Lab 01: Python Building Blocks — Answer Key

COMP 102 — Introduction to Computing

Forman Christian University — Spring 2026

---

1 Scalar Objects & Types

Exercise 1: `type(42)` → `<class 'int'>` `type(3.14)` → `<class 'float'>`

Exercise 2: `type(True)` → `<class 'bool'>` `type(None)` → `<class 'NoneType'>`

Exercise 3: `type(9.0)` → `<class 'float'>` — because of the decimal point, even though 9 is a whole number.
`type(7)` → `<class 'int'>` `type(False)` → `<class 'bool'>`

Exercise 4: `type(0)` → `<class 'int'>` `type(0.0)` → `<class 'float'>` `type(-3.5)` → `<class 'float'>`

Exercise 5:
`3 + 4.0` → `float` (mixing int and float promotes to float, result is `7.0`)
`10 / 2` → `float` (the `/` operator *always* returns a float, result is `5.0`)
`10 // 2` → `int` (integer division with two ints returns int, result is `5`)

2 Type Conversions

Exercise 6: `float(5)` → `5.0` `int(3.0)` → `3`

Exercise 7: `float(True)` → `1.0` `int(False)` → `0`
(`True` is treated as `1`, `False` is treated as `0`)

Exercise 8: `int(7.9)` → `7` (truncates — drops the decimal, does NOT round)
`round(7.9)` → `8` (rounds to nearest integer)

Exercise 9: `int(3.1) + int(4.9)` = `3 + 4` = `7` (both truncated)
`round(3.1) + round(4.9)` = `3 + 5` = `8` (both rounded)

Exercise 10:
Step 1: `int(9.7)` = `9` (truncates)
Step 2: `9 + 0.6` = `9.6`
Step 3: `round(9.6)` = `10`
Step 4: `float(10)` = `10.0`

Final answer: `10.0`

3 Arithmetic Expressions

Exercise 11: `3 + 4` → `7` `10 - 6` → `4`

Exercise 12: `10 / 2` → `5.0` (always float!) `2 ** 3` → `8`

Exercise 13: `7 % 3` → `1` (remainder of $7 \div 3$) `10 % 5` → `0` (divides evenly)

Exercise 14: `15 // 4` → `3` (integer division, drops remainder) `7 // 2` → `3`

Exercise 15: `3 + 4 * 2` → `11` (multiplication first: $4 \times 2 = 8$, then $3 + 8 = 11$)
`(3 + 4) * 2` → `14` (parentheses first: $3 + 4 = 7$, then $7 \times 2 = 14$)

Exercise 16: `type(4 * 3)` → `<class 'int'>` (int × int = int)
`type(4.0 * 3)` → `<class 'float'>` (float × int = float)

Exercise 17: `(13 - 4) / (12 * 12)` = `9 / 144` = `0.0625`
The `/` operator always returns a float.

Exercise 18: `5 % 2` → `1` `2 ** 5` → `32` `17 % 5` → `2`

Exercise 19: `2 ** 3 ** 2` = `2 ** (3 ** 2)` = `2 ** 9` = `512`
Note: `(2 ** 3) ** 2` would give `64`, which is different. The `**` operator associates right-to-left.

Exercise 20:
Trace of `(13 - 4) / (2 * 3) + 5 % 3 ** 2`:

1. `3 ** 2` = `9` (`**` has highest precedence)
2. `13 - 4` = `9` (parentheses)
3. `2 * 3` = `6` (parentheses)
4. `9 / 6` = `1.5` (left-to-right, same level as `%`)
5. `5 % 9` = `5` (remainder of $5 \div 9$)
6. `1.5 + 5` = `6.5` (addition last)

Final answer: `6.5`

4 Call Expressions & Imports

Exercise 21: `max(3, 5)` → `5` `max(10, 2, 7)` → `10`

Exercise 22: `min(1, 2, 3)` → `1` `min(8, 5)` → `5`

Exercise 23: `pow(2, 10)` → `1024` (2^{10}) `pow(10, 2)` → `100` (10^2)
`pow(a, b)` computes a^b.

Exercise 24: `sqrt(144)` → `12.0`
The type is `float` (the `sqrt` function always returns a float).

Exercise 25:
Step 1: `mul(2, 3)` = `6`
Step 2: `sub(5, 2)` = `3`
Step 3: `pow(2, 3)` = `8`
Step 4: `add(6, 8)` = `14`

5 Variables & Assignment

Exercise 26: `10`

Exercise 27: `Hello`

Exercise 28: (a) `my_age` — **Valid** (b) `2fast` — **Invalid** (cannot start with a number) (c) `_count` — **Valid** (d) `my-name` — **Invalid** (hyphen is subtraction operator) (e) `lunchPrice` — **Valid**

Exercise 29: `78.5` ($3.14 \times 5^2 = 3.14 \times 25 = 78.5$)

Exercise 30: `6` (`x` starts at 5, then becomes $5 + 1 = 6$)

Exercise 31: `6`
No — changing `x` afterward does **not** affect `y`. `y` was bound to the value `6` (which was 3×2 at the time), and it stays `6`.

Exercise 32: `328.08`
`feet` was calculated as $3.2808 \times 100 = 328.08$. Changing `meters` to 200 afterward does **not** recalculate `feet`. Python evaluates lines sequentially — it does not go back.

Exercise 33: `5`
`b = a` binds `b` to the *current value* of `a` (which is 5). Re-binding `a` to 10 does not affect `b`.

Exercise 34: `x = 4, y = 5`
Trace: `x=2` → `y=3` → `x=4` → `y=5`

Exercise 35:

```
x = 1
y = 2
temp = x      # temp = 1
x = y         # x = 2
y = temp      # y = 1
print(x, y) # 2 1
```

6 Strings — Basics

Exercise 36: `len(name)` → `6`

Exercise 37: `Hello World`

Exercise 38: `s1` = `":))"` (`":"` + `"))"` — the `*` is applied to `c` first: 2 × `")"` = `"))"`)

Exercise 39: `result` = `"abababab"` `len(result)` = `6`

Exercise 40:
Step 1: `f + g` = `"ab"`
Step 2: `int(h)` = `3`
Step 3: `"ab" * 3` = `"abababab"`

7 String Indexing & Slicing

Exercise 41: `s[0]` → `'c'` `s[3]` → `'p'`

Exercise 42: `s[4]` → `'1'` `s[6]` → `'2'`

Exercise 43: `s[2:5]` → `'cde'` `s[0:3]` → `'abc'`

Exercise 44: `s[:4]` → `'abcd'` `s[5:]` → `'fgh'`

Exercise 45: `s[-1]` → `'2'` (last character) `s[-3]` → `'1'`

Exercise 46: `s[::2]` → `'aceg'` (every 2nd character starting from 0)
`s[1::2]` → `'bdfh'` (every 2nd character starting from 1)

Exercise 47: `s[::-1]` → `'hgfedcba'` (reverses the string)

Exercise 48: `s = "ABC d3f ghi"` has indices A=0, B=1, C=2, ␣=3, d=4, 3=5, f=6, ␣=7, g=8, h=9, i=10.
Step 1: `len(s)` = `11`
Step 2: `len(s) - 1` = `10`
Step 3: `s[3:10]` = `" d3f gh"` (indices 3 through 9)

Exercise 49: `s[4:0:-1]` → `"d CB"`
With step −1, we go from index 4 down to (but not including) index 0: indices 4, 3, 2, 1 → `'d'`, `' '`, `'C'`, `'B'`

Exercise 50:
(a) `s[6:1:-2]` → `"gec"`
With step −2, we go from index 6 down to (but not including) index 1: indices 6, 4, 2 → `'g'`, `'e'`, `'c'`
(b) `"r" + s[1:]` produces `"rat"`
(Concatenate the new character `"r"` with the slice `s[1:]` which is `"at"`)

8 Input / Output

Exercise 51: `Hello, World|`

Exercise 52: `Hello, Ali|`

Exercise 53: `15`
Without the `int()` cast, `num` would be the string `"5"`, and `"5" * 3` would give `"555"` (string repetition, not multiplication).

Exercise 54: Line 2 prints: `444` (string `"4"` repeated 3 times)
Line 3 prints: `12` (integer 4×3)
They differ because `input()` returns a **string**. `x * 3` repeats the string; `int(x) * 3` multiplies the number.

Exercise 55:

```
verb = input("Enter a verb: ")
print("I can " + verb + " better than you!")
print(verb, verb, verb, verb, verb)
```

Alternative for line 3 using string operations:
`print((verb + " ") * 4 + verb)`

9 f-strings

Exercise 56: `Hello, Ali|`

Exercise 57: `10 + 20 = 30`

Exercise 58: `Python has 6 letters`

Exercise 59: `7/3 = 2.3333333333333335`
(The expression `a/b` is evaluated inside the f-string and produces a float.)

Exercise 60: `3^4 = 81`
(`base=3`, `exp=4`, `result = 3**4 = 81`)

10 Capstone Challenges

Exercise 61 — Temperature Converter:

```
c = float(input("Enter temperature in Celsius: "))
f = 9/5 * c + 32
print(f"{c} C is {f} F")
```

For `c = 37`: $F = \frac{9}{5}(37) + 32 = 66.6 + 32 = 98.6$

Exercise 62 — Tip Splitter:

```
bill = float(input("Enter the bill amount: "))
friends = int(input("How many friends? "))
total = bill * 1.15
share = total / friends
print(f"Each person pays: {share}")
```

For `bill = 80`, `friends = 4`: $80 \times 1.15 = 92$, then $92/4 = 23.0$

Exercise 63 — Initials:

```
first = input("Enter your first name: ")
last = input("Enter your last name: ")
print(first[0].upper() + "." + last[0].upper() + ".")
```

For `first = "ali"`, `last = "khan"`: `"a"[0].upper()` → `"A"`, etc. → `A.K.`

Exercise 64 — Time Breakdown:

```
total = int(input("Enter total seconds: "))
hours = total // 3600
```

```
minutes = (total % 3600) // 60
seconds = total % 60
print(f"{hours} hour(s), {minutes} minute(s), {seconds} second(s)")
```

For `total = 3672`: hours = 3672//3600 = 1, remaining = 3672%3600 = 72, minutes = 72//60 = 1, seconds = 72%60 = 12.

Exercise 65 — Swap Halves:

```
word = input("Enter a word (even length): ")
mid = len(word) // 2
print(word[mid:] + word[:mid])
```

For `word = "abcdef"`: `mid = 3`, so `word[3:] + word[:3]` = `"def" + "abc"` = `"defabc"`

Exercise 66 — Distance Between Two Points:

```
x1 = float(input("x1: "))
y1 = float(input("y1: "))
x2 = float(input("x2: "))
y2 = float(input("y2: "))
d = ((x2 - x1)**2 + (y2 - y1)**2) ** 0.5
print(f"Distance: {d}")
```

For $(0,0)$ to $(3,4)$: $d = \sqrt{9+16} = \sqrt{25} = 5.0$

Exercise 67 — Digit Sum:

```
n = int(input("Enter a 3-digit number: "))
d1 = n // 100
d2 = (n // 10) % 10
d3 = n % 10
total = d1 + d2 + d3
print(f"{d1} + {d2} + {d3} = {total}")
```

For `n = 472`: `d1 = 472 // 100` = 4, `d2 = (472 // 10) % 10 = 47 % 10` = 7, `d3 = 472 % 10` = 2, total = 13.