

H.NO:2303A51679

Batch :23

### Lab assignment 4.1 Q1:

**Prompt:#Write a Python function that classifies a given text as Spam or Not Spam Code:**

```
def classify_text(text):
    spam_keywords = ['win', 'free', 'prize', 'click', 'buy now', 'limited time', 'offer', 'winner']
    text_lower = text.lower()

    for keyword in spam_keywords:
        if keyword in text_lower:
            return "Spam"

    return "Not Spam" #
```

Example usage

```
texts = [
    "Congratulations! You have won a free lottery ticket."
    "Hello, how are you doing today?",
    "Limited time offer! Click here to buy now.",
    "Don't forget our meeting tomorrow."
]

classifications = {text: classify_text(text) for text in texts}
print(classifications)

# Output: {'Congratulations! You have won a free lottery ticket.': 'Spam', 'Hello, how are you doing today?': 'Not Spam', 'Limited time offer! Click here to buy now.': 'Spam', "Don't forget our meeting tomorrow.": 'Not Spam'}
```

**Output:**

```
{'Congratulations! You have won a free lottery ticket.Hello, how are you doing today?': 'Spam',
```

'Limited time offer! Click here to buy now.': 'Spam', "Don't forget our meeting tomorrow.": 'Not Spam'}

**Q2:prompt:**

""

i am happy today. output:happy

""

#write a python program that detects the emotion of a sentence and

emotions=[happy,sad,angry,excited,nervous,neutral] def

detect\_emotion(sentence):

    emotions = ['happy', 'sad', 'angry', 'excited', 'nervous', 'neutral']

    sentence = sentence.lower()

    if 'happy' in sentence or 'joy' in sentence or 'pleased' in sentence:

        return 'happy'    elif 'sad' in sentence or 'unhappy' in sentence or  
    'sorrow' in sentence:    return 'sad'

    elif 'angry' in sentence or 'mad' in sentence or 'furious' in sentence:

        return 'angry'    elif 'excited' in sentence or 'thrilled' in sentence or  
    'eager' in sentence:

        return 'excited'

    elif 'nervous' in sentence or 'anxious' in sentence or 'worried' in sentence:

        return 'nervous'

    else:

        return 'neutral'

# Example usage sentence = "i am

very sad"  emotion =

detect\_emotion(sentence)

print(emotion)

# Output: happy

**Q3**

**Prompt:**

""

**90–100 → A**

**80–89 → B** **70–**

**79 → C**

**60–69 → D**

**Below 60 → F**

""

**#write a python program that predict a student's grade based on marks.**

```
marks = int(input("Enter the student's marks: ")) if 90 <= marks <= 100:
```

```
    grade = 'A'
```

```
elif 80 <= marks < 90:
```

```
    grade = 'B'
```

```
elif 70 <= marks < 80:
```

```
    grade = 'C'
```

```
elif 60 <= marks < 70:
```

```
    grade = 'D' else:
```

```
    grade = 'F'
```

```
print(f"The student's grade is: {grade}")
```

Output: Enter the student's marks: 88

The student's grade is: B

**Q4:**

**Prompt:**

""

**March → Mesha**

**April → Vrishabha**

**May → Mithuna**

**June → Karka**

**July → Simha**

**August → Kanya**

**September → Tula**

**October → Vrischika November**

**→ Dhanu**

**December → Makara**

**January → Kumbha**

**February → Meena**

**...**

**#write a python program that predicts a persons zodiac sign based on month of birth**

```
def zodiac_sign(month):    zodiac_dict = {
```

```
    "March": "Mesha",
```

```
    "April": "Vrishabha",
```

```
    "May": "Mithuna",
```

```
    "June": "Karka",
```

```
    "July": "Simha",
```

```
    "August": "Kanya",
```

```
    "September": "Tula",
```

```
    "October": "Vrischika",
```

```
    "November": "Dhanu",
```

```
    "December": "Makara",
```

```
    "January": "Kumbha",
```

```
    "February": "Meena"
```

```
}
```

```
return zodiac_dict.get(month, "Invalid month")
```

```
month_input = input("Enter your month of birth: ")
```

```
print(f"Your zodiac sign is: {zodiac_sign(month_input)}")
```

output: Enter your month of birth: October

Your zodiac sign is: Vrischika

Q5:

**Prompt:**

""

**Read the marks from the user.**

**based on marks determine whether student passed or not.**

**handle invalid inputs**

""

**Code:**

```
def check_pass_fail(marks):  
    try:  
        marks = float(marks)      if  
        marks < 0 or marks > 100:  
            raise ValueError("Marks should be between 0 and 100.")
```

```
    if marks >= 40:
```

```
        return "Passed"
```

```
    else:
```

```
        return "Failed"
```

```
except ValueError as e:
```

```
    return str(e)
```

```
print("Enter the marks obtained by the student:")
```

```
user_input = input() result =
```

```
check_pass_fail(user_input) print(result) Output:
```

45

Passed **Q6.**

**Prompt:**

'''

**read the age of the user print  
he can vote or not handle  
invalid cases**

'''

Code:

```
def can_vote(age):
```

```
    try:
```

```
        age = int(age)      if age < 0:      return
```

```
"Invalid age. Age cannot be negative."
```

```
    elif age >= 18:
```

```
        return "You are eligible to vote."
```

```
    else:
```

```
        return "You are not eligible to vote."
```

```
except ValueError:
```

```
    return "Invalid input. Please enter a valid age." user_age
```

```
= input("Enter your age: ") print(can_vote(user_age))
```

Output: Enter your age: 19

You are eligible to vote.

**Q7:**

**Prompt:**

'''

**Generate a list that named names consists of some names from list names if  
the name is palindrome add it to the list palindrome\_names handle invalid  
cases**

'''

```
def is_palindrome(name):
```

```

# Check if the name is a palindrome
return name == name[::-1]

def get_palindrome_names(names):
    palindrome_names = []
    for name in names:
        if not isinstance(name, str):
            print(f"Invalid name: {name}. Skipping.")
            continue
        if is_palindrome(name):
            palindrome_names.append(name)
    return palindrome_names
names = ["Anna", "Bob", "Cathy", "David",
"Eve", 123, None, "Madam"]
palindrome_names = get_palindrome_names(names)
print("Palindrome names:", palindrome_names)

```

**Output:**

[Anna,bob,David,eve]

Q8:

Prompt:

""

generate a list of words named words\_list traverse the list and calculate the length of each word. if length is less than 5 make a list and add as "short" to list if length greater than 7 add as "long" to list

""

Code:

```

words_list = ["apple", "banana", "kiwi", "strawberry", "fig", "watermelon", "pear", "grape"]
length_category = []
for word in words_list:
    if len(word) < 5:
        length_category.append("short")
    elif len(word) > 7:

```

```
length_category.append("long")
print(length_category)
```

**Output:** ['short', 'long', 'short', 'long', 'short']