```python
def non_recursive_fibonacci(n):
    first = 0
    second = 1
    steps = 0
    print(first, second, end=" ")  # Print first two numbers in one
line
    for _ in range(n - 2):
        third = first + second
        steps+=1
        print(third, end=" ")
        first = second
        second = third
    print(f"\nTotal Steps : {steps}")

# Get user input for the number of elements
n = int(input("Enter the number of elements: "))
non_recursive_fibonacci(n)
```

```
Enter the number of elements:  10

0 1 1 2 3 5 8 13 21 34
Total Steps : 8
```

```python
def recursive_fibonacci(n):
    # Increment the step count for each call
    if n <= 1:
        return n
    else:
        return recursive_fibonacci(n - 1) + recursive_fibonacci(n - 2)

if __name__ == "__main__":
    n = int(input("Enter the number of elements: "))
    for i in range(n):
        print(recursive_fibonacci(i), end=" ")
```

```
Enter the number of elements:  10

0 1 1 2 3 5 8 13 21 34
```

```python
def recursive_fibonacci(n, steps):
    # Increment the step count for each call
    steps[0] += 1
    if n <= 1:
        return n
    else:
        return recursive_fibonacci(n - 1, steps) +
recursive_fibonacci(n - 2, steps)

if __name__ == "__main__":
    n = int(input("Enter the number of elements: "))
    steps = [0]  # Use a list to store steps count as a mutable
```

```
reference
    for i in range(n):
        print(recursive_fibonacci(i, steps), end=" ")
    print(f"\nTotal Steps: {steps[0]}")

Enter the number of elements:  10

0 1 1 2 3 5 8 13 21 34
Total Steps: 276

def fib(n, f):
    if n == 0 or n == 1:
        return n
    if f[n] != 0:
        return f[n]

    f[n] = fib(n - 1, f) + fib(n - 2, f)
    return f[n]

if __name__ == "__main__":
    n = int(input("Enter the number of elements: "))
    f = [0] * (n + 1)  # List to store previously computed Fibonacci
numbers
    for i in range(n):
        print(fib(i, f), end=" ")

Enter the number of elements:  10

0 1 1 2 3 5 8 13 21 34
```