# Task 3 – Coding Assistance Challenge

## Objective :

Evaluate code generation and debugging ability of different Large Language Models (LLMs).

## Instructions / Prompt to be given :

Each LLM was given the same coding problem: “**Build a function that returns all prime numbers in a given range**”.  
  
The generated code was tested in an IDE and evaluated on:  
1. Whether it ran without errors  
2. Ease of fixing errors (if any)  
3. Code efficiency (readability and performance)  
4. Deliverable: Comparison chart + code snippets

## Comparison Table :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LLM Model | Did it Run Without Errors? | Ease of Fixing Errors | Readability | Performance | Code Snippet Provided |
| ChatGPT (GPT-5) | Yes | N/A (no error) | Very readable, clear comments | Optimized (√n loop) | ✔ |
| Claude | Minor indentation error | Easy to fix | Readable but verbose | Less optimized | ✔ |
| Gemini | Yes | N/A | Moderately readable | Optimized, less concise | ✔ |
| Llama 3 | No (logic bug) | Medium effort | Slightly confusing | Inefficient (no √n) | ✔ |

### Code Snippets

### ChatGPT (GPT-5)

def get\_primes\_in\_range(start, end):  
 primes = []  
 for num in range(start, end + 1):  
 if num > 1:  
 for i in range(2, int(num \*\* 0.5) + 1):  
 if num % i == 0:  
 break  
 else:  
 primes.append(num)  
 return primes  
print(get\_primes\_in\_range(10, 50))

### Claude

def primes\_in\_range(start, end):  
 result = []  
 for n in range(start, end+1):  
 if n > 1:  
 is\_prime = True  
 for i in range(2, n):  
 if n % i == 0:  
 is\_prime = False  
 break  
 if is\_prime:  
 result.append(n)  
 return result

### Gemini

def primes\_range(start, end):  
 return [n for n in range(start, end+1) if all(n % i for i in range(2, int(n\*\*0.5)+1)) and n > 1]

### Llama 3

def prime\_numbers(start, end):  
 primes = []  
 for n in range(start, end+1):  
 for i in range(2, n):  
 if n % i == 0:  
 break  
 else:  
 primes.append(n)  
 return primes