**PALINDROME INTEGER CODE AND EXAMPLE EXPLAINATION**

**PYTHON CODE:**

original\_number = int(input("Enter a positive integer: "))

temp\_number = original\_number

reversed\_number = 0

while temp\_number > 0:

last\_digit = temp\_number % 10 # Get the last digit

reversed\_number = reversed\_number \* 10 + last\_digit # Append last digit to reversed number

temp\_number = temp\_number // 10 # Remove the last digit

if reversed\_number == original\_number:

print(f"The number {original\_number} is a palindrome integer.")

else:

print(f"The number {original\_number} is not a palindrome integer.")

**Input: 121**

1. **Initial Setup**:
   * The user enters the number 121.
   * original\_number = 121 (the number provided by the user).
   * temp\_number = 121 (copy of the original number, used for manipulation).
   * reversed\_number = 0 (starting value of the reversed number, initially empty).

**Step-by-Step Execution of the while Loop:**

**First Iteration:**

* **Condition Check**:
  + temp\_number > 0 → 121 > 0 is True, so we enter the loop.
* **Extract the Last Digit**:
  + last\_digit = temp\_number % 10 = 121 % 10 = 1.
  + last\_digit stores the value 1, which is the last digit of temp\_number.
* **Update reversed\_number**:
  + reversed\_number = reversed\_number \* 10 + last\_digit.
  + reversed\_number = 0 \* 10 + 1 = 1.
  + reversed\_number is now 1.
* **Remove the Last Digit from temp\_number**:
  + temp\_number = temp\_number // 10 = 121 // 10 = 12.
  + temp\_number is now 12.

**Second Iteration:**

* **Condition Check**:
  + temp\_number > 0 → 12 > 0 is True, so we continue the loop.
* **Extract the Last Digit**:
  + last\_digit = temp\_number % 10 = 12 % 10 = 2.
  + last\_digit stores the value 2, which is the last digit of temp\_number.
* **Update reversed\_number**:
  + reversed\_number = reversed\_number \* 10 + last\_digit.
  + reversed\_number = 1 \* 10 + 2 = 12.
  + reversed\_number is now 12.
* **Remove the Last Digit from temp\_number**:
  + temp\_number = temp\_number // 10 = 12 // 10 = 1.
  + temp\_number is now 1.

**Third Iteration:**

* **Condition Check**:
  + temp\_number > 0 → 1 > 0 is True, so we continue the loop.
* **Extract the Last Digit**:
  + last\_digit = temp\_number % 10 = 1 % 10 = 1.
  + last\_digit stores the value 1, which is the last digit of temp\_number.
* **Update reversed\_number**:
  + reversed\_number = reversed\_number \* 10 + last\_digit.
  + reversed\_number = 12 \* 10 + 1 = 121.
  + reversed\_number is now 121.
* **Remove the Last Digit from temp\_number**:
  + temp\_number = temp\_number // 10 = 1 // 10 = 0.
  + temp\_number is now 0.

**Exit the Loop:**

* The loop ends because temp\_number > 0 → 0 > 0 is False.

**Palindrome Check:**

1. **Comparison**:
   * Check if reversed\_number is equal to original\_number.
   * reversed\_number == original\_number → 121 == 121 is True.
2. **Output**:
   * Since the condition is True, the program prints:
   * The number 121 is a palindrome integer.

**Memory States After Each Iteration:**

| **Iteration** | **temp\_number** | **last\_digit** | **reversed\_number** |
| --- | --- | --- | --- |
| Initial | 121 | N/A | 0 |
| 1st | 12 | 1 | 1 |
| 2nd | 1 | 2 | 12 |
| 3rd | 0 | 1 | 121 |

**Why Is This a Palindrome?**

* The number 121 reads the same forward (121) and backward (121), so it is a palindrome.

**Summary of Operations:**

* The number is reversed by repeatedly extracting the last digit and adding it to the reverse, shifted left by a decimal place.
* After constructing the reversed number, it matches the original, confirming the palindrome property.

This is how the program checks for palindromes with **extreme detail**! 😊