Online Test Example

The questions below are examples. The real exam uses different questions, in the same style and at the same level.

About the exam

Test is taken online, from anywhere, before you are invited to the in-person admission exam.

- **24 questions**, multiple choice, exactly one correct answer each.
- **About 25 minutes.**
- Six areas, four questions each: programming fundamentals, OOP concepts, programming logic, code reading, debugging, and logical thinking.
- Available in **Armenian and English** — choose your language before you start.
- You do not write any code. Every question is answered by choosing one of four options.

Its purpose is to confirm that applications come from real, genuinely interested candidates. It does not rank you, and it is not the admission exam — that one is taken in person and is the only one that decides admission.

Pseudocode conventions

The questions use a simple pseudocode that is not tied to any one programming language. The same conventions apply in the real exam:

- `for i = A to B` is inclusive of both A and B.
- Arrays/lists are 0-indexed; strings are indexed the same way as arrays of characters.
- Indentation defines block scope (like Python), regardless of the absence of explicit `end` keywords or braces.
- `=` always means assignment; `==` always means equality comparison.
- AND and OR are evaluated left to right, and the right-hand side is not evaluated at all if the left-hand side already decides the result.
- `for each x in list` iterates in the list's given order.
- `break` immediately ends the innermost loop it appears in; execution continues at the first line after that loop.
- `length(x)` returns the number of elements in a list, or characters in a string. `str(n)` converts a number to its text form.
- `class Name` starts a class; its fields and methods are written indented beneath it.

- `new Name(args)` creates an object of that class. If the class defines a method named `init`, it runs automatically at that moment and receives those arguments.
- Inside a method, `this` refers to the object it was called on; outside, `obj.field` and `obj.method(args)` reach its members.
- `class B extends A` makes B inherit A's fields and methods; if B defines a method A also defines, B's version runs for B objects.
- Line numbers shown in debugging questions are for reference only and are not part of the executed pseudocode.

Example questions

Programming fundamentals

1. What will the following code print?

```
values = [7, 3, 9]
total = 0
for each v in values
    total = total + v
print(total)
```

A. 19

B. 16

C. 21

D. 3

2. What will the following code print?

```
function triple(n)
    return n * 3
x = triple(2) + triple(1)
print(x)
```

A. 18

B. 6

C. 9

D. 3

OOP concepts

3. What is a "method" in OOP?

A. A variable stored inside an object

B. A function defined inside a class, which operates on that class's objects

C. A copy of an object that is made when it is passed to a function

D. A loop that visits every object of a class

4. What will the following code print?

```
class Lamp
    init()
        this.on = false
    toggle()
        this.on = NOT this.on

l = new Lamp()
l.toggle()
l.toggle()
print(l.on)
```

A. true

B. 2

C. Nothing is printed

D. false

Programming logic

5. What will the following code print?

```
x = 2
if NOT (x > 5)
    print(x + 1)
else
    print(x - 1)
```

A. 3

B. 1

C. 2

D. 7

6. What is printed in total?

```
function check()
    print("checked")
    return true
n = 0
if n > 0 AND check()
    print("ok")
else
    print("skip")
```

A. checked, then skip

B. ok (only)

C. skip (only)

D. checked, then ok

Code reading

7. What will the following code print?

```
word = "test"
print(word[1])
```

A. t

B. e

C. s

D. 1

8. What will the following code print?

```
total = 0
for i = 1 to 5
    if i MOD 2 == 0
        total = total + i
print(total)
```

A. 15

B. 9

C. 4

D. 6

Debugging

9. This is intended to print how many values in the list are even (expected: 2), but it prints 3. Which line is the bug?

```
line1: values = [3, 8, 5, 12, 7]
line2: count = 0
line3: for each v in values
line4:     if v MOD 2 == 1
line5:         count = count + 1
line6: print(count)
```

A. line1

B. line3

C. line4

D. line5

10. This is intended to print the larger of a and b (expected: 9), but it prints 4. Which line is the bug?

```
line1: a = 4
line2: b = 9
line3: larger = a
line4: if a > b
line5:     larger = b
line6: print(larger)
```

A. line4

B. line3

C. line5

D. line6

Logical thinking

11. Every pull request in this project is labelled either "bug" or "feature" — never both, and never neither. This pull request is not labelled "bug". Is it labelled "feature"?

A. No — it could have no label at all

B. Yes — but only if someone has reviewed it

C. No — it could carry both labels

D. Yes — the two labels cover every pull request, so it must be the other one

12. A colleague says: "It is not true that both tests passed." What must be true?

A. Both tests failed

B. At least one of the two tests failed

C. Exactly one test failed

D. Nothing follows about either test

Answers

#	Answer	Why
1	**A — 19**	The loop adds every element: 7 + 3 + 9 = 19.
2	**C — 9**	triple(2) is 6 and triple(1) is 3; the two results are added.
3	**B**	A method is behaviour defined in a class and available on its objects. A describes a field.
4	**D — false**	`on` starts false; two toggles return it to its starting value. Toggling twice is not the same as toggling once.
5	**A — 3**	`x` is 2, so `x > 5` is false and NOT makes the condition true; the first branch prints 2 + 1.
6	**C — skip (only)**	AND stops as soon as one side is false. `n > 0` is false, so `check()` is never called and never prints.
7	**B — e**	Indexing starts at 0, so `word[1]` is the second character of "test".
8	**D — 6**	Only the even values are added: 2 + 4 = 6.
9	**C — line4**	The test keeps values with remainder 1 — the odd ones (3, 5, 7). It should be `v MOD 2 == 0`.
10	**A — line4**	`a > b` is false, so `larger` keeps its initial value, 4. The comparison should be `b > a`.
11	**D — Yes**	The two labels are stated to cover every pull request with no overlap, so not being one means being the other.
12	**B — At least one failed**	"Not both passed" rules out only the case where both passed. It does not establish that both failed.