

## Assignment-3.2

**NAME:** v.snehitha

**ROLL NO:** 2303A52331

**COURSE:** AI Assisted Coding

**BATCH :** 39

---

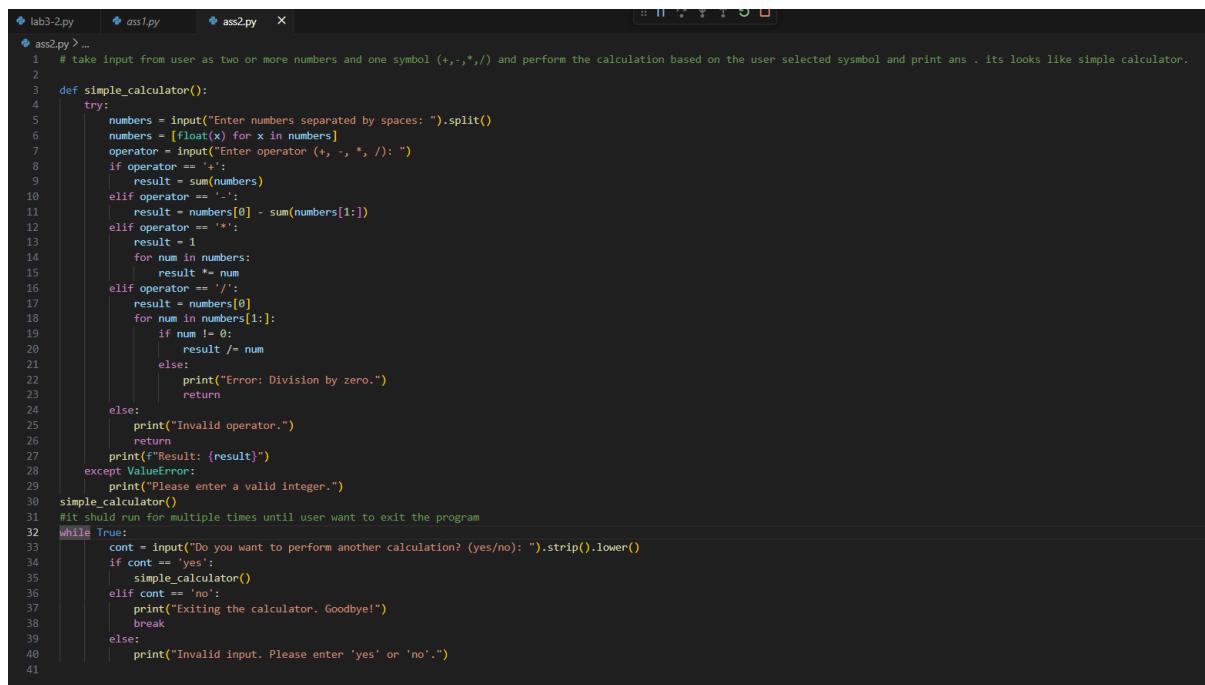
### Task Description-1

- Progressive Prompting for Calculator Design: Ask the AI to design a simple calculator program by initially providing only the function name. Gradually enhance the prompt by adding comments and usage examples.

### Expected Output-1

- Comparison showing improvement in AI-generated calculator logic and structure.

### PROMPT



```
# lab3-2.py | ass1.py | ass2.py | 
◆ ass2.py > ...
1 # take input from user as two or more numbers and one symbol (+,-,*,/) and perform the calculation based on the user selected sysmbol and print ans . its looks like simple calculator.
2
3 def simple_calculator():
4     try:
5         numbers = input("Enter numbers separated by spaces: ").split()
6         numbers = [float(x) for x in numbers]
7         operator = input("Enter operator (+, -, *, /): ")
8         if operator == '+':
9             result = sum(numbers)
10        elif operator == '-':
11            result = numbers[0] - sum(numbers[1:])
12        elif operator == '*':
13            result = 1
14            for num in numbers:
15                result *= num
16        elif operator == '/':
17            result = numbers[0]
18            for num in numbers[1:]:
19                if num != 0:
20                    result /= num
21                else:
22                    print("Error: Division by zero.")
23                    return
24            else:
25                print("Invalid operator.")
26            return
27        print("Result: {result}")
28    except ValueError:
29        print("Please enter a valid integer.")
30    simple_calculator()
31 #it shuld run for multiple times until user want to exit the program
32 while True:
33     cont = input("Do you want to perform another calculation? (yes/no): ").strip().lower()
34     if cont == 'yes':
35         simple_calculator()
36     elif cont == 'no':
37         print("Exiting the calculator. Goodbye!")
38         break
39     else:
40         print("Invalid input. Please enter 'yes' or 'no'.")
```

### OUTPUT

```

PS C:\Users\harsh\OneDrive\Documents\AIAC> c:;
cd 'c:\Users\harsh\OneDrive\Documents\AIAC'; & 'c:\office\miniforge\python.exe' debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '57396' '--' 'C:\Users\harsh\OneDrive\Documents\AIAC\ass2.py'
Enter numbers separated by spaces: 2 5 7 2 1
Enter operator (+, -, *, /): +
Result: 17.0
Enter operator (+, -, *, /): +
Enter operator (+, -, *, /): +
Result: 17.0
Do you want to perform another calculation? (yes/no): yes
Enter numbers separated by spaces: 45 2
Enter operator (+, -, *, /): *
Result: 90.0
Do you want to perform another calculation? (yes/no): 45 5 267 73
Invalid input. Please enter 'yes' or 'no'.
Do you want to perform another calculation? (yes/no): yes
Enter numbers separated by spaces: 563 673 7 8
Enter operator (+, -, *, /): *
Result: 17.0
Do you want to perform another calculation? (yes/no): yes
Enter numbers separated by spaces: 45 2
Enter operator (+, -, *, /): *
Result: 90.0
Do you want to perform another calculation? (yes/no): 45 5 267 73
Invalid input. Please enter 'yes' or 'no'.
Do you want to perform another calculation? (yes/no): yes
Enter numbers separated by spaces: 563 673 7 8
Enter operator (+, -, *, /): *
Result: 90.0
Do you want to perform another calculation? (yes/no): 45 5 267 73
Invalid input. Please enter 'yes' or 'no'.
Do you want to perform another calculation? (yes/no): yes
Enter numbers separated by spaces: 563 673 7 8
Enter operator (+, -, *, /): *
Enter operator (+, -, *, /): *
Result: 21218344.0
Result: 21218344.0
Result: 21218344.0
Do you want to perform another calculation? (yes/no):

```

## ANALYSIS

- The logic is better and basic operations are added.
- Still no error handling or input checking.
- It handles wrong inputs and division by zero.

## Task Description-2

- Refining Prompts for Sorting Logic: Start with a vague prompt for sorting student marks, then refine it to clearly specify sorting order and constraints.

## Expected Output-2

- AI-generated sorting function evolves from ambiguous logic to an accurate and efficient implementation.

## PROMPT

```

lab3.py ass1.py ass2.py
ass2.py > student_marks_memo

41     # take input from user student name and student roll number and shuld be print the student marks memo.
42     # in specific oder like name and roll number at the top and then marks obtaind in each marks subject wise and total grades and percentage.
43     # total subjects are 5 like maths ,science ,english ,hindi ,social studies.
44     # pass marks for each subject is 40 marks and each subject is of 100 marks.
45     # and programs shuld runs for multiple students until user want to exit the program.
46     # also it should handel invalid inputs.
47     # meantion weather he passed or failed accrding to each subject pass marks, if two subjects are failed it shuld show at butom
48     def student_marks_memo():
49         while True:
50             try:
51                 name = input("Enter student name: ")
52                 roll_number = input("Enter student roll number: ")
53                 subjects = ['Maths', 'Science', 'English', 'Hindi', 'Social Studies']
54                 marks = {}
55                 total_marks = 0
56                 passed_subjects = 0
57
58                 for subject in subjects:
59                     mark = int(input(f"Enter marks obtained in {subject} (out of 100): "))
60                     if mark < 0 or mark > 100:
61                         print("Invalid marks. Please enter marks between 0 and 100.")
62                         return
63                     marks[subject] = mark
64                     total_marks += mark
65                     if mark >= 40:
66                         passed_subjects += 1
67
68                     percentage = (total_marks / 500) * 100
69                     grade = 'A' if percentage >= 90 else 'B' if percentage >= 75 else 'C' if percentage >= 60 else 'D' if percentage >= 50 else 'F'
70
71                     print("\n--- Student Marks Memo ---")
72                     print(f"Name: {name}")
73                     print(f"Roll Number: {roll_number}")
74                     for subject, mark in marks.items():
75                         print(f"{subject}: {mark}")
76                     print(f"Total Marks: {total_marks}/500")
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93

```

```

student_marks_memo()
    print(f"Subject: {subject} ")
    print(f"Total Marks: {total_marks}/500")
    print(f"Percentage: {percentage:.2f}%")
    print(f"Grade: {grade}")

    if passed_subjects < len(subjects):
        failed_subjects = len(subjects) - passed_subjects
        print(f"Failed Subjects: {failed_subjects}")

    cont = input("Do you want to enter details for another student? (yes/no): ").strip().lower()
    if cont != 'yes':
        print("Exiting the program. Goodbye!")
        break
except ValueError:
    print("Invalid input. Please enter valid data.")

student_marks_memo()

```

## OUTPUT

```

PS C:\Users\harsh\OneDrive\Documents\AIAC>
PS C:\Users\harsh\OneDrive\Documents\AIAC> c;; cd 'c:\Users\harsh\OneDrive\Documents\AIAC'; & 'c:\office\miniforge\python.exe
2-x64\bundled\libs\debugpy\launcher' '57899' '--' 'C:\Users\harsh\OneDrive\Documents\AIAC\ass2.py'
Enter student name: harsha
Enter student roll number: 2303
Enter marks obtained in Maths (out of 100): 20
Enter marks obtained in Science (out of 100): 49
Enter marks obtained in English (out of 100): 58
Enter marks obtained in Hindi (out of 100): 39
Enter marks obtained in Social Studies (out of 100): 59

--- Student Marks Memo ---
Name: harsha
Roll Number: 2303
Maths: 20
Science: 49
English: 58
Hindi: 39
Social Studies: 59
Total Marks: 225/500
Percentage: 45.00%
Grade: F
Failed Subjects: 2
Do you want to enter details for another student? (yes/no): 

```

## ANALYSIS

- The first prompt was unclear, so the AI gave a basic sorting program.
- After refining the prompt, the sorting order became clear.
- Adding constraints helped the AI handle only valid student marks.
- Refined prompts result in accurate and reliable code.

## Task Description-3

- Few-Shot Prompting for Prime Number Validation: Provide multiple input-output examples for a function that checks whether a number is prime. Observe how few-shot prompting improves correctness.

## Expected Output-3

- Improved prime-checking function with better edge-case handling

## PROMPT

```
❸ ass2.py > ...
92
93 # take input from user as number and print whether the number is prime number or not
94 # output shud be like
95 #input :7 -> prime number
96 def is_prime_number():
97     num = int(input("Enter a number: "))
98
99     if num <= 1:
100         print(f"{num} is not a prime number.")
101         return
102
103     for i in range(2, int(num**0.5) + 1):
104         if num % i == 0:
105             print(f"{num} is not a prime number.")
106             return
107
108         print(f"{num} is a prime number.")
109 is_prime_number()
110 #program shud run for multiple times until user want to exit the program
111 while True:
112     cont = input("Do you want to check another number? (yes/no): ").strip().lower()
113     if cont == 'yes':
114         is_prime_number()
115     elif cont == 'no':
116         print("Exiting the program. Goodbye!")
117         break
118     else:
119         print("Invalid input. Please enter 'yes' or 'no'.")
120
121
```

## OUTPUT

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
-1 is not a prime number.
Do you want to check another number? (yes/no):
PS C:\Users\harsh\OneDrive\Documents\AIAC> ^C
PS C:\Users\harsh\OneDrive\Documents\AIAC>
PS C:\Users\harsh\OneDrive\Documents\AIAC> c;; cd 'c:\Users\harsh\OneDrive\Documents\AIAC'; & 'c:\office\miniforge\python.exe' '2-x64\bundled\libs\debugpy\launcher' '54084' '--' 'C:\Users\harsh\OneDrive\Documents\AIAC\ass2.py'
Enter a number: 2
2 is a prime number.
Do you want to check another number? (yes/no): yes
Enter a number: 4
4 is not a prime number.
Do you want to check another number? (yes/no): yes
Enter a number: -1
-1 is not a prime number.
Do you want to check another number? (yes/no): yes
Enter a number: 0
0 is not a prime number.
Do you want to check another number? (yes/no): █
```

## ANALYSIS

- The examples helped the AI understand prime and non-prime cases.
- The program now handles edge cases like 1 and negative numbers.
- Few-shot prompting improved correctness compared to zero-shot.
- The logic is efficient and reliable.

## Task Description-4

- Prompt-Guided UI Design for Student Grading System: Create a user interface for a student grading system that calculates total marks, percentage, and grade based on user input.

## Expected Output-4

- Well-structured UI code with accurate calculations and clear output display.

## PROMPT

```
lab3-2.py ass1.py ass2.py
ass2.py > ...

121 # create a UI design for student garding system using tkinter module in python.
122 # it shuld ask to enter student name ,roll number and marks obtained in 5 subjects like maths,science,english,hindi,social studies.
123 # in a box and button to submit the details.
124 # it shuld calculate total grade and print how many subjects are faild acording to pass marks for each subject 40
125 # and it shuld show the result in a message box when user click on submit button.
126 import tkinter as tk
127 from tkinter import messagebox
128 def calculate_grade():
129     try:
130         name = entry_name.get()
131         roll_number = entry_roll.get()
132         subjects = ['Maths', 'Science', 'English', 'Hindi', 'Social Studies']
133         marks = []
134         total_marks = 0
135         passed_subjects = 0
136
137         for entry in entries_marks:
138             mark = int(entry.get())
139             if mark < 0 or mark > 100:
140                 messagebox.showerror("Invalid Input", "Please enter marks between 0 and 100.")
141                 return
142             marks.append(mark)
143             total_marks += mark
144             if mark >= 40:
145                 passed_subjects += 1
146
147             percentage = (total_marks / 500) * 100
148             grade = 'A' if percentage >= 90 else 'B' if percentage >= 75 else 'C' if percentage >= 60 else 'D' if percentage >= 50 else 'F'
149
150             result_message = f"Name: {name}\nRoll Number: {roll_number}\nTotal Marks: {total_marks}/500\nPercentage: {percentage:.2f}%\nGrade: {grade}"
151
152             if passed_subjects < len(subjects):
153                 failed_subjects = len(subjects) - passed_subjects
154                 result_message += f"\nFailed Subjects: {failed_subjects}\n"
155
156     messagebox.showinfo("Student Grade", result_message)
```

```

ass2.py > ...
128 def calculate_grade():
129     result_message += f"Failed Subjects: {failed_subjects}\n"
130
131     messagebox.showinfo("Student Grade", result_message)
132 except ValueError:
133     messagebox.showerror("Invalid Input", "Please enter valid data.")
134
135 root = tk.Tk()
136 root.title("Student Grading System")
137 tk.Label(root, text="Student Name:").grid(row=0, column=0)
138 entry_name = tk.Entry(root)
139 entry_name.grid(row=0, column=1)
140 tk.Label(root, text="Roll Number:").grid(row=1, column=0)
141 entry_roll = tk.Entry(root)
142 entry_roll.grid(row=1, column=1)
143 subjects = ['Maths', 'Science', 'English', 'Hindi', 'Social Studies']
144 entries_marks = []
145 for i, subject in enumerate(subjects):
146     tk.Label(root, text=f"{subject} Marks:").grid(row=i+2, column=0)
147     entry = tk.Entry(root)
148     entry.grid(row=i+2, column=1)
149     entries_marks.append(entry)
150
151 tk.Button(root, text="Submit", command=calculate_grade).grid(row=len(subjects)+2, columnspan=2)
152
153 root.mainloop()
154

```

## OUTPUT

The screenshot shows the 'Student Grading System' application interface. On the left, there is a window titled 'Student Grading System' containing input fields for student details and marks. The fields include:

- Student Name: harsha
- Roll Number: 2302
- Maths Marks: 20
- Science Marks: 56
- English Marks: 46
- Hindi Marks: 78
- Social Studies Marks: 49

Below these fields is a 'Submit' button.

On the right, a separate message box titled 'Student Grade' displays the following information:

- Name: harsha
- Roll Number: 2302
- Total Marks: 249/500
- Percentage: 49.80%
- Grade: F
- Failed Subjects: 1

At the bottom of the message box is an 'OK' button.

## ANALYSIS

- The prompt clearly guides the AI to design a user interface for a student grading system.
- The UI allows users to enter marks and view total, percentage, and grade easily.
- Clear instructions help the AI generate accurate calculations and structured output.
- Prompt-guided UI design improves usability and correctness of the system.

### Task Description-5

- Analyzing Prompt Specificity in Unit Conversion Functions: Improving a Unit Conversion Function (Kilometers to Miles and Miles to Kilometers) Using Clear Instructions.

### Expected Output-5

- Analysis of code quality and accuracy differences across multiple prompt variations.

### PROMPT

```
❶ ass2.py > ...
1/6
177 # take input from user it shuld be a distance in kilometers and convert it into meters.
178 # and if meters gives then convert it into kilometers.
179 #no negative values should be accepted.
180 # program shuld run for multiple times until user want to exit the program.
181 def convert_distance():
182     while True:
183         try:
184             choice = input("Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): ")
185             if choice == '1':
186                 km = float(input("Enter distance in kilometers: "))
187                 if km < 0:
188                     print("Please enter a non-negative value.")
189                     continue
190                 meters = km * 1000
191                 print(f"{km} kilometers is equal to {meters} meters.")
192             elif choice == '2':
193                 meters = float(input("Enter distance in meters: "))
194                 if meters < 0:
195                     print("Please enter a non-negative value.")
196                     continue
197                 km = meters / 1000
198                 print(f"{meters} meters is equal to {km} kilometers.")
199             else:
200                 print("Invalid choice. Please enter 1 or 2.")
201                 continue
202
203             cont = input("Do you want to perform another conversion? (yes/no): ").strip().lower()
204             if cont != 'yes':
205                 print("Exiting the program. Goodbye!")
206                 break
207             except ValueError:
208                 print("Invalid input. Please enter a valid number.")
209         convert_distance()
```

### OUTPUT

```
PS C:\Users\harsh\OneDrive\Documents\AIAC> c;; cd 'c:\Users\harsh\OneDrive\Documents\AIAC'; & 'c:\office\miniforge\python.exe' 'c:\Users\harsh\vs2-x64\bundled\libs\debugpy\launcher' '55845' '--' 'C:\Users\harsh\OneDrive\Documents\AIAC\ass2.py'
Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): 2
Enter distance in meters: 467
467.0 meters is equal to 0.467 kilometers.
Do you want to perform another conversion? (yes/no): yes
Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): 1
Enter distance in kilometers: -584
Please enter a non-negative value.
Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): yes
Invalid choice. Please enter 1 or 2.
Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): 1
Enter distance in kilometers: 56
56.0 kilometers is equal to 56000.0 meters.
Do you want to perform another conversion? (yes/no): yes
Convert (1) Kilometers to Meters or (2) Meters to Kilometers? (enter 1 or 2): 2
Enter distance in meters: 567679
567679.0 meters is equal to 567.679 kilometers.
Do you want to perform another conversion? (yes/no):
```

## ANALYSIS

- Clear and specific prompts help the AI generate accurate unit conversion functions.
- When the prompt is vague, the code may miss correct formulas or input validation.
- Adding clear instructions improves code quality, accuracy, and readability.
- Prompt specificity ensures correct conversion between kilometres and miles.