Course Title	AI Assisted Coding

Name: G. Snehith Roll Number: 2303A52324 Batch: 45

Assignment Number: 4.5

Problem-1

Task-1:

Prompt: Classify the following email into one of the categories: Billing, Technical Support, Feedback, Others.

Email: “I was charged twice for my subscription this month.”

give structured python code

Code:

```python
"""Task-1"""
from typing import List
def classify_email(email: str) -> str:
    text = email.lower()
    billing_keywords: List[str] = ["charge", "charged", "billing", "refund", "payment", "invoice", "double", "subscription"]
    tech_keywords: List[str] = ["error", "bug", "issue", "crash", "not working", "cannot", "can't", "login", "password", "connect", "slow"]
    feedback_keywords: List[str] = ["feedback", "suggestion", "feature request", "feature", "improve", "comment", "love", "hate"]
    if any(k in text for k in billing_keywords):
        return "Billing"
    if any(k in text for k in tech_keywords):
        return "Technical Support"
    if any(k in text for k in feedback_keywords):
        return "Feedback"
    return "Others"
if __name__ == "__main__":
    email = "I was charged twice for my subscription this month."
    print(classify_email(email))
```

Output:

```
PS D:\3-2\AI Assitant Coding>  & 'c:\Users\SNEH
on.debugpy-2025.18.0-win32-x64\bundled\libs\deb
Billing
PS D:\3-2\AI Assitant Coding>
```

Task-2:

Prompt: Classify the following email into one of the categories: Billing, Technical Support, Feedback, Others.

Example:

Email: “My payment failed but the amount was deducted.”

Category: Billing

Now classify:

Email: “The app crashes whenever I try to upload a file.”

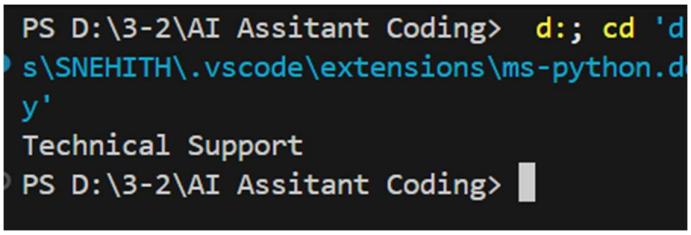
Give me structured python code

Code:

```python
def categorize_email(msg: str) -> str:
    text = msg.lower()
    billing_keywords = {"charge", "charged", "billing", "refund", "payment", "invoice", "double", "subscription"}
    tech_keywords = {"error", "bug", "issue", "crash", "crashes", "not working", "cannot", "can't", "login", "password", "connect", "slow",
    feedback_keywords = {"feedback", "suggestion", "feature request", "feature", "improve", "comment", "love", "hate"}
    if any(k in text for k in billing_keywords):
        return "Billing"
    if any(k in text for k in tech_keywords):
        return "Technical Support"
    if any(k in text for k in feedback_keywords):
        return "Feedback"
    return "Others"

if __name__ == "__main__":
    email = "The app crashes whenever I try to upload a file."
    print(categorize_email(email))
```

Output:



Task-3:

Prompt: Classify the following email into one of the categories: Billing, Technical Support, Feedback, Others.

Examples:

Email: “I was charged twice for my subscription.” → Billing

Email: “The app crashes when I open it.” → Technical Support

Email: “I love the new update.” → Feedback

Email: “What are your support working hours?” → Others

Now classify:

Email: “Invoice amount does not match my usage.”

Give A structured python code

Code:

```python
def classify_email_v3(text: str) -> str:
    t = text.lower()
    billing_kw = {"invoice", "amount", "usage", "bill", "charged", "payment", "refund", "subscription"}
    tech_kw = {"error", "bug", "issue", "crash", "not working", "cannot", "can't", "login", "password", "connect", "slow", "upload"}
    feedback_kw = {"feedback", "suggestion", "feature", "improve", "love", "hate", "comment"}
    if any(k in t for k in billing_kw):
        return "Billing"
    if any(k in t for k in tech_kw):
        return "Technical Support"
    if any(k in t for k in feedback_kw):
        return "Feedback"
    return "Others"

if __name__ == "__main__":
    email = "Invoice amount does not match my usage."
    print(classify_email_v3(email))
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3-2\
s\SNEHITH\.vscode\extensions\ms-python.debugpy
y'
Billing
PS D:\3-2\AI Assitant Coding>
```

Problem-2

Task-1:

Prompt: Classify the following travel query into one of the categories:
Flight Booking, Hotel Booking, Cancellation, General Travel Info.

Query: *“Cancel my flight scheduled for next Monday.”*

Give a structured python code

Code:

```python
def classify_travel_query(query: str) -> str:
    q = query.lower()
    cancellation_kw = {"cancel", "cancellation", "cancelled", "terminate", "refund", "void", "annul"}
    flight_kw = {"flight", "airline", "boarding", "ticket", "flight booking", "seat", "departure", "arrival"}
    hotel_kw = {"hotel", "room", "reservation", "check in", "check-in", "check out", "checkout", "stay"}
    general_kw = {"visa", "baggage", "luggage", "weather", "transportation", "airport", "policy", "travel info", "information"}

    if any(k in q for k in cancellation_kw):
        return "Cancellation"
    if any(k in q for k in flight_kw):
        return "Flight Booking"
    if any(k in q for k in hotel_kw):
        return "Hotel Booking"
    if any(k in q for k in general_kw):
        return "General Travel Info"
    return "General Travel Info"

if __name__ == "__main__":
    query = "Cancel my flight scheduled for next Monday."
    print(classify_travel_query(query))
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3-2\
s\SNEHITH\.vscode\extensions\ms-python.debugpy
y'
Cancellation
PS D:\3-2\AI Assitant Coding>
```

Task-2:

Prompt: Classify the following travel query into one of the categories:

Flight Booking, Hotel Booking, Cancellation, General Travel Info.

Example:

Query: “I want to book a flight from Delhi to Mumbai.”

Category: Flight Booking

Now classify:

Query: “Book a hotel near the airport in Chennai.”

Give a structured python code

Code:

```python
def classify_travel(q: str) -> str:
    s = q.lower()
    hotel_terms = ["hotel", "room", "reservation", "stay", "book a hotel", "book hotel"]
    flight_terms = ["flight", "airline", "ticket", "boarding", "departure", "arrival"]
    cancel_terms = ["cancel", "cancellation", "refund", "terminate", "cancelled"]

    if any(t in s for t in hotel_terms):
        return "Hotel Booking"
    if any(t in s for t in flight_terms):
        return "Flight Booking"
    if any(t in s for t in cancel_terms):
        return "Cancellation"
    return "General Travel Info"

if __name__ == "__main__":
    query = "Book a hotel near the airport in Chennai."
    print(classify_travel(query))  # Expected output: Hotel Booking
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3-2\AI Assit
s\SNEHITH\.vscode\extensions\ms-python.debugpy-2025.18
y'
Hotel Booking
PS D:\3-2\AI Assitant Coding>
```

Task-3:

Prompt: Classify the following travel query into one of the categories:

Flight Booking, Hotel Booking, Cancellation, General Travel Info.

Examples:

Query: “Find me a flight to Bangalore.” → Flight Booking

Query: “Book a 5-star hotel in Goa.” → Hotel Booking

Query: “Cancel my hotel reservation.” → Cancellation

Query: “Do I need a visa for Singapore?” → General Travel Info

Now classify:

Query: “What is the best season to visit Manali?”

Give a structured python code

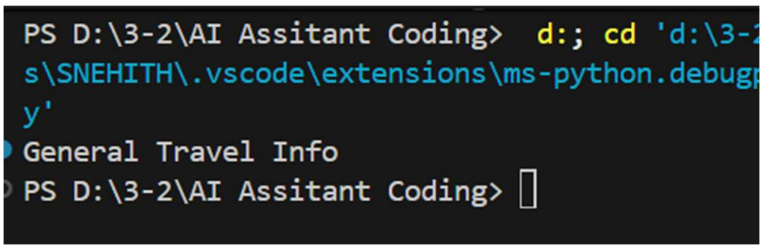
Code:

```python
def classify_travel_v4(query: str) -> str:
    q = query.lower()
    flight_keywords = ["flight", "airline", "ticket", "fly", "airfare"]
    hotel_keywords = ["hotel", "room", "reservation", "stay", "check-in", "checkout"]
    cancel_keywords = ["cancel", "cancellation", "refund", "terminate", "cancelled"]
    info_keywords = ["visa", "baggage", "luggage", "weather", "season", "best season", "when to visit", "climate", "travel info"]

    if any(k in q for k in cancel_keywords):
        return "Cancellation"
    if any(k in q for k in flight_keywords):
        return "Flight Booking"
    if any(k in q for k in hotel_keywords):
        return "Hotel Booking"
    if any(k in q for k in info_keywords):
        return "General Travel Info"
    return "General Travel Info"

if __name__ == "__main__":
    query = "What is the best season to visit Manali?"
    print(classify_travel_v4(query))
```

Output:



Problem-3:

Task-1:

Prompt: Classify the following programming question into one of the categories:

Syntax Error, Logic Error, Optimization, Conceptual Question.

Query: “My program runs but gives the wrong output.”

Give a structured Python code

Code:

```python
def classify_question(question: str) -> str:
    q = question.strip().lower()
    # A program that runs but produces incorrect results indicates a logic error
    if "runs" in q and ("wrong output" in q or "incorrect" in q or "gives the wrong output" in q):
        return "Logic Error"
    # fallback: classify common cases
    if "syntax" in q or "syntax error" in q:
        return "Syntax Error"
    if "slow" in q or "optimi" in q:
        return "Optimization"
    return "Conceptual Question"

if __name__ == "__main__":
    query = "My program runs but gives the wrong output."
    print(classify_question(query))
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd '
s\SNEHITH\.vscode\extensions\ms-python.
y'
Logic Error
PS D:\3-2\AI Assitant Coding>
```

Task-2:

Prompt: Classify the following programming question into one of the categories:

Syntax Error, Logic Error, Optimization, Conceptual Question.

Example:

Query: “I get a SyntaxError: unexpected indent.”

Category: Syntax Error

Now classify:

Query: “This code works but is very slow for large inputs.”

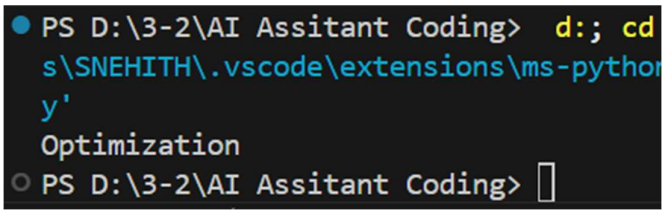
Give a structured Python code

Code:

```python
def classify_programming_question(question: str) -> str:
    q = question.lower()
    # Syntax error indicators
    if "syntax error" in q or "syntaxerror" in q or "unexpected indent" in q or "indentationerror" in q:
        return "Syntax Error"
    # Logic error indicators
    if ("runs" in q or "runs but" in q) and ("wrong output" in q or "incorrect" in q or "gives the wrong output" in q):
        return "Logic Error"
    # Optimization / performance indicators
    perf_keywords = ["slow", "time limit", "tle", "time complexity", "complexity", "optimi", "optimize", "performance", "large inputs", "la
    if any(k in q for k in perf_keywords):
        return "Optimization"
    # Fallback
    return "Conceptual Question"

if __name__ == "__main__":
    query = "This code works but is very slow for large inputs."
    print(classify_programming_question(query))  # Expected output: Optimization
```

Output:



Task-3:

Prompt: Classify the following programming question into one of the categories: Syntax Error, Logic Error, Optimization, Conceptual Question.

Examples:

1. **Query: *"I missed a colon in my if statement."* → Syntax Error**
2. **Query: *"The output is incorrect even though there are no errors."* → Logic Error**
3. **Query: *"How can I make this algorithm faster?"* → Optimization**
4. **Query: *"What is recursion in programming?"* → Conceptual Question**

Now classify:
Query: *"My loop never stops executing."*

Give me a Structured Python code

Code:

```python
def classify_inquiry(question: str) -> str:
    q = question.lower().strip()
    # Syntax indicators
    syntax_tokens = ["syntax error", "syntaxerror", "unexpected indent", "indentationerror", "missing colon", "missing parenthesis"]
    if any(tok in q for tok in syntax_tokens):
        return "Syntax Error"
    # Infinite-loop / non-terminating indicators -> logic error
    infinite_tokens = ["never stops", "infinite loop", "endless loop", "loop never stops", "doesn't terminate", "does not terminate", "loops forever", "stuck in a loop",
    if any(tok in q for tok in infinite_tokens):
        return "Logic Error"
    # Performance / optimization indicators
    perf_tokens = ["slow", "time limit", "tle", "time complexity", "optimi", "optimize", "performance"]
    if any(tok in q for tok in perf_tokens):
        return "Optimization"
    # Fallback
    return "Conceptual Question"

if __name__ == "__main__":
    query = "My loop never stops executing."
    print(classify_inquiry(query))  # Expected output: Logic Error
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3
.debugpy-2025.18.0-win32-x64\bundled\libs\d
Logic Error
PS D:\3-2\AI Assitant Coding>
```

Problem-4:

Task-1:

Prompt: Classify the following social media post into one of the categories:

Promotion, Complaint, Appreciation, Inquiry.

Post: "Worst service ever, my order arrived damaged."

Give a structured python code

Code:

```python
def classify_social_post(post: str) -> str:
    t = post.lower()
    complaint_kw = {"worst", "disappointed", "damaged", "broken", "not received", "late", "delay", "refund", "missing", "arrived damaged", "bad"}
    promotion_kw = {"sale", "discount", "buy now", "promo", "offer", "deal", "limited time", "subscribe", "order now"}
    appreciation_kw = {"love", "great", "thanks", "thank you", "awesome", "amazing", "appreciate"}
    inquiry_kw = {"how", "where", "when", "help", "do i", "does", "can i", "what", "why", "?"}

    if any(k in t for k in complaint_kw):
        return "Complaint"
    if any(k in t for k in promotion_kw):
        return "Promotion"
    if any(k in t for k in appreciation_kw):
        return "Appreciation"
    if "?" in t or any(k in t for k in inquiry_kw):
        return "Inquiry"
    return "Complaint"

if __name__ == "__main__":
    post = "Worst service ever, my order arrived damaged."
    print(classify_social_post(post))
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3-2\AI Assitant Coding'
.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '6512
Complaint
PS D:\3-2\AI Assitant Coding>
```

Task-2:

Prompt: Classify the following social media post into one of the categories:

Promotion, Complaint, Appreciation, Inquiry.

Example:

Post: "Limited-time offer on premium plans!"

Category: Promotion

Now classify:

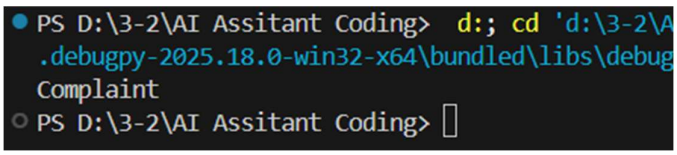
Post: “Still waiting for my refund after two weeks.”

Give a structured python code

Code:

```python
def classify_social_post(post: str) -> str:
    text = post.lower()
    complaint_kw = {"refund", "not received", "late", "waiting", "delay", "missing", "damaged", "broken", "worst", "disappointed"}
    promotion_kw = {"sale", "discount", "offer", "promo", "deal", "limited time", "buy now", "subscribe", "free"}
    appreciation_kw = {"love", "great", "thanks", "thank you", "awesome", "amazing", "appreciate", "grateful"}
    inquiry_kw = {"how", "where", "when", "help", "can i", "what", "why", "?"}
    if any(k in text for k in complaint_kw):
        return "Complaint"
    if any(k in text for k in promotion_kw):
        return "Promotion"
    if any(k in text for k in appreciation_kw):
        return "Appreciation"
    if "?" in text or any(k in text for k in inquiry_kw):
        return "Inquiry"
    return "Complaint"
if __name__ == "__main__":
    post = "Still waiting for my refund after two weeks."
    print(classify_social_post(post))  # Expected output: Complaint
```

Output:



Task-3:

Prompt: Classify the following social media post into one of the categories:

Promotion, Complaint, Appreciation, Inquiry.

Examples:

Post: “Flat 50% off on all products!” → Promotion

Post: “My package arrived broken and late.” → Complaint

Post: “Great service, really impressed!” → Appreciation

Post: “When will this be back in stock?” → Inquiry

Now classify:

Post: “Thanks a lot for resolving my issue so fast ”

Give a structured python code

Code:

```python
def classify_post_label(post: str) -> str:
    s = post.strip().lower()
    appreciation = {"thanks", "thank you", "thanks a lot", "appreciate", "great", "awesome", "amazing", "grateful"}
    complaint = {"broken", "damaged", "late", "not received", "refund", "disappointed", "worst", "missing"}
    promotion = {"sale", "discount", "offer", "promo", "flat", "off", "buy now", "free", "limited time"}
    inquiry_starters = {"when", "where", "how", "what", "why"}
    if any(kw in s for kw in appreciation):
        return "Appreciation"
    if any(kw in s for kw in complaint):
        return "Complaint"
    if any(kw in s for kw in promotion):
        return "Promotion"
    if "?" in s or any(s.startswith(w + " ") or (" " + w + " " in s) for w in inquiry_starters):
        return "Inquiry"
    return "Appreciation"

if __name__ == "__main__":
    post = "Thanks a lot for resolving my issue so fast "
    print(classify_post_label(post))  # Expected output: Appreciation
```

Output:

```
PS D:\3-2\AI Assitant Coding>  d:; cd 'd:\3-2
.debugpy-2025.18.0-win32-x64\bundled\libs\deb
Appreciation
PS D:\3-2\AI Assitant Coding>
```