

School of Computer Science and Artificial Intelligence

Lab Assignment # 1

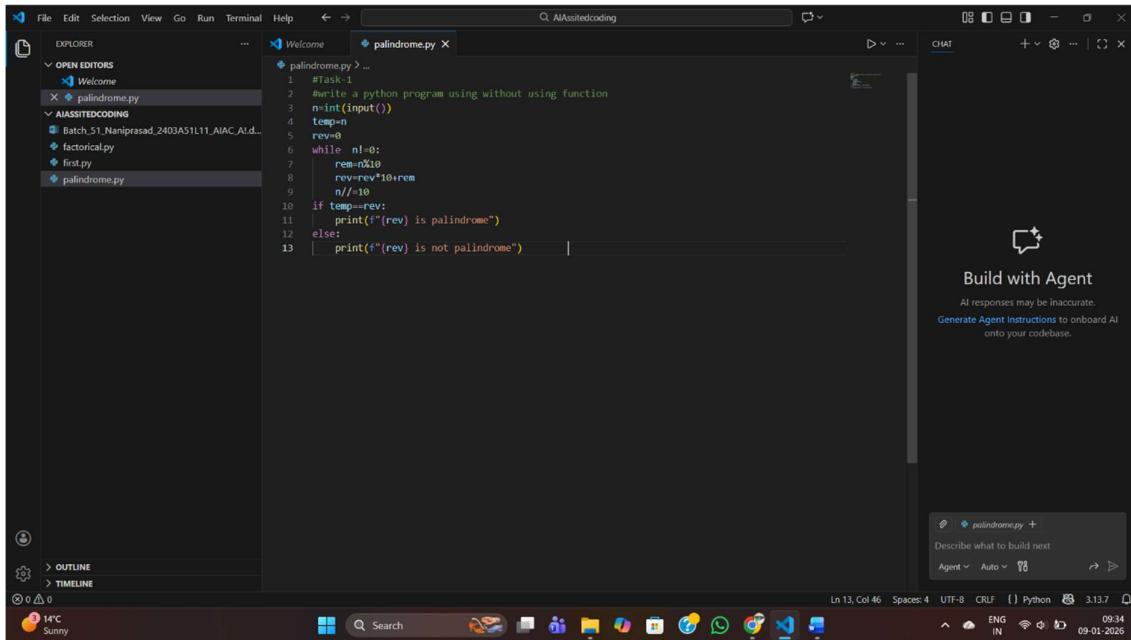
Program	: B. Tech (CSE)
Specialization	:
Course Title	: AI Assisted coding
Course Code	:
Semester	: II
Academic Session	: 2025-2026
Name of Student	: S. Bharath
Enrollment No.	: 2403A51L17
Batch No.	: 51
Date	: 06-01-2026

Submission Starts here

OUTPUT :
SCREENSHOTS:

#Task1

Write a python program for palindrome without using function



```
palindrome.py > ...
1 #Task 1
2 #write a python program using without using function
3 n=int(input())
4 temp=n
5 rev=0
6 while n!=0:
7     rem=n%10
8     rev=rev*10+rem
9     n/=10
10 if temp==rev:
11     print("rev) is palindrome")
12 else:
13     print("rev) is not palindrome")
```

Output:

```
PS C:\Users\nanip\OneDrive\Desktop\AIAssitedcoding> & c:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding\doodling/palindrome.py
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssitedcoding> & c:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding\doodling/palindrome.py
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssitedcoding>
```

Palindrome check steps for the given code

1. Read input:
 - o Take an integer from the user and store it in n.
2. Store original number:
 - o Copy n into temp so you can compare later after reversing.
3. Initialize reverse:
 - o Set rev = 0. This will be built digit by digit into the reversed number.
4. Loop until n becomes 0:
 - o Keep extracting the last digit and removing it from n using integer division.
5. Extract last digit:
 - o $\text{rem} = \text{n} \% 10$
 - o This gives the rightmost digit of n.
6. Append digit to reversed number:
 - o $\text{rev} = \text{rev} * 10 + \text{rem}$
 - o Shifts existing digits in rev left and adds the new last digit.
7. Remove last digit from n:
 - o $\text{n} //= 10$
 - o Drops the rightmost digit from n to process the next one.

8. End of loop:

- When n becomes 0, rev now holds the full reversed number.

9. Compare original with reversed:

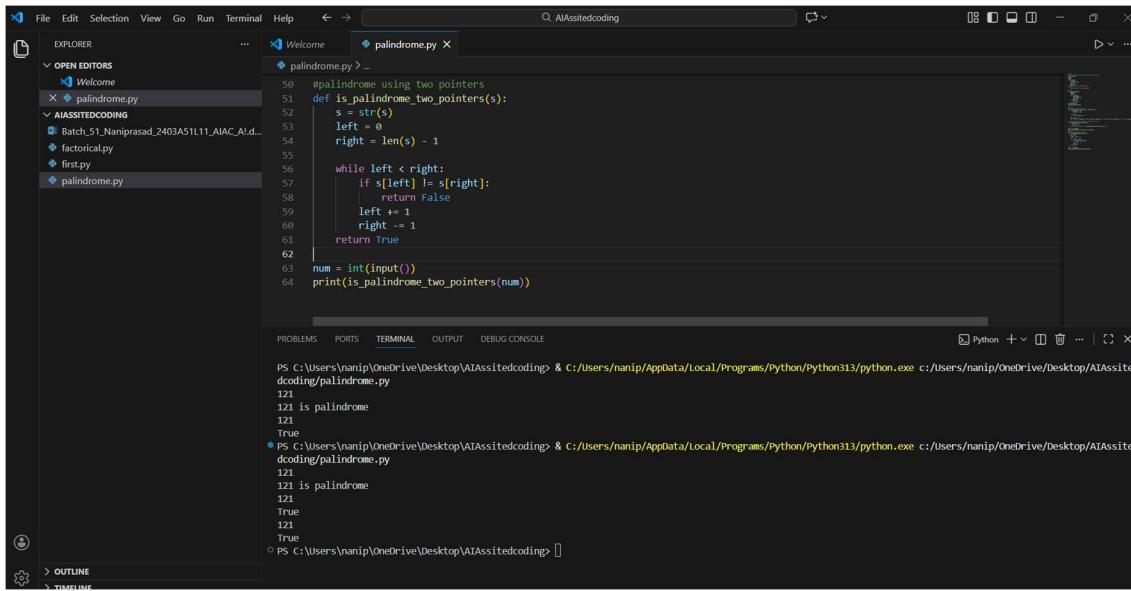
- If temp == rev, the original number reads the same backward → it's a palindrome.
- Otherwise, it's not a palindrome.

10. Output result:

- Print "rev is palindrome" if equal, else "rev is not palindrome".

#Task2:

Write optimal solution for palindrome solution



The screenshot shows a Visual Studio Code (VS Code) interface. The left sidebar has a tree view with 'OPEN EDITORS' expanded, showing files like 'Welcome', 'palindrome.py', and others under 'AIASSISTEDCODING'. The main editor area contains a Python script named 'palindrome.py' with the following code:

```
#palindrome using two pointers
def is_palindrome_two_pointers(s):
    s = str(s)
    left = 0
    right = len(s) - 1

    while left < right:
        if s[left] != s[right]:
            return False
        left += 1
        right -= 1
    return True

num = int(input())
print(is_palindrome_two_pointers(num))
```

Below the editor is a terminal window showing the execution of the script:

```
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & C:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding/palindrome.py
121
121 is palindrome
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & C:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding/palindrome.py
121
121 is palindrome
121
True
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding>
```

Output:

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows multiple files including `palindrome.py`, `factorial.py`, and `first.py`.
- Editor:** Displays the code for `palindrome.py` which uses two pointers to check if a string is a palindrome.
- Terminal:** Shows the output of running the script with the command `python palindrome.py`. The output indicates that the number 121 is a palindrome.
- Bottom Bar:** Includes tabs for PROBLEMS, PORTS, TERMINAL, OUTPUT, and DEBUG CONSOLE.

Explanation:

Create function

Pass the input with some value

In two pointer if last and first value are equal then

Last-=1

And first+=1

So if all index values are equal checking the last and first return True

If not return False

#Task 3

Write python program for palindrome using function

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows multiple files including `palindrome.py`, `factorial.py`, and `first.py`.
- Editor:** Displays the code for `palindrome.py` which defines a function to check if a number is a palindrome by reversing it and comparing it with the original.
- Terminal:** Shows the output of running the script with the command `python palindrome.py`. The output indicates that the number 121 is a palindrome.
- Bottom Bar:** Includes tabs for PROBLEMS, PORTS, TERMINAL, OUTPUT, and DEBUG CONSOLE.

Output:

The screenshot shows a Windows desktop environment with the Visual Studio Code (VS Code) application open. The code editor displays a Python file named 'palindrome.py' with the following content:

```
13     print(f'{rev} is not palindrome')
14
15 #Task2
16 def palindrome(num):
17     temp=num
18     rev=0
19     while num!=0:
20         rem=num%10
21         rev=rev*10+rem
22         num//=10
23     if temp==rev:
24         return True
25     return False
26 num=int(input())
27 print(palindrome(num))
```

The terminal tab at the bottom shows the output of running the script:

```
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & c:/users/nanip/appdata/local/programs/python/python313/python.exe c:/users/nanip/Desktop/AIAssistedcoding/palindrome.py
121
121 is palindrome
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding>
```

The status bar at the bottom right indicates the file is saved, the language is Python, and the current date and time are 09-01-2026.

Explanation:

Step-by-Step Explanation

1. Function Definition

- `def palindrome(num):`
- A function named `palindrome` is created that takes one argument `num`.

2. Store Original Number

- `temp = num`
- The original number is stored in `temp` so we can compare later.

3. Initialize Reverse

- `rev = 0`
- This variable will hold the reversed number.

4. Loop to Reverse Number

- `while num != 0:` → keep looping until `num` becomes 0.
- Inside the loop:

- $\text{rem} = \text{num} \% 10 \rightarrow$ extract the last digit.
- $\text{rev} = \text{rev} * 10 + \text{rem} \rightarrow$ build the reversed number digit by digit.
- $\text{num} //= 10 \rightarrow$ remove the last digit from num.

5. Check Palindrome

- After the loop ends, rev contains the reversed number.
- Compare temp (original number) with rev.
- If they are equal \rightarrow return True.
- Otherwise \rightarrow return False.

Main Program

- $\text{num} = \text{int}(\text{input}()) \rightarrow$ take user input.
- $\text{print}(\text{palindrome}(\text{num})) \rightarrow$ call the function and print the result (True or False).

Example Walkthrough

Suppose input is 121:

- $\text{temp} = 121$, $\text{rev} = 0$
- Loop:
 - Iteration 1: $\text{rem} = 1$, $\text{rev} = 1$, $\text{num} = 12$
 - Iteration 2: $\text{rem} = 2$, $\text{rev} = 12$, $\text{num} = 1$
 - Iteration 3: $\text{rem} = 1$, $\text{rev} = 121$, $\text{num} = 0$
- Loop ends $\rightarrow \text{rev} = 121$
- Compare: $\text{temp} == \text{rev} \rightarrow 121 == 121 \rightarrow \text{True}$
- Output: True

If input is 123:

- Reverse becomes 321
- Compare: $123 != 321 \rightarrow \text{False}$

- Output: False

#Task4:

Write Python program with using function and without using function

```

File Edit Selection View Go Run Terminal Help < - > Q AIAssistedcoding
EXPLORER OPEN EDITORS Welcome palindrome.py ...
1 #Task_1
2 write a python program using without using function
3 n=int(input())
4 temp=n
5 rev=0
6 while n!=0:
7     rem=n%10
8     rev=rev*10+rem
9     n/=10
10 if temp==rev:
11     print(f"{rev} is palindrome")
12 else:
13     print(f"{rev} is not palindrome")

```

Build with Agent
AI responses may be inaccurate.
Generate Agent instructions to onboard AI onto your codebase.

OUTLINE TIMELINE

14°C Sunny

Ln 13, Col 46 Spaces: 4 UTF-8 CRLF Python 3.13.7 09:54 IN 09-01-2025

```

File Edit Selection View Go Run Terminal Help < - > Q AIAssistedcoding
EXPLORER OPEN EDITORS Welcome palindrome.py ...
66 def is_palindrome_stack(s):
67     s = str(s)
68     stack = []
69     for char in s:
70         stack.append(char)
71
72     for char in s:
73         if char != stack.pop():
74             return False
75     return True
76
77 num = int(input())
78 print(is_palindrome_stack(num))

```

PROBLEMS PORTS TERMINAL OUTPUT DEBUG CONSOLE

PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & c:/users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/users/nanip/OneDrive/Desktop\coding/palindrome.py
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding>

Output:

Step-by-Step

1. **Input:** User enters a number → stored in n.
2. **Save original:** temp = n keeps the original number safe.

3. Reverse logic:

- Extract last digit using $\text{rem} = n \% 10$.
- Build reversed number: $\text{rev} = \text{rev} * 10 + \text{rem}$.
- Remove last digit: $n // 10$.
- Repeat until n becomes 0.

4. Compare: If $\text{temp} == \text{rev}$, the number is palindrome.

5. Output: Prints directly whether palindrome or not.

Step-by-Step

1. Function defined: `palindrome(num)` encapsulates the logic.

2. Inside function:

- Store original number in `temp`.
- Reverse the number using same loop logic.
- Compare `temp` with `rev`.
- Return True if palindrome, else False.

3. Main program:

- Take input from user.
- Call the function: `palindrome(num)`.
- Print the returned result (True or False).

A screenshot of the Visual Studio Code (VS Code) interface. The code editor shows a Python file named `palindrome.py` with the following content:

```
def is_palindrome_stack(s):
    s = str(s)
    stack = []
    for char in s:
        stack.append(char)

    for char in s:
        if char != stack.pop():
            return False
    return True

num = int(input())
print(is_palindrome_stack(num))
```

The terminal below the editor shows the output of running the script:

```
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & c:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding/palindrome.py
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding> & c:/Users/nanip/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nanip/OneDrive/Desktop/AIAssistedcoding/palindrome.py
121
True
PS C:\Users\nanip\OneDrive\Desktop\AIAssistedcoding>
```

The status bar at the bottom indicates the file is 3.13.7.

#Task5:

Write python program for palindrome using recursion

A screenshot of the Visual Studio Code (VS Code) interface. The code editor shows a Python file named `palindrome.py` with the following content:

```
def palindrome(num):
    if num == 0:
        return True
    return False

num=int(input())
print(palindrome(num))

#Task-3
def is_palindrome_using_recursion():
    def is_palindrome_recursive(num, original=None):
        if original is None:
            original = num
        if num == 0:
            return original == 0
        rem = num % 10
        return rem == (original % (10 ** len(str(original)))) // (10 ** (len(str(original)) - 1)) and is_palindrome_recursive(num // 10)
    # Alternative simpler approach using string reversal
    def is_palindrome_recursive_str(s):
        if len(s) <= 1:
            return True
        return s[0] == s[-1] and is_palindrome_recursive_str(s[1:-1])
    num = int(input())
    print(is_palindrome_recursive(str(num)))
```

Output:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and several icons. The left sidebar has sections for Explorer, Open Editors, and AI Assisted Coding. In the Open Editors section, there are files named Welcome, palindrome.py, and first.py. The main editor area contains Python code for checking if a number is a palindrome. The code uses two functions: `is_palindrome` (which converts a number to a string and checks it) and `is_palindrome_recursive` (which uses recursion). The terminal at the bottom shows the command `PS C:\Users\manip\OneDrive\Desktop\AIAssistedCoding> & c:/users/manip/AppData/Local/Programs/Python/Python311/python.exe c:/users/manip/OneDrive/Desktop/AIAssistedCoding/palindrome.py` followed by the output: 121 is palindrome, True. The status bar at the bottom right shows information like lines (1n 30), columns (Col 28), spaces (Spaces: 4), encoding (UTF-8), file type (CR LF), language (Python), and date/time (09-01-2026).

```
def is_palindrome(num):
    s = str(num)
    return s == s[::-1]

def is_palindrome_recursive(num, original=None):
    if original is None:
        original = num

    if num == 0:
        return original == 0

    rem = num % 10
    num = num // 10
    rem == (original % (10 ** len(str(original)))) // (10 ** (len(str(original)) - 1)) and is_palindrome_recursive(num // 10)

print(is_palindrome(121))
print(is_palindrome_recursive(121))
```

Step-by-Step Explanation

1. Convert number to string

- `str(num)` turns the input number into a string.
- Example: if user enters 121, then s = "121".

2. Recursive function logic

- `is_palindrome_recursive(s)` checks if the string s is a palindrome.

3 Execution Example: Input = 121

- s = "121"
- Step 1: Compare "1" (first) and "1" (last) → equal → recurse on "2".
- Step 2: "2" has length 1 → base case → return True.
- Final result: True.