Quiz 08–14: For-Loops and While-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 — For-loop with lists (traversal + accumulators)

1) Double each number (what does it print?)

```
numbers = [2, 5, 1, 4]

for x in numbers:
    print(x * 2)
```

Output:

```

```

2) Running total (fill in the blanks)

Fill the blanks so the code prints the running total.

```
numbers = [3, 1, 4]

total = 0
for x in numbers:
    total = _____________________
    print(total)
```

Expected output:

```
3
4
8
```

3) Indentation fix

This code is supposed to print every number in the list. Fix the indentation.

```
numbers = [9, 8, 7]
for x in numbers:
print(x)
```

Write the corrected code:

```

```

4) Build a new list: squares (write code)

Write code to create a new list `squares` that contains the square of each number in `nums`. Then print `squares`.

```
nums = [1, 3, 5, 2]
squares = []

# complete the code:




print(squares)
```

Expected output:

```
[1, 9, 25, 4]
```

Part B — `break` and searching

5) First multiple of 3 (what does it print?)

```
nums = [4, 6, 9, 12, 15]

for x in nums:
    if x % 3 == 0:
        print("first multiple of 3:", x)
        break
    print("not yet:", x)
```

Output:

```

```

6) Find an index (fill in the blanks)

Fill the blanks so the code prints the index of `target` in the list, or `-1` if not found.

```
nums = [10, 20, 30, 40]
target = 30

idx = -1
for i in range(len(nums)):
    if nums[i] == target:
        idx = ____
        __________
print(idx)
```

Expected output:

```
2
```

7) Stop when the total gets too big (write code)

- Traverse `nums` and add numbers to `total`.
- Print `total` after each addition.
- If `total` becomes **greater than 10**, stop early using `break`.

```
nums = [2, 4, 5, 3]

total = 0

# complete the code:




print(total)
```

Output:

```

```

8) First even number (what does it print?)

```
nums = [3, 9, 5, 8, 6]

for x in nums:
    if x % 2 == 0:
        print("first even:", x)
        break
```

Output:

```

```

Part C — Number problems (`%` and `//`)

9) Last digit and “remove last digit” (what does it print?)

```
n = 507

print(n % 10)
print(n // 10)
print((n // 10) % 10)
```

Output:

```

```

10) Sum of digits (write code)

Write code to compute the sum of digits of `n` using a `while` loop.

```
n = 2305
total = 0

# complete the code:

while ____________________:




print(total)
```

Output:

```

```

11) Factors by hand

List all positive factors of `24` (in increasing order):

```
_____________________________________________________________
```

12) Print factors, but stop early (write code)

Print factors of `n` in ascending order (from small to big).
Stop early if you have printed **4** factors.

```
n = 30
count = 0

# complete the code:
for i in range(_____, _____):
    if ____________________:  # i is a factor of n
        print(i)
        count = _____________
    if ____________________:
        break
```

Expected output:

```
1
2
3
5
```

13) Prime or not?

A number is prime if it has exactly two factors: `1` and itself.

Is `29` prime?

```
__________ (True or False?)
```

Part D — Nested loops and 2D lists

14) Pairs (what does it print?)

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

for ch in letters:
    for x in nums:
        print(ch, x)
```

Output:

```

```

15) Counting prints

```
outer = [10, 20, 30]
inner = ["X", "Y"]

for a in outer:
    for b in inner:
        print(a, b)
```

How many times does `print` run?

```
__________ (write a whole number here)
```

16) Sum all numbers in a 2D list (write code)

```
grid = [
    [1, 2, 3],
    [4, 5, 6]
]

total = 0

# complete the code:
for _____________________:
    for _____________________:
        total = _______________

print(total)
```

Output:

```

```

17) Build a small multiplication table (write code)

Create a 2D list `table` with 3 rows and 4 columns.

```
table = []

# complete the code:
for i in range(_______, _______):  # row i
    row = []
    for j in range(_______, _______):  # column j
        s = str(i) + " x " + str(j) + " = " + str(i * j)
        row.append(s)
    table.append(__________)  # add the row to table

for row in table:
    print(row)
```

Expected output:

```
['1 x 1 = 1', '1 x 2 = 2', '1 x 3 = 3', '1 x 4 = 4']
['2 x 1 = 2', '2 x 2 = 4', '2 x 3 = 6', '2 x 4 = 8']
['3 x 1 = 3', '3 x 2 = 6', '3 x 3 = 9', '3 x 4 = 12']
```

18) Print a rectangle of stars (write code)

Print a 3-by-5 rectangle of `*` (3 rows, 5 stars per row):

```
*****
*****
*****
```

Write code:

```

```

Part E — Challenges (mix topics)

19) Digit sums for a list (for + while)

Create a list `digit_sums` where each item is the sum of digits of the corresponding number.

```
nums = [12, 305, 40, 7]
digit_sums = []

# complete the code:
for n in nums:
    total = 0
    k = n
    while k > 0:
        x = _______________  # get the last digit of k
        k = _______________  # remove the last digit of k
        total = ___________

    digit_sums.append(total)

print(digit_sums)
```

Expected output:

```
[3, 8, 4, 7]
```

---

20) Max digit sum (track both number and sum)

Traverse `nums` and find the **largest digit sum**.

Example: - `nums = [91, 28, 100, 47]` is the input. - `digit_sums = [10, 10, 1, 11]` stores the digit sum of each number in `nums`. - Output is `11`, the max of `digit_sums`.

```
nums = [91, 28, 100, 47]

# Find digit sum for every number in nums:
digit_sums = []
for n in nums:
    total = 0
    k = n
    while k ___ 0:  # choose from >, >=, <, <=, ==
        x = _________________  # get the last digit of k
        k = _________________  # remove the last digit of k
        total = ____________

    digit_sums.append(total)

# Find the max in digit_sums:
m = digit_sums[0]
for s in digit_sums:
    if ______________________:
        ______________________

print(m)
```

Expected output:

```
11
```

21) Search a 2D list and stop early (nested loops + `break`)

Check whether `target` appears in the grid.

- If found, print `Found`
- Otherwise, print `Not found`
- Stop early using `break` when you find it.

```
grid = [
    [5, 1, 9],
    [2, 7, 3],
    [4, 6, 8]
]
target = 7

found = False

# complete the code:
for row in grid:
    for n in row:
        if _____________________:
            found = ____________
            _____________  # stop the inner loop
    if found:
        _____________  # stop the outer loop

if found:
    print("Found")
else:
    print("Not found")
```

Output:

```

```

22) Fibonacci list + count (while + list)

- Start with `fib = [1, 1]`.
- Keep appending the next Fibonacci number while the next number would be `<= 100`.
- Then print:
  - the list `fib`
  - the number of items in the list (`len(fib)`)

```
fib = [1, 1]

# complete the code:
while ____________________________:
    fib.append(_____________________)

print(fib)
print(len(fib))
```

Output:

```
__________________________________________________
________
```

23) Two Sum (nested loops + `break`)

Given `nums` and `target`, determine whether there exists a pair of **different indices** `(i, j)` such that:
`nums[i] + nums[j] == target`

- If such a pair exists, print `True`
- Otherwise, print `False`
- Stop early using `break` when you know the answer

```
nums = [4, 1, 9, 2, 7]
target = 11

n = len(nums)
found = _______________  # True or False?

for i in range(______, ______):
    for j in range(______, ______):
        if _____________________________:
            found = ________  # True or False?
            _________________  # stop the inner loop

    if _________________:
        _________________  # stop the outer loop

print(found)
```

Output:

```

```