

ASSIGNMENT-3.1

2303A510E3

BATCH-14

Task 1: Zero-Shot Prompting (Palindrome Number Program)

Prompt: write a python function to check given number is palindrome or not return output int True or False. Provide code in user input.

Code with output:

```
❸ zeroshot(palindrome).py > ...
1  #write a python function to check given number is palindrome or not return output in True or False.provide code in user input
2  def is_palindrome(number):
3      # Convert the number to string
4      str_num = str(number)
5      # Check if the string is equal to its reverse
6      return str_num == str_num[::-1]
7  # Get user input
8  user_input = int(input("Enter a number: "))
9  # Check if the number is palindrome and print the result
10 result = is_palindrome(user_input)
11 print(result)

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 5
120
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 9
362880
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 4
24
```

Justification:

- Correct for positive numbers
- Negative numbers not handled clearly
- No input validation for non-integers

Task 2: One-Shot Prompting (Factorial Calculation)

Prompt: write a python function to calculate factorial of given number. Example: input=5 ; output=120

Code with output:

```
❸ oneshot(factorial).py > ...
1  #write a python function to calculate factorial of given number. Example: input=5 ; output=120
2  def factorial(n):
3      # Initialize result
4      result = 1
5      # Calculate factorial
6      for i in range(1, n + 1):
7          result *= i
8      return result
9  # Get user input
10 user_input = int(input("Enter a number: "))
11 # Calculate factorial and print the result
12 result = factorial(user_input)
13 print(result)
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 5
120
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 9
362880
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/oneshot(factorial).py"
Enter a number: 4
24
PS C:\Users\akhil\OneDrive\Documents\AI(vs)>
```

Justification:

- One-shot gives clearer logic
- More readable and reliable

Task 3: Few-Shot Prompting (Armstrong Number Check)

Code with output:

```
fewshot(armstrong).py > ...
1 # Write a python function to check given number is an Armstrong number.
2 # Example: input=153 ; output=Armstrong Number , input=123 ; output=Not an Armstrong Number
3
4 def is_armstrong_number(number):
5     # Convert the number to string
6     str_num = str(number)
7     # Calculate the number of digits
8     num_digits = len(str_num)
9     # Calculate the sum of each digit raised to the power of the number of digits
10    sum_of_powers = sum(int(digit) ** num_digits for digit in str_num)
11    # Check if the sum equals the original number
12    return sum_of_powers == number
13
14 # Get user input
15 user_input = int(input("Enter a number: "))
16
17 # Check if the number is an Armstrong number and print the result
18 result = is_armstrong_number(user_input)
19 if result:
20     print("Armstrong Number")
21 else:
22     print("Not an Armstrong Number")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot(armstrong).py"
Enter a number: 123
Not an Armstrong Number
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot(armstrong).py"
Enter a number: 153
Armstrong Number
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot(armstrong).py"
Enter a number: 370
Armstrong Number
```

Justification:

- Few-shot improves accuracy
- Code structure matches examples
- Still lacks input validation for negatives

Task 4: Context-Managed Prompting (Optimized Number Classification)

Prompt: generate an optimized python program that classifies a number is prime, composite or neither Code with output:

```
❸ contextmanaged(optimizednumberclassification.py) > ...
 1  #generate an optimized Python program that classifies a number is prime , composite or neither.
 2  def classify_number(number):
 3      if number <= 1:
 4          return "Neither Prime nor Composite"
 5      elif number == 2:
 6          return "Prime"
 7      elif number % 2 == 0:
 8          return "Composite"
 9      else:
10          for i in range(3, int(number**0.5) + 1, 2):
11              if number % i == 0:
12                  return "Composite"
13          return "Prime"
14  # Get user input
15  user_input = int(input("Enter a number: "))
16  # Classify the number and print the result
17  result = classify_number(user_input)
18  print(result)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/contextmanaged(optimizednumberclassification).py"
Enter a number: 5
Prime
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/contextmanaged(optimizednumberclassification).py"
Enter a number: 6
Composite
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/contextmanaged(optimizednumberclassification).py"
Enter a number: -5
Neither Prime nor Composite

Justification:

- Best performance among all prompts
- Clear validation and optimized logic
- Context improves efficiency and correctness

Task 5: Zero-Shot Prompting (Perfect Number Check)

Prompt:generate a python function to check given number is a perfect number.

Code with output:

```

❷ zeroshot(perfectnumber).py > ...
 1  #generate a python function to check given number is a perfect number
 2
 3  def is_perfect_number(number):
 4      if number < 1:
 5          return False
 6      # Calculate the sum of divisors
 7      divisors_sum = sum(i for i in range(1, number) if number % i == 0)
 8      # A perfect number is equal to the sum of its proper divisors
 9      return divisors_sum == number
10
11 # Get user input
12 user_input = int(input("Enter a number: "))
13 # Check if the number is a perfect number and print the result
14 result = is_perfect_number(user_input)
15 print(result)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

```

PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/zerohot/perfectnumber.py"
Enter a number: 22
False
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/zerohot/perfectnumber.py"
Enter a number: 6
True
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/zerohot/perfectnumber.py"
Enter a number: -3
False

```

Task 6: Few-Shot Prompting (Even or Odd Classification with Validation)

Code with output:

```

❷ fewshot(even or odd).py > ...
 1  #generate a Python program that checks if a number is even or odd.
 2  #example: input=4 ; output=Even
 3  #example: input=7 ; output=Odd
 4  #input=24 ; output=Even
 5  def check_even_odd(number):
 6      if number % 2 == 0:
 7          return "Even"
 8      else:
 9          return "Odd"
10
11 # Get user input
12 user_input = int(input("Enter a number: "))
13 # Check if the number is even or odd and print the result
14 result = check_even_odd(user_input)
15 print(result)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot/even or odd.py"
Enter a number: 22
Even
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot/even or odd.py"
Enter a number: 447
Odd
PS C:\Users\akhil\OneDrive\Documents\AI(vs)> & C:/Users/akhil/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/akhil/OneDrive/Documents/AI(vs)/fewshot/even or odd.py"
Enter a number: -55
Odd

```