

# Quiz 06–12: Lists and For-Loops

Name: \_\_\_\_\_ Date: \_\_\_\_\_

## Instructions

---

- Answer in the blanks.
  - For “write code” questions, write valid Python code (no functions needed).
  - For “what does it print” questions, write the **exact** output (line by line).
  - You may assume all inputs are valid (unless the question says otherwise).
- 

## Part A — Lists (create, index, change)

---

### 1) Build a list (mixed types)

Write Python code to create a list named `items` that contains (in this order):

- the integer `5`
- the string `"hi"`
- the integer `0`

---

### 2) Index and change

Given:

```
nums = [10, 20, 30, 40]
nums[1] = nums[1] + 5
nums[3] = 99
print(nums)
```

Output:

---

### 3) `append` and `pop`

Given:

```
letters = [ "A", "B" ]
letters.append( "C" )
x = letters.pop()
letters.append( "D" )
print(x)
print(letters)
```

Output:

#### 4) Insert and remove

You are given a list of strings. Write code to:

- 1) insert "yellow" at index 1
- 2) remove "blue" from the list
- 3) print the final list

```
colors = [ "red", "green", "blue" ]  
  
_____ # insert  
_____ # remove  
_____ # print
```

Output:

### Part B – `range(...)` (start/stop/step)

#### 5) What does `range(stop)` produce?

```
nums = list(range(6))
print(nums)
```

Output:

#### 6) What does `range(start, stop)` produce?

```
nums = list(range(3, 9))
print(nums)
```

Output:

## 7) Step forward

```
nums = list(range(2, 13, 3))
print(nums)
```

Output:

## 8) Step backward (negative step)

```
nums = list(range(10, 2, -2))
print(nums)
```

Output:

# Part C – For-loop with lists (traversal + accumulators)

---

## 9) Count positives

Write code to count how many numbers are **greater than 0**. Print the count.

```
nums = [-2, 5, 0, 7, -1, 3]

count = 0

# complete the code:

print(count)
```

Output:

## 10) Sum of even numbers in a list

Write code to compute the sum of the **even** numbers only. Print the sum.

```
nums = [4, 1, 6, 9, 2, 8]
total = 0
# complete the code:
print(total)
```

## Output:

## 11) Build a new list (not in-place)

You are given a list: `words` .

Build a new list named `long_words` that keeps only the words with length  $\geq 3$ .

Then print `long_words` .

```
words = ["a", "hi", "robot", "ok"]  
  
# complete the code:  
  
long_words = _____  
  
for _____:  
  
    print(long_words)
```

## Output:

## 12) Reverse a list (not in-place)

You are given a list: `nums` .

Write code to create a **new** list named `rev` that is the reverse of `nums` .

(Do not change  nums .)

```
nums = [3, 1, 4, 1, 5]

# complete the code:

n = len(nums)
rev = _____

for i in range(_____, _____, _____):
    rev._____

print(rev)
```

Output:

## Part D — **break** (stop early)

### 13) First multiple of 7

You are given a list: `nums`.

Write code that finds the **first** number in `nums` that is a multiple of 7.

Print that number and stop the loop using `break`.

```
nums = [5, 11, 13, 21, 8, 28]

for x in nums:
    # complete the code:
```

Output (the first multiple of 7):

### 14) Search a range with **break**

Write code to find the **first** number between 1 and 50 (inclusive) that is divisible by both **6 and 8**.

Print it, then stop.

```
for x in range(_____, _____):
    if _____:
        print(_____)
        _____ # stop the loop
```

Output:

## 15) Stop when the running sum gets big

**Running sum** means the sum from the leftmost item to the current item.

Add numbers from left to right until the running sum becomes  $\geq 13$ . Print the running sum at the moment you stop.

```
nums = [2, 4, 6, 8, 10]

total = 0 # running sum

for x in nums:
    # complete the code
    total = _____
    if total >= 13:
        print(total)
        _____ # stop the loop
```

Output:

## Part E – Number problems ( $\%$ and $//$ )

### 16) Digits: last digit and “remove last digit”

Let  $n = 5089$ .

Fill in the blanks:

- $n \% 10$  is \_\_\_\_\_
- $n // 10$  is \_\_\_\_\_

### 17) Sum of digits

Write code to compute the sum of digits in  $n$ .

(Hint: Use  $\% 10$  and  $// 10$  in a loop, and stop when  $n$  becomes 0.)

```
n = 32046

total = 0

for _ in range(1000):
    x = _____ # get the last digit
    n = _____ # remove the last digit from n
    total = _____ # add digit to total

    if _____:
        break

print(total)
```

Output:

## 18) Count how many digits

- Write code to count the number of digits in `n`.
- Example: `7` has 1 digit, `2026` has 4 digits.
- Assume `n` is a positive integer.

```
n = 900120

count = 0

for _ in range(1000):
    if _____:
        break

    n = _____ # remove the last digit from n
    count = _____ # increase the digit count

print(count)
```

Output:

## 19) Factors in a range

- Write code to print all factors of `n`.
- A factor is a number `d` such that `n % d == 0`.
- Print factors in increasing order.

```
n = 36

for d in range(_____, _____):
    if _____:
        print(d)
```

Output:

## 20) Prime check

- Write code to decide whether an integer `n` is prime.
- Assume `n > 2`.
- Print `"prime"` or `"not prime"`.

```

n = 29
is_prime = _____

for d in range(_____, _____):
    if _____:
        is_prime = _____
        break

if is_prime:
    print(_____)
else:
    print(_____)

```

Output:

## Part F — Nested for-loops (combinations + patterns)

### 21) All pairs (store in a list)

Use a **nested** loop to build a list named `pairs` that contains:

```
[ "A1", "A2", "A3", "B1", "B2", "B3" ]
```

```

letters = ["A", "B"]
nums = [1, 2, 3]

pairs = []

for l in letters:
    for n in nums:
        # complete the code

print(pairs)

```

Expected output:

```
[ 'A1', 'A2', 'A3', 'B1', 'B2', 'B3' ]
```

### 22) Mini table (nested loops + spacing)

Use nested loops to print this 3x4 rectangle of `#`.

Write the code:

Expected output:

```
####  
####  
####
```

### 23) Multiplication grid (nested loops + `range`)

Write code to print a 1–5 multiplication grid (5 rows).

Each row should look like:

- Row 1: `[1, 2, 3, 4, 5]`
- Row 2: `[2, 4, 6, 8, 10]`
- ...
- Row 5: `[5, 10, 15, 20, 25]`

```
for n in range(______):  
    row = list(range(______, _____, _____))  
    print(row)
```

Expected output:

```
[1, 2, 3, 4, 5]  
[2, 4, 6, 8, 10]  
[3, 6, 9, 12, 15]  
[4, 8, 12, 16, 20]  
[5, 10, 15, 20, 25]
```