SR UNIVERSITY

School of Computer Science and Artificial Intelligence

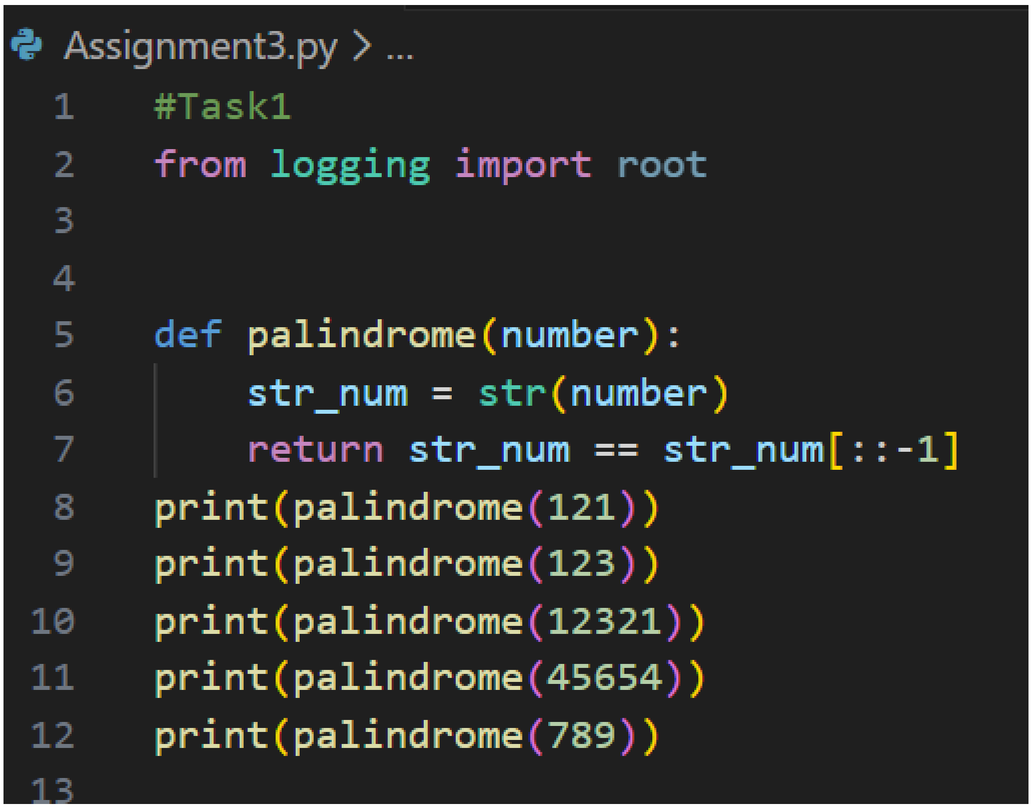
Lab Assignment # 3

Name of Student : B. Sai Rithvik
Enrollment No. : 2303A51536
Batch No. : 22

Question 1: Zero-Shot Prompting (Palindrome Number)

Prompt:
Write a Python function that takes an integer as input and returns True if the number is a palindrome and False otherwise. The function should not print anything and should not include any example inputs or outputs.

Code-

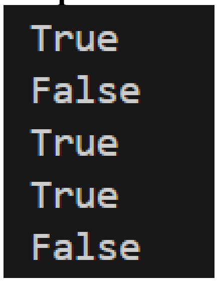


```
Assignment3.py > ...
#Task1
from logging import root


def palindrome(number):
    str_num = str(number)
    return str_num == str_num[::-1]
print(palindrome(121))
print(palindrome(123))
print(palindrome(12321))
print(palindrome(45654))
print(palindrome(789))

```

Output-

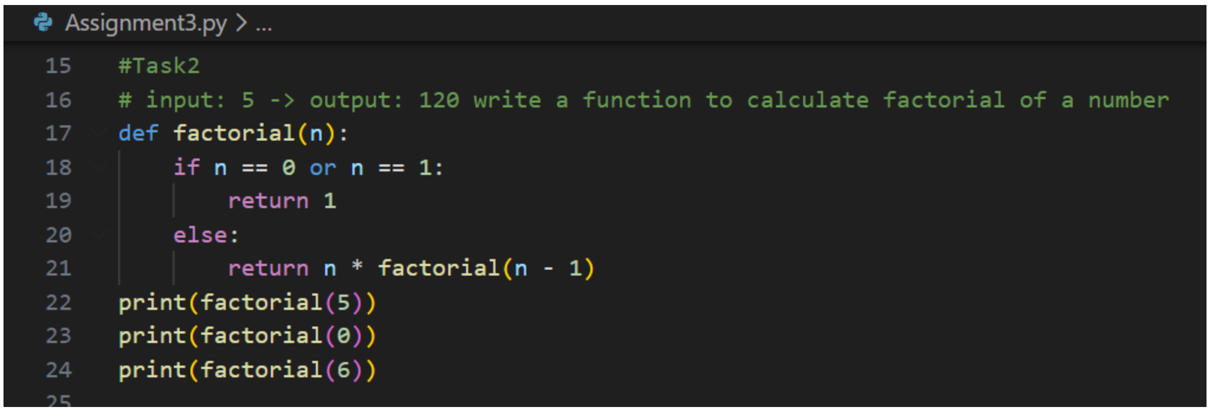


```
True
False
True
True
False
```

Question 2: One-Shot Prompting (Factorial Calculation)

Prompt-
Input: 5
Output: 120
Using the above example as a reference, write a Python function that calculates the factorial of a given non-negative integer.

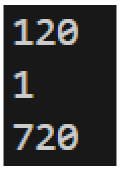
Code-



Assignment3.py > ...

```python
#Task2
# input: 5 -> output: 120 write a function to calculate factorial of a number
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
print(factorial(5))
print(factorial(0))
print(factorial(6))
```

Output-



```
120
1
720
```

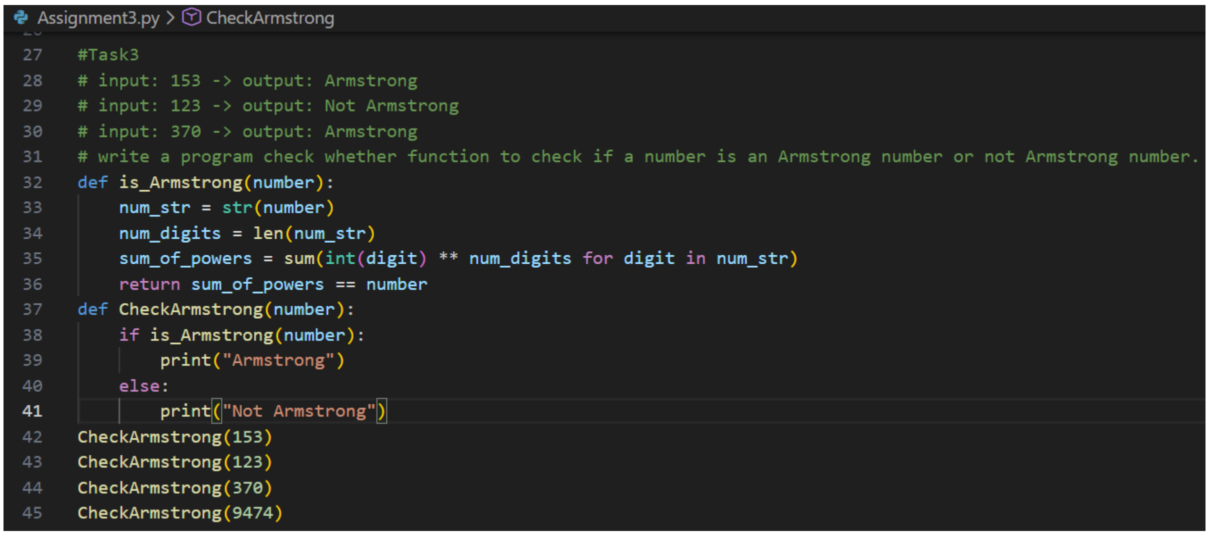
SR UNIVERSITY

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

Prompt-
Input: 153 → Output: Armstrong Number
Input: 370 → Output: Armstrong Number
Input: 123 → Output: Not an Armstrong Number

Based on the above examples, write a Python function that checks whether a given integer is an Armstrong number. The function should return an appropriate message.

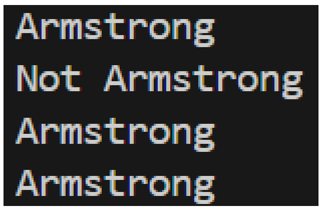
Code-



Assignment3.py > CheckArmstrong

```python
#Task3
# input: 153 -> output: Armstrong
# input: 123 -> output: Not Armstrong
# input: 370 -> output: Armstrong
# write a program check whether function to check if a number is an Armstrong number or not Armstrong number.
def is_Armstrong(number):
    num_str = str(number)
    num_digits = len(num_str)
    sum_of_powers = sum(int(digit) ** num_digits for digit in num_str)
    return sum_of_powers == number
def CheckArmstrong(number):
    if is_Armstrong(number):
        print("Armstrong")
    else:
        print("Not Armstrong")
CheckArmstrong(153)
CheckArmstrong(123)
CheckArmstrong(370)
CheckArmstrong(9474)
```

Output-



```
Armstrong
Not Armstrong
Armstrong
Armstrong
```

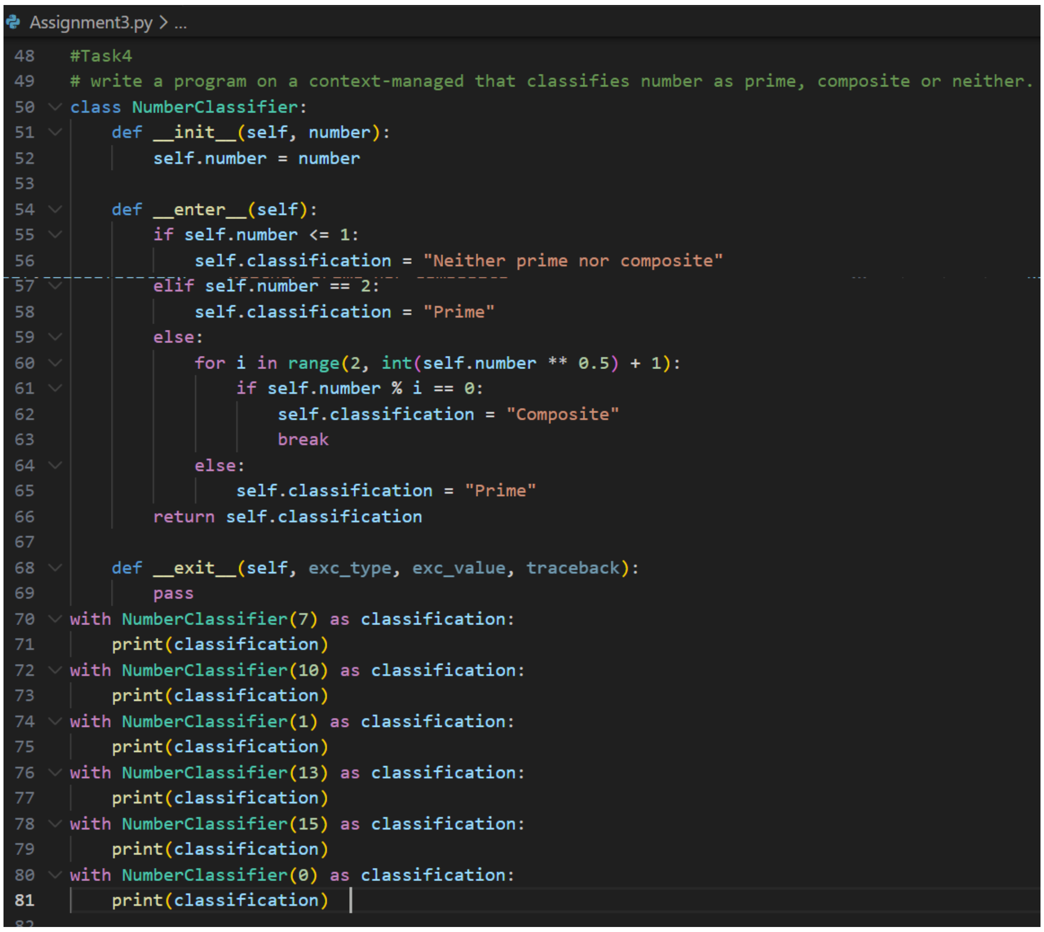
SR UNIVERSITY

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

Prompt-
Write an optimized Python program that classifies a given integer as **Prime**, **Composite**, or **Neither**.
Constraints and requirements:

- Validate the input to ensure it is an integer
- Handle edge cases such as 0, 1, and negative numbers
- Use an efficient algorithm to check for primality
- Display a clear and meaningful output

Code-



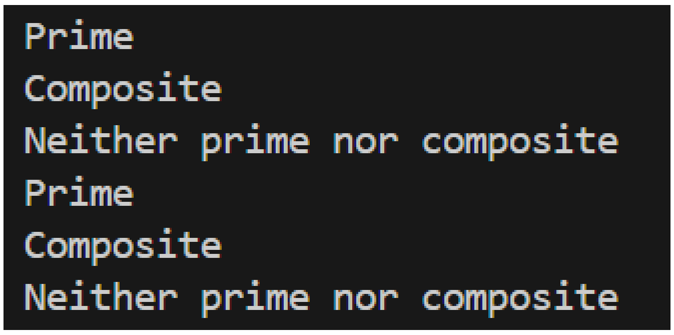
Assignment3.py > ...

```python
#Task4
# write a program on a context-managed that classifies number as prime, composite or neither.
class NumberClassifier:
    def __init__(self, number):
        self.number = number

    def __enter__(self):
        if self.number <= 1:
            self.classification = "Neither prime nor composite"
        elif self.number == 2:
            self.classification = "Prime"
        else:
            for i in range(2, int(self.number ** 0.5) + 1):
                if self.number % i == 0:
                    self.classification = "Composite"
                    break
            else:
                self.classification = "Prime"
        return self.classification

    def __exit__(self, exc_type, exc_value, traceback):
        pass
with NumberClassifier(7) as classification:
    print(classification)
with NumberClassifier(10) as classification:
    print(classification)
with NumberClassifier(1) as classification:
    print(classification)
with NumberClassifier(13) as classification:
    print(classification)
with NumberClassifier(15) as classification:
    print(classification)
with NumberClassifier(0) as classification:
    print(classification)
```

Output-

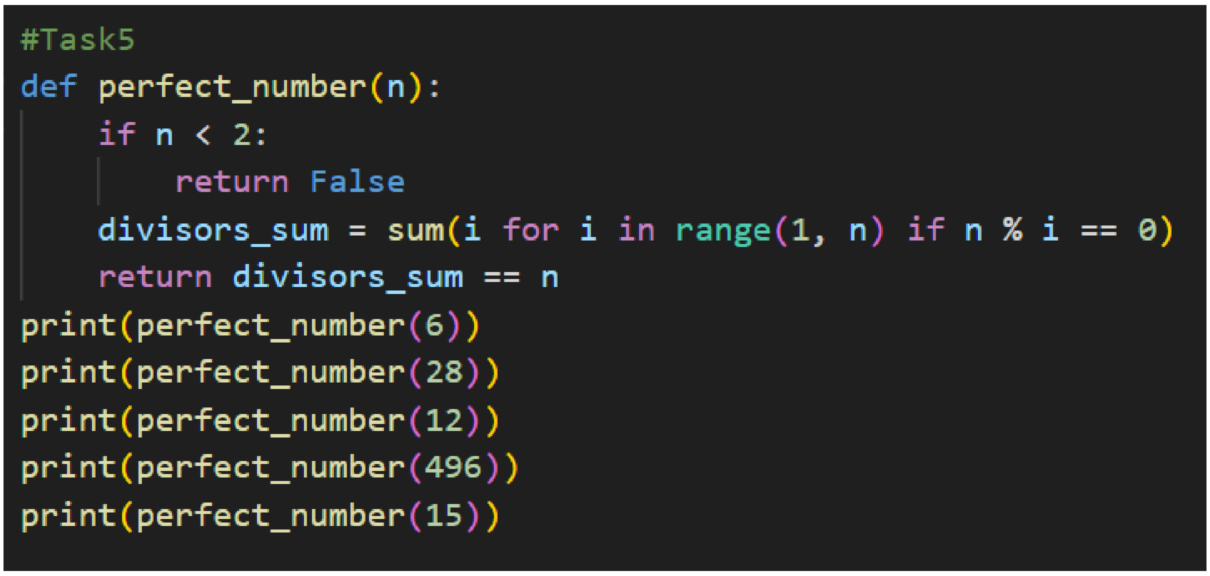


```
Prime
Composite
Neither prime nor composite
Prime
Composite
Neither prime nor composite
```

Question 5: Zero-Shot Prompting (Perfect Number)

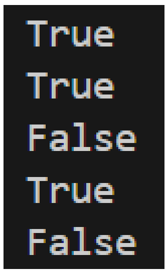
Prompt:
Write a Python function that determines whether a given positive integer is a perfect number. The function should return True if the number is perfect and False otherwise. Do not include any example inputs or outputs.

Code-



```python
#Task5
def perfect_number(n):
    if n < 2:
        return False
    divisors_sum = sum(i for i in range(1, n) if n % i == 0)
    return divisors_sum == n
print(perfect_number(6))
print(perfect_number(28))
print(perfect_number(12))
print(perfect_number(496))
print(perfect_number(15))
```

Output-

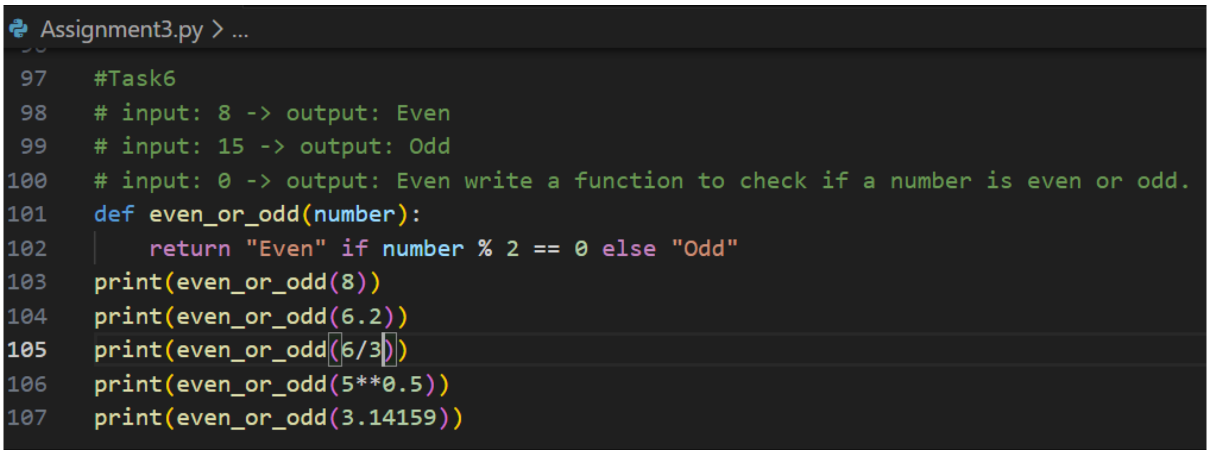


```
True
True
False
True
False
```

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

Prompt-
Input: 8 → Output: Even
Input: 15 → Output: Odd
Input: 0 → Output: Even
Using the above examples, write a Python function that checks whether a given number is even or odd. The function should include proper input validation and handle negative numbers and non-integer inputs gracefully.

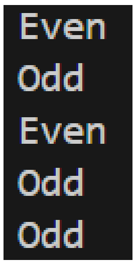
Code-



Assignment3.py > ...

```python
#Task6
# input: 8 -> output: Even
# input: 15 -> output: Odd
# input: 0 -> output: Even write a function to check if a number is even or odd.
def even_or_odd(number):
    return "Even" if number % 2 == 0 else "Odd"
print(even_or_odd(8))
print(even_or_odd(6.2))
print(even_or_odd(6/3))
print(even_or_odd(5**0.5))
print(even_or_odd(3.14159))
```

Output-



```
Even
Odd
Even
Odd
Odd
```