LAB 3

AI-Generated Code (Initial)

```
def calculator(a, b, op):
    if op == '+':
        return a + b
    elif op == '-':
        return a - b
    elif op == '*':
        return a * b
    elif op == '/':
        return a / b
```

Refined Prompt (With Comments)

AI-Generated Code (Improved)

```
def calculator(a, b, operator):
    """Performs basic arithmetic operations"""
    if operator == '+':
        return a + b
    elif operator == '-':
        return a - b
    elif operator == '*':
        return a * b
    elif operator == '/':
        if b != 0:
            return a / b
        else:
            return "Division by zero error"
    else:
        return "Invalid operator"
```

Final Prompt (With Usage Example)

Create a Python calculator function.

Include error handling and an example of usage.

Final AI Output

```
def calculator(a, b, operator):
    """Calculator function with error handling"""
    try:
        if operator == '+':
            return a + b
        elif operator == '-':
            return a - b
        elif operator == '*':
            return a * b
        elif operator == '/':
            return a / b
        else:
            return "Invalid operator"
    except ZeroDivisionError:
        return "Cannot divide by zero"

# Example
print(calculator(10, 5, '+'))
```

Task Description 2: Refining Prompts for Sorting Logic

Vague Prompt

Sort student marks.

Initial AI Output

```
def sort_marks(marks):
    return sorted(marks)
```

Refined Prompt

Sort student marks in descending order.

Do not modify the original list.

Improved AI Output

```
def sort_marks(marks):
```

```
    return sorted(marks, reverse=True)
```

Expected Output

* Sorting logic becomes precise and efficient.

Task Description 3: Few-Shot Prompting for Prime Number Validation

Prompt with Examples

Check if a number is prime.

Examples:

Input: 2 → Output: Prime

Input: 4 → Output: Not Prime

Input: 13 → Output: Prime

AI-Generated Code

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
```

Expected Output

* Correct handling of edge cases (0, 1, negative numbers).

Task Description 4: Prompt-Guided UI Design for Student Grading System

Prompt

Create a Python UI that takes student marks,

calculates total, percentage, and grade.

AI-Generated Code

```
name = input("Enter student name: ")
m1 = int(input("Enter marks 1: "))
m2 = int(input("Enter marks 2: "))
m3 = int(input("Enter marks 3: "))


total = m1 + m2 + m3
```

```
percentage = total / 3

if percentage >= 75:
    grade = 'A'
elif percentage >= 60:
    grade = 'B'
elif percentage >= 50:
    grade = 'C'
else:
    grade = 'Fail'

print("Name:", name)
print("Total:", total)
print("Percentage:", percentage)
print("Grade:", grade)
```

Expected Output

* Well-structured UI code with clear output.

Task Description 5: Analyzing Prompt Specificity in Unit Conversion Functions

Basic Prompt

Convert kilometers to miles.

Improved Prompt

Create a Python function to convert kilometers to miles
and miles to kilometers with accurate formulas.

AI-Generated Code

```
def km_to_miles(km):
    return km * 0.621371

def miles_to_km(miles):
    return miles / 0.621371
```

Analysis

* Clear instructions result in accurate formulas.

* Specific prompts reduce ambiguity and errors.

Output:

Calculator examples:

10 + 5 = 15

10 / 0 = Error: Division by zero

10 ? 3 = Error: Invalid operator

Sort marks:

Original: [72, 88, 54, 91]

Sorted (desc): [91, 88, 72, 54]

Original unchanged: [72, 88, 54, 91]

Prime checks:

0: Not prime

1: Not prime

2: Prime

3: Prime

4: Not prime

18: Not prime

19: Prime

Compute grade:

Marks: [80, 75, 90]

18: Not prime

19: Prime

Compute grade:

Marks: [80, 75, 90]

19: Prime

Compute grade:

Marks: [80, 75, 90]

Compute grade:

Marks: [80, 75, 90]

Marks: [80, 75, 90]

{'total': 245, 'percentage': 81.66666666666667, 'grade': 'A'}

Unit conversions:

10 km -> miles: 6.21371

6.21371 miles -> km: 10.0

C:\Users\kunda>

{'total': 245, 'percentage': 81.66666666666667, 'grade': 'A'}

Unit conversions:

{'total': 245, 'percentage': 81.66666666666667, 'grade': 'A'}

{'total': 245, 'percentage': 81.66666666666667, 'grade': 'A'}

Unit conversions:

10 km -> miles: 6.21371

6.21371 miles -> km: 10.0