4.1 Objects and classes

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS class • reference

Objective

Design classes, trace object state through references, and tell equality from identity.

1 Key knowledge and worked examples

■ The lookup order

`obj.x` walks outward: the instance dict, then the class dict, then the bases. Assigning `obj.count = 9` shadows the class value for that one instance only – the class attribute is untouched ✓.

■ Worked example: one object, two names

`a = Book('Signals', 3)`, `b = Book('Signals', 3)`, `c = a`; then `a.take()` twice: `a.copies` is 1, `b.copies` is 3, `c.copies` is 1 – c and a are the same object ✓. `a is c` is `True`, `a == b` is `False`: default equality is identity ✓.

2 Practice

2.1 A class counts its instances in a class attribute starting at 1. Two instances are created. State where the counter lives and what it reads. [3]

2.2 After `d = c` on two Book references, `c.take()` runs once. State what `d.copies` shows and why. [3]

2.3 Two distinct Books hold identical fields. State the result of `==` (no `__eq__` defined) and of `is`, and name what each compares. [3]

3 Exam-style questions

3.1 A `Copy` class starts as:

```
class Copy:
    total = 0
    def __init__(self, title):
        self.title = title
        Copy.total += 1
```

Three copies are created; then `x.total = 99` is executed on the first of them. State the value of `Copy.total`, the value of `x.total`, and the value of `.total` on the other two copies, explaining your answers by the lookup order. Explain in one line why the shared counter belongs on the class, not in each instance. [8]

3.2 This program runs:

```
p = Book('Signals', 3)
q = Book('Signals', 3)
r = p
p.take()
```

Give the value of `.copies` read through each of `p`, `q` and `r`, with a reason for each. State the outcome of `p == q`, of `p is r`, and of `q is p`, naming what equality and identity each compare. [7]

Solutions

2.1

- On the class (class attribute) ✓.
- Reads 3: incremented once per construction ✓.

2.2

- One less than before – the same object ✓.
- `d` is an alias of `c`: assignment copies the reference, not the object ✓.

2.3

- `==` is `False`: with no `__eq__`, equality is identity ✓.
- `is` is `False`: two distinct objects ✓. Identity compares objects; equality compares values once defined.

3.1

- `Copy.total` is 3 ✓.
- `x.total` is 99 – the assignment shadows in the instance dict ✓.
- The other two read 3: their instance dicts are empty, lookup falls to the class ✓.
- One counter must agree across all instances – shared state lives where it is shared, on the class ✓.

3.2

- `p.copies` is 2, `r.copies` is 2 – same object, one take ✓.
- `q.copies` is 3 – a second object, untouched ✓.
- `p == q` is `False` (default equality is identity); `p is r` is `True` (one object); `q is p` is `False` (distinct objects) ✓. Equality compares values, identity compares objects ✓.