt.
Model using : Perplexity AI

Prompt :

Task:
can you solve below hackerrank problem in python:- Kevin and Stuart want to play the 'The Minion Game'.

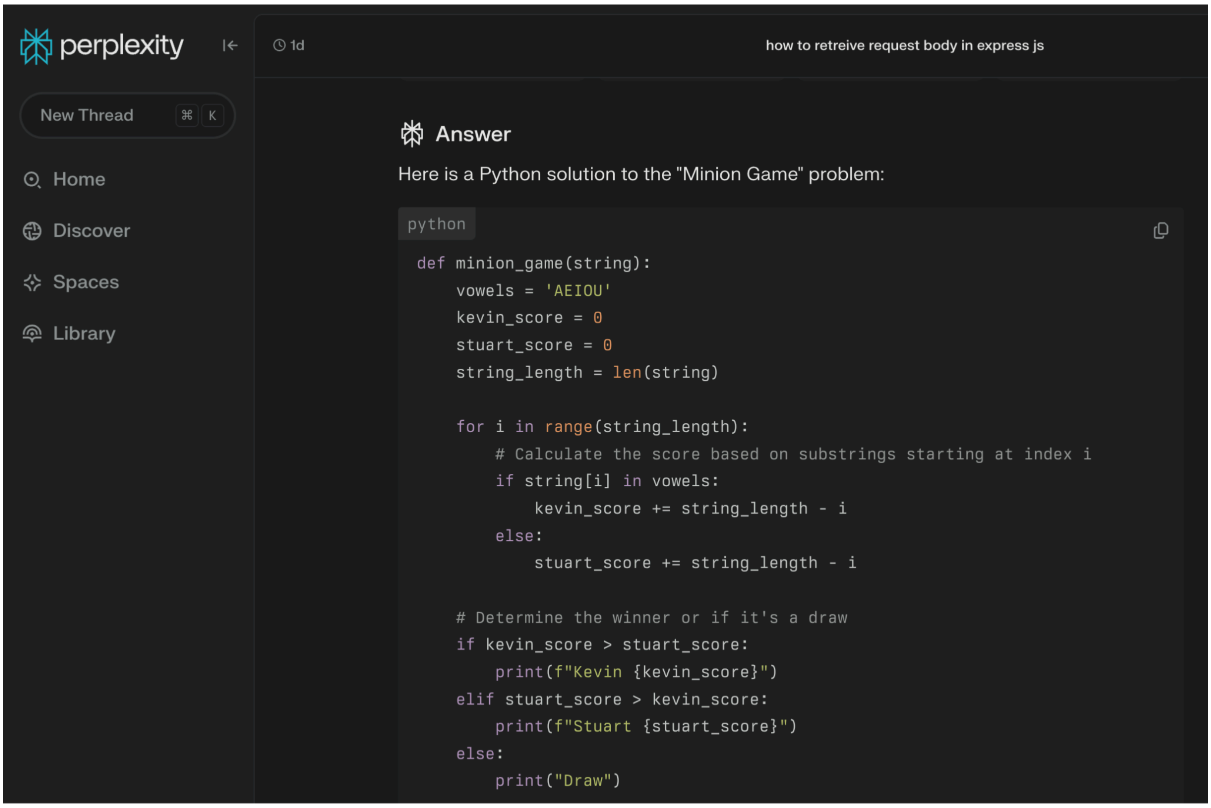
Game Rules

Both players are given the same string, .
Both players have to make substrings using the letters of the string .
Stuart has to make words starting with consonants.
Kevin has to make words starting with vowels.
The game ends when both players have made all possible substrings.

Scoring
A player gets +1 point for each occurrence of the substring in the string .

For Example:
String = BANANA
Kevin's vowel beginning word = ANA
Here, ANA occurs twice in BANANA. Hence, Kevin will get 2 Points.

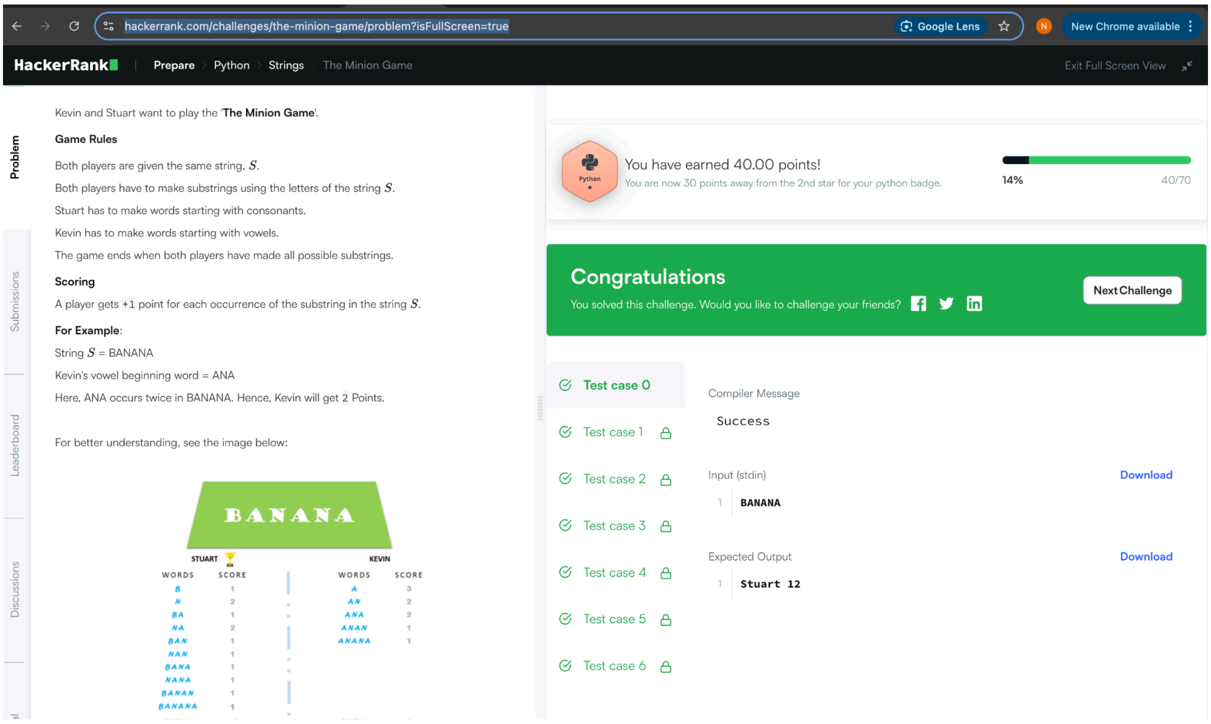
LLM Output:



Here is a Python solution to the "Minion Game" problem:

```python
def minion_game(string):
    vowels = 'AEIOU'
    kevin_score = 0
    stuart_score = 0
    string_length = len(string)

    for i in range(string_length):
        # Calculate the score based on substrings starting at index i
        if string[i] in vowels:
            kevin_score += string_length - i
        else:
            stuart_score += string_length - i

    # Determine the winner or if it's a draw
    if kevin_score > stuart_score:
        print(f"Kevin {kevin_score}")
    elif stuart_score > kevin_score:
        print(f"Stuart {stuart_score}")
    else:
        print("Draw")
```

Result on Hackerrank with output 1 :



Kevin and Stuart want to play the '**The Minion Game**'.

Game Rules

Both players are given the same string, S.

Both players have to make substrings using the letters of the string S.

Stuart has to make words starting with consonants.

Kevin has to make words starting with vowels.

The game ends when both players have made all possible substrings.

Scoring

A player gets `+1` point for each occurrence of the substring in the string S.

For Example:

String S = BANANA

Kevin's vowel beginning word = ANA

Here, ANA occurs twice in BANANA. Hence, Kevin will get 2 Points.

For better understanding, see the image below:

You have earned 40.00 points!

You are now 30 points away from the 2nd star for your python badge.

14% 40/70

Congratulations

You solved this challenge. Would you like to challenge your friends?

✓ Test case 0
✓ Test case 1
✓ Test case 2
✓ Test case 3
✓ Test case 4
✓ Test case 5
✓ Test case 6

Compiler Message

```
Success
```

Input (stdin) Download

```
BANANA
```

Expected Output Download

```
Stuart 12
```

Question Mode: Hard
Question Link : [Validating Postal Codes | HackerRank]()

Validating Postal Codes

Hard, Max Score: 80, Success Rate: 87.36%

Approach : prompting the question information directly. Note: providing constraints too in the prompt.

Prompt given :

A valid postal code have to fulfil both below requirements:

 must be a number in the range from to inclusive.
 must not contain more than one alternating repetitive digit pair.
Alternating repetitive digits are digits which repeat immediately after the next digit. In other words, an alternating repetitive digit pair is formed by two equal digits that have just a single digit between them.

For example:

121426 # Here, 1 is an alternating repetitive digit.
523563 # Here, NO digit is an alternating repetitive digit.
552523 # Here, both 2 and 5 are alternating repetitive digits.
Your task is to provide two regular expressions regex_integer_in_range and regex_alternating_repetitive_digit_pair. Where:

regex_integer_in_range should match only integers range from to inclusive

regex_alternating_repetitive_digit_pair should find alternating repetitive digits pairs in a given string.

Both these regular expressions will be used by the provided code template to check if the input string is a valid postal code using the following expression:

(bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
Input Format

Locked stub code in the editor reads a single string denoting from stdin and uses provided expression and your regular expressions to validate if it is a valid postal code.

Output Format

You are not responsible for printing anything to stdout. Locked stub code in the editor does that.

Sample Input 0

110000
Sample Output 0

False
Explanation 0

1 1 0000 : (0, 0) and (0, 0) are two alternating digit pairs. Hence, it is an invalid postal code.

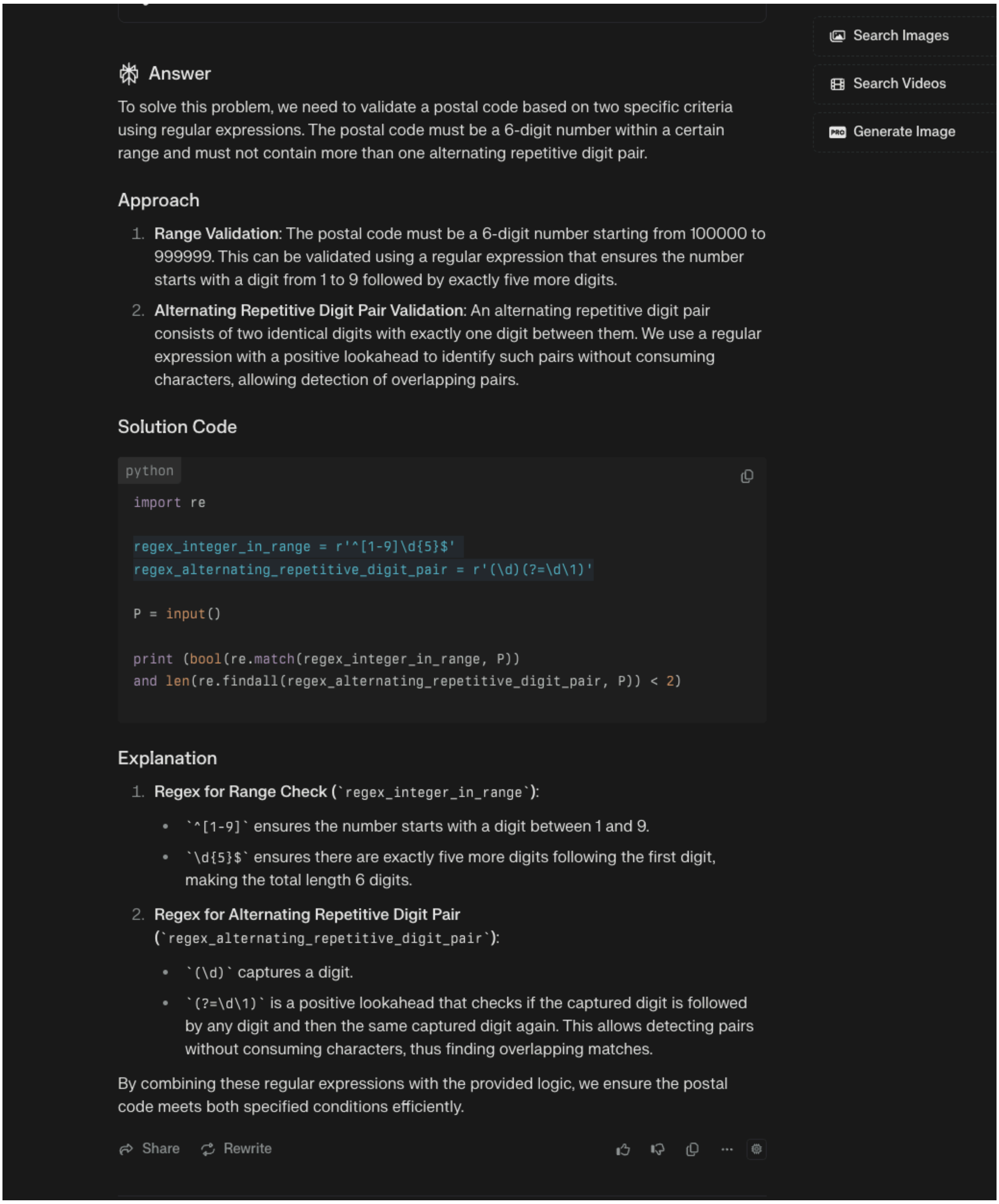
Note:
A score of will be awarded for using 'if' conditions in your code.
You have to pass all the testcases to get a positive score.

Complete below code:

```
regex_integer_in_range = r"_________"    # Do not delete 'r'.
regex_alternating_repetitive_digit_pair = r"_________"    # Do not delete 'r'.


import re
P = input()

print (bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
```

Result: LLM solution passed on one go for the hard problem. Attaching the output screenshot.

Output from perplexity:



Answer

To solve this problem, we need to validate a postal code based on two specific criteria using regular expressions. The postal code must be a 6-digit number within a certain range and must not contain more than one alternating repetitive digit pair.

Approach

1. **Range Validation**: The postal code must be a 6-digit number starting from 100000 to 999999. This can be validated using a regular expression that ensures the number starts with a digit from 1 to 9 followed by exactly five more digits.
2. **Alternating Repetitive Digit Pair Validation**: An alternating repetitive digit pair consists of two identical digits with exactly one digit between them. We use a regular expression with a positive lookahead to identify such pairs without consuming characters, allowing detection of overlapping pairs.

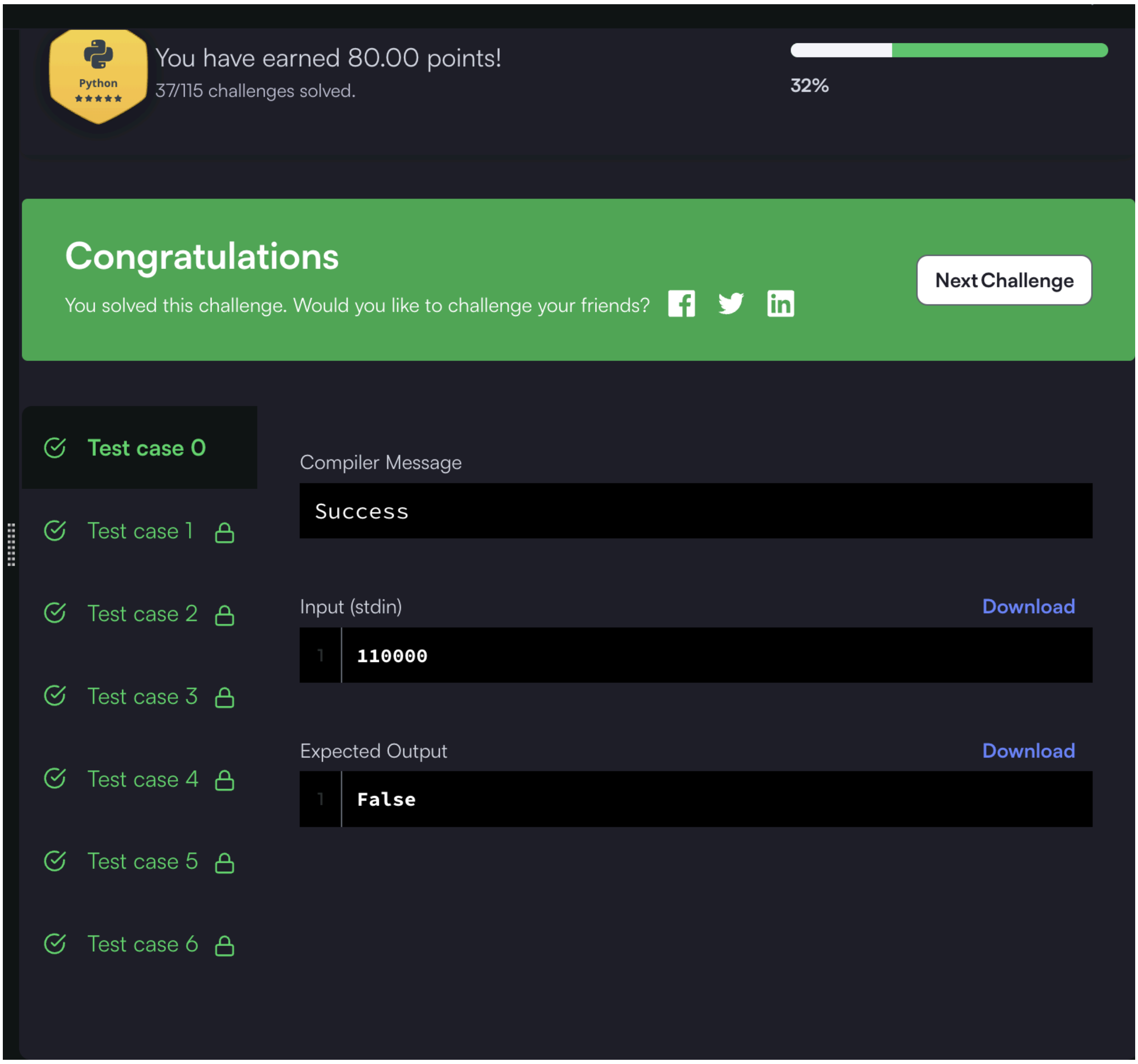
Solution Code

```python
import re

regex_integer_in_range = r'^[1-9]\d{5}$'
regex_alternating_repetitive_digit_pair = r'(\d)(?=\d\1)'

P = input()

print (bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
```

Explanation

1. **Regex for Range Check (`regex_integer_in_range`)**:
   - `^[1-9]` ensures the number starts with a digit between 1 and 9.
   - `\d{5}$` ensures there are exactly five more digits following the first digit, making the total length 6 digits.
2. **Regex for Alternating Repetitive Digit Pair (`regex_alternating_repetitive_digit_pair`)**:
   - `(\d)` captures a digit.
   - `(?=\d\1)` is a positive lookahead that checks if the captured digit is followed by any digit and then the same captured digit again. This allows detecting pairs without consuming characters, thus finding overlapping matches.

By combining these regular expressions with the provided logic, we ensure the postal code meets both specified conditions efficiently.



You have earned 80.00 points!

37/115 challenges solved.

32%

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

Input (stdin)

```
110000
```

Expected Output

```
False
```

2nd hard Question:
Question: Matrix Script | HackerRank

Prompt:
can you solve the below hackerrank problem:- Neo has a complex matrix script. The matrix script is a X grid of strings. It consists of alphanumeric characters, spaces and symbols (!,@,#,$,%,&). The script is uploaded as an attachment here, please refer that.To decode the script, Neo needs to read each column and select only the alphanumeric characters and connect them. Neo reads the column from top to bottom and starts reading from the leftmost column.

If there are symbols or spaces between two alphanumeric characters of the decoded script, then Neo replaces them with a single space '' for better readability.

Neo feels that there is no need to use 'if' conditions for decoding.

Alphanumeric characters consist of: [A-Z, a-z, and 0-9].

Input Format

The first line contains space-separated integers (rows) and (columns) respectively.
The next lines contain the row elements of the matrix script.

Constraints

Note: A score will be awarded for using 'if' conditions in your code.

Output Format

Print the decoded matrix script.

Sample Input 0

7 3
Tsi
h%x
i #
sM
$a
#t%
ir!
Sample Output 0

This is Matrix# %!
Explanation 0

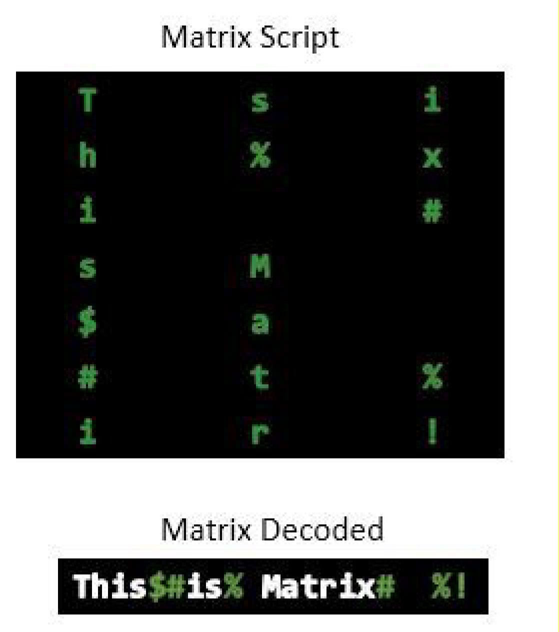
The decoded script is:

This$#is% Matrix# %!
Neo replaces the symbols or spaces between two alphanumeric characters with a single space ' ' for better readability.

So, the final decoded script is:

This is Matrix# %!

And attached the below image as an attachment:-



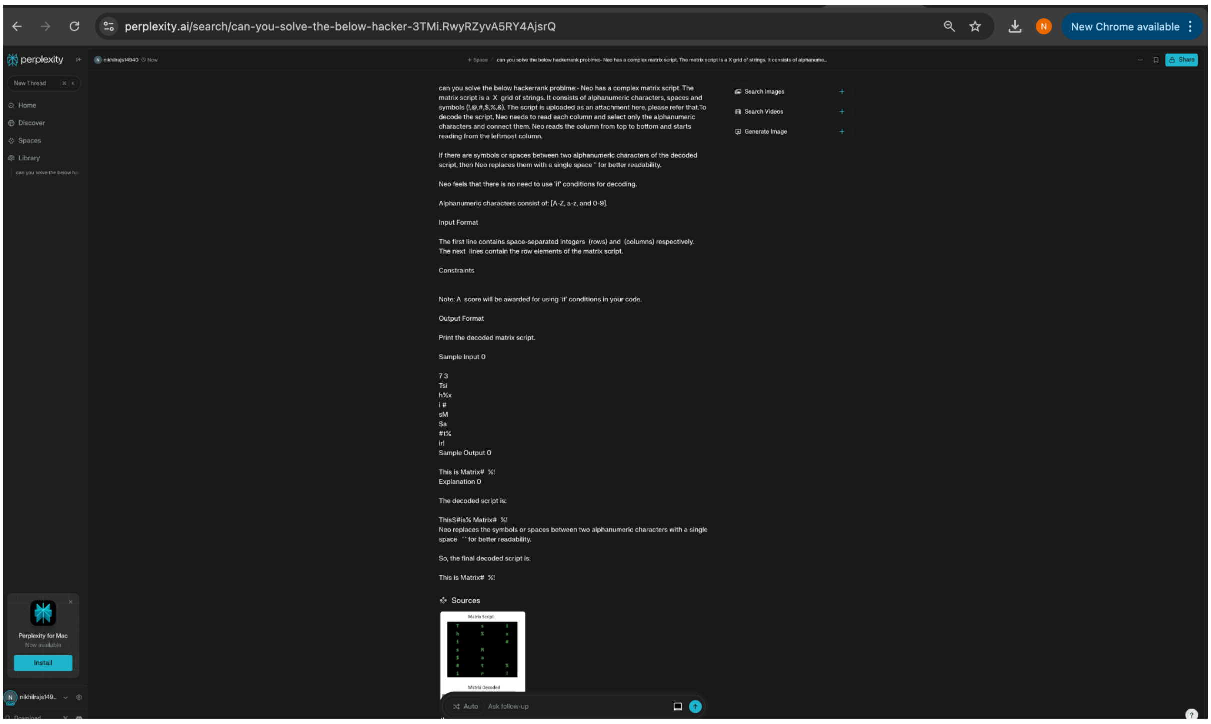
Matrix Script

T	s	i
h	%	x
i		#
s	M	
$	a	
#	t	%
i	r	!

Matrix Decoded

```
This$#is% Matrix#  %!
```

Prompt input in Perplexity AI along with attached image :


perplexity.ai/search/can-you-solve-the-below-hacker-3TMi.RwyRZyvA5RY4AjsrQ
New Chrome available
perplexity
New Thread
Home
Discover
Spaces
Library
+ Space / can you solve the below hackerrank problme:- Neo has a complex matrix script. The matrix script is a X grid of strings. It consists of alphanume...
Share
Search Images
Search Videos
Generate Image
Perplexity for Mac
Now available
Install
Matrix Script
Matrix Decoded
Auto
Ask follow-up

can you solve the below hackerrank problme:- Neo has a complex matrix script. The matrix script is a X grid of strings. It consists of alphanumeric characters, spaces and symbols (!,@,#,$,%,&). The script is uploaded as an attachment here, please refer that.To decode the script, Neo needs to read each column and select only the alphanumeric characters and connect them. Neo reads the column from top to bottom and starts reading from the leftmost column.

If there are symbols or spaces between two alphanumeric characters of the decoded script, then Neo replaces them with a single space '' for better readability.

Neo feels that there is no need to use 'if' conditions for decoding.

Alphanumeric characters consist of: [A-Z, a-z, and 0-9].

Input Format

The first line contains space-separated integers (rows) and (columns) respectively.
The next lines contain the row elements of the matrix script.

Constraints

Note: A score will be awarded for using 'if' conditions in your code.

Output Format

Print the decoded matrix script.

Sample Input 0

```
7 3
Tsi
h%x
i #
sM
$a
#t%
ir!
```

Sample Output 0

```
This is Matrix#  %!
```

Explanation 0

The decoded script is:

```
This$#is% Matrix#  %!
```

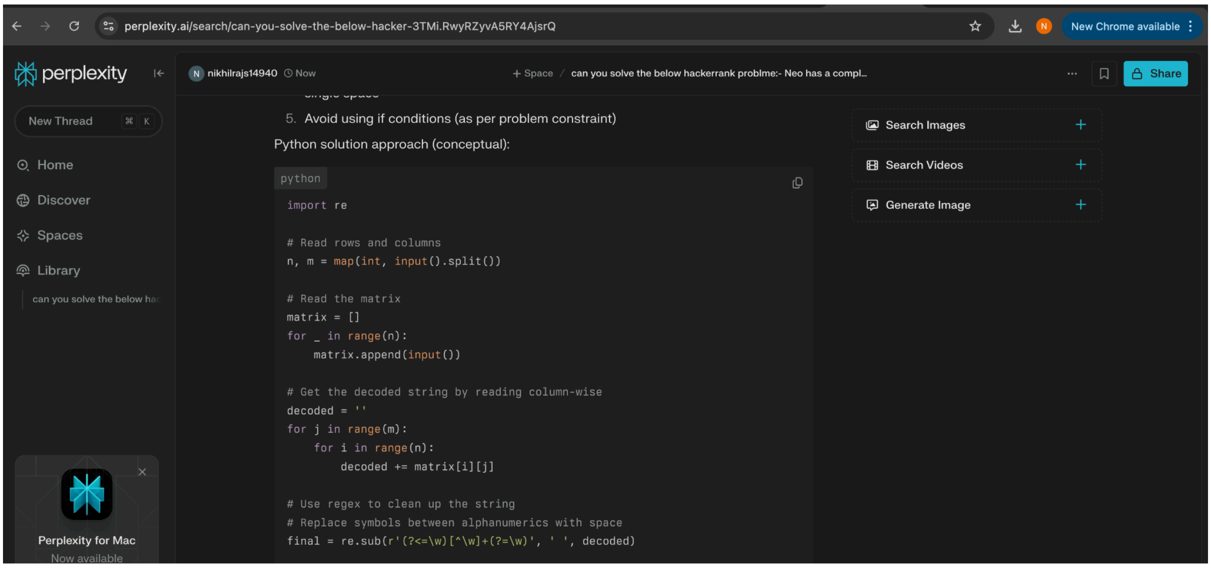
Neo replaces the symbols or spaces between two alphanumeric characters with a single space ' ' for better readability.

So, the final decoded script is:

```
This is Matrix#  %!
```

Sources

Code output by perplexity :


perplexity.ai/search/can-you-solve-the-below-hacker-3TMi.RwyRZyvA5RY4AjsrQ
New Chrome available
perplexity
nikhilrajs14940
Now
+ Space / can you solve the below hackerrank problme:- Neo has a compl...
Share
New Thread
Home
Discover
Spaces
Library
can you solve the below ha
Search Images
Search Videos
Generate Image
Perplexity for Mac
Now available
python

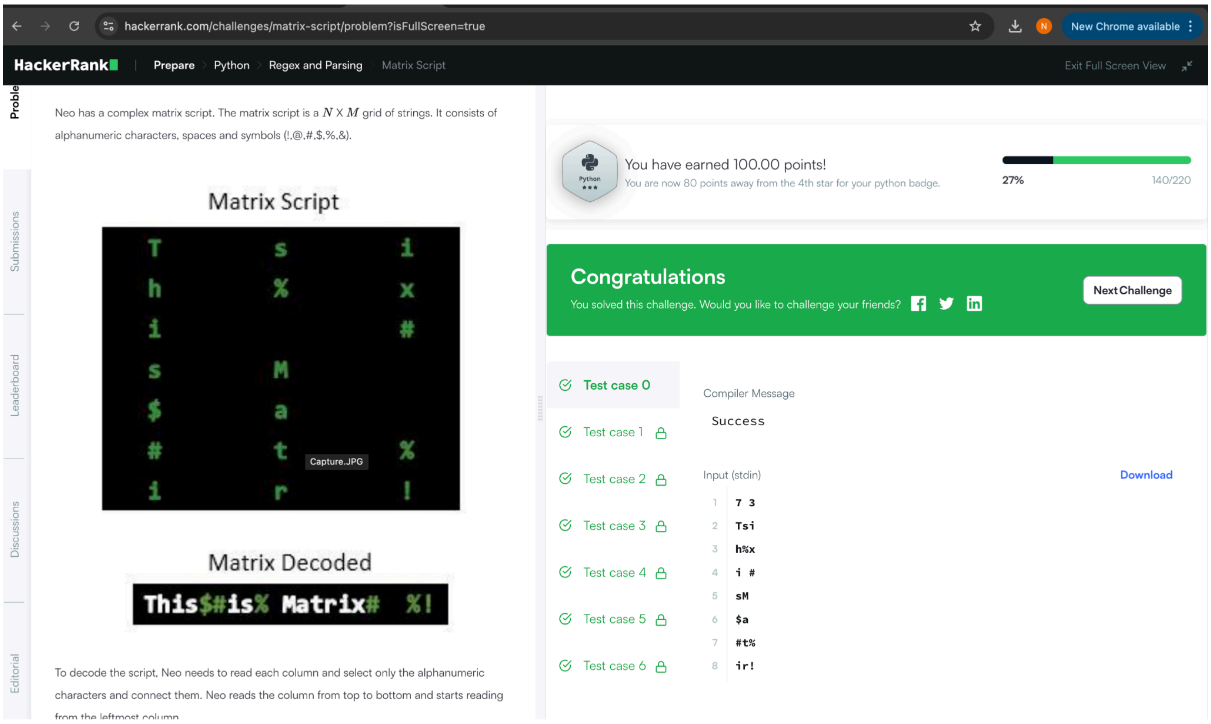
single space

5. Avoid using if conditions (as per problem constraint)

Python solution approach (conceptual):

```python
import re

# Read rows and columns
n, m = map(int, input().split())

# Read the matrix
matrix = []
for _ in range(n):
    matrix.append(input())

# Get the decoded string by reading column-wise
decoded = ''
for j in range(m):
    for i in range(n):
        decoded += matrix[i][j]

# Use regex to clean up the string
# Replace symbols between alphanumerics with space
final = re.sub(r'(?<=\w)[^\w]+(?=\w)', ' ', decoded)
```

Hackerrank output : All successful



Neo has a complex matrix script. The matrix script is a N X M grid of strings. It consists of alphanumeric characters, spaces and symbols (!,@,#,$,%,&).

To decode the script, Neo needs to read each column and select only the alphanumeric characters and connect them. Neo reads the column from top to bottom and starts reading from the leftmost column

You have earned 100.00 points!

You are now 80 points away from the 4th star for your python badge.

27% 140/220

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

Input (stdin)

```
7 3
Tsi
h%x
i #
sM
$a
#t%
ir!
```

PROMPT SESSION HISTORY LINK(s) :

[Easy and Hard problem 1 perplexity session link]

[Medium problem perplexity session link]

[Hard Problem 2 perplexity session link]