

# Solution 14: Python While Loop

---

## 1) Fill in the blanks: What is a `while` loop?

---

A `while` loop keeps running while the condition is **True**.

- The condition is checked **before** each loop.
  - The code inside the loop must be indented **4** spaces.
  - If variables inside the loop don't change, the while loop **won't** stop.
- 

## 2) What does this print?

---

Output:

```
3
5
7
```

---

## 3) Fix the infinite loop

---

One correct answer is:

```
n = n - 1
```

Full fixed code:

```
n = 5

while n > 0:
    print(n)
    n = n - 1
```

---

## 4) What does this print?

---

We add `1 + 2 + 3 + 4` .

Output:

10

---

## 5) Count digits

---

`9000 → 900 → 90 → 9 → 0` , so there are 4 digits.

Output:

4

---

## 6) Sum of digits

---

Digits are `5` , `0` , and `7` . Sum is `5 + 0 + 7 = 12` .

Output:

12

---

## 7) Product of digits (write code)

---

One correct solution:

```
n = 2468
product = 1

while n > 0:
    digit = n % 10
    product = product * digit
    n = n // 10

print(product)
```

Output:

384

---

## 8) Count even digits

---

Even digits in `24861` are `2, 4, 8, 6` → 4 even digits.

Output:

4

---

## 9) Fibonacci numbers up to 20

---

Output:

[1, 1, 2, 3, 5, 8, 13]

---

## 10) Squares up to 50 (write code)

---

One correct solution:

```
squares = []
i = 1

while i * i <= 50:
    squares.append(i * i)
    i = i + 1

print(squares)
```

Output:

[1, 4, 9, 16, 25, 36, 49]

---

## 11) Traverse a list with an index

---

Output:

```
10
20
30
```

---

## 12) Collatz list

Output:

```
[6, 3, 10, 5, 16, 8, 4, 2, 1]
```

---

## 13) Fill in the Collatz rules

- If `n` is even: `n = n // 2`
- If `n` is odd: `n = 3 * n + 1`

---

## 14) Challenge: Count Collatz steps (write code)

One correct solution:

```
n = 7
steps = 0

while n != 1:
    if n % 2 == 0:
        n = n // 2
    else:
        n = 3 * n + 1
    steps = steps + 1

print(steps)
```

Output:

```
16
```

