Python Programming

Level 3 (Hard)

https://cedricf6.github.io/python-course/

Python Programming - Level 3 (Hard) - Questions

Section A: Advanced Output & Variables

1. **Formatted Output**: What is the output of the following code snippet using string concatenation?

```
name = "Sarah"
age = 16
print("Student: " + name + ", Age: " + str(age))
```

Answer:

---

2. **Variable Types and Operations**: What will be the data type of `result` and its value after the following operations?

```
num1 = 7.5
num2 = 2
result = num1 // num2 + num1 % num2
```

Answer:

---

3. **Type Conversion Challenge**: The following code is intended to calculate a percentage. Identify and fix the error:

```
score = "85"
total = "100"
percentage = (score / total) * 100
print(f"{percentage}%")
```

Answer:

Section B: Complex Operators

4. **Operator Precedence:** What is the value of `result` after this calculation?

```
result = 3 * 4 ** 2 // 5 - 2 + 6
```

Answer:

---

5. **Multiple Assignment Operators:** What is the final value of `x` after these operations?

```
x = 10
x += 5
x **= 2
x %= 7
```

Answer:

---

6. **Boolean Logic:** Given `a = True`, `b = False`, and `c = True`, what is the result of:

```
not (a or b) and (c or not b)
```

Answer:

Section C: Nested Decision Making

7. **Multiple Conditions:** What does this code print?

```
temperature = 25
humidity = 80
if temperature > 30:
    if humidity > 75:
        print("Extreme heat warning")
    else:
        print("Hot day")
elif temperature > 20:
    print("Pleasant day")
else:
    print("Cool day")
```

Answer:

---

8. **Complex Condition**: Write a condition that checks if a year is a leap year. A year is a leap year if:

- It is divisible by 4
- But not divisible by 100 unless also divisible by 400

Answer:

---

9. **Membership Operators:** What does this code output?

```
vowels = "aeiou"
word = "rhythm"
count = 0
for letter in word:
    if letter not in vowels:
        count += 1
print(f"Non-vowel letters: {count}")
```

Answer:

---

10. **Error in Logic**: Identify the logical error in this code that checks if a number is between 10 and 50 (exclusive):

```
num = 25
if 10 <= num <= 50:
    print("Within range")
else:
    print("Outside range")
# What's wrong with this logic for checking exclusive range?
```

Answer:

Section D: Advanced Loops

11. **Nested Loop Pattern:** What pattern does this code produce?

```
for i in range(1, 5):
    for j in range(i):
        print("*", end="")
    print()
```

Answer:

---

12. **While Loop with Break:** How many iterations occur before this loop ends?

```
count = 0
total = 0
while True:
    total += count
    count += 1
    if total > 20:
        break
print(f"Count: {count}")
```

Answer:

---

13. **Loop with Continue:** What numbers are printed by this loop?

```
for i in range(1, 11):
    if i % 3 == 0:
        continue
    if i == 8:
        break
    print(i, end=" ")
```

Answer:

---

14. **Loop Efficiency:** This code contains an inefficiency. How can it be improved?

```
numbers = [1, 2, 3, 4, 5]
for i in range(len(numbers)):
    for j in range(len(numbers)):
        print(numbers[i], numbers[j])
```

Answer:

Section E: Functions & Modules

15. **Function Return Values:** What is printed by this code?

```
def process(x, y):
    return x * 2, y + 10

a, b = process(5, 3)
print(f"a={a}, b={b}")
```

Answer:

---

16. **Function Definition:** Write a function called `is_positive_even` that takes a number and returns `True` if it's both positive and even, otherwise `False`.

Answer:

---

17. **Module Import:** What's the difference between these two import statements?

```
import math
# vs
from math import sqrt
```

Answer:

18. **Built-in vs User-defined:** Give an example of when you would create a user-defined function instead of using built-in functions.

Answer:

Section F: String Processing

19. **String Methods Chain:** What is the output?

```
text = "  Hello World!  "
result = text.strip().lower().replace("world", "Python")
print(result)
```

Answer:

---

20. **String Slicing:** What substring is extracted here?

```
message = "Programming is fun"
extract = message[message.find("is"):message.find("is")+4]
print(extract)
```

Answer:

---

21. **String Analysis:** Write code that counts how many words in a sentence start with a vowel.

Answer:

Section G: Lists & Tuples

22. **List Operations**: What is the final state of this list?

```
colors = ["red", "blue", "green"]
colors.insert(1, "yellow")
colors.append(colors.pop(0))
colors.sort(reverse=True)
print(colors)
```

Answer:

---

23. **Tuple vs List**: Why would you use a tuple instead of a list for storing RGB color values like `(255, 0, 0)`?

Answer:

---

24. **List Comprehension:** Convert this loop to a list comprehension:

```
squares = []
for i in range(1, 6):
    squares.append(i ** 2)
```

Answer:

---

25. **Multi-dimensional List**: How do you access the value 9 in this structure?

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

Answer:

Section H: Dictionaries & Random

26. **Dictionary Methods:** What does this code output?

```
student = {"name": "Alex", "grades": [85, 90, 78]}
student["grades"].append(95)
print(student.get("average", "Not calculated"))
print(len(student["grades"]))
```

Answer:

---

27. **Random Application:** Write code that randomly selects 3 unique numbers between 1 and 10.

Answer:

---

28. **Dictionary Iteration:** How would you print all key-value pairs in a dictionary?

Answer:

Section I: Error Handling & Best Practices

29. **Error Types**: Classify these errors:

    a) `print("Hello"` (missing parenthesis)
    b) `x = 5 / 0` (division by zero)
    c) Using `=` instead of `==` in a condition

Answer:

a)
b)
c)

---

30. **Code Quality:** Rewrite this code with better practices:

```
x=10  #stores age
y="yes"  #eligibility
if x>18:print(y)
```

Answer:

Python Programming - Level 3 (Hard) - Answers

Section A Answers:

1. **Student: Sarah, Age: 16** (String concatenation combines all parts)

2. **Data type: float, Value: 4.5** (7.5 // 2 = 3.0, 7.5 % 2 = 1.5, 3.0 + 1.5 = 4.5)

3. **Error**: Cannot divide strings. Fix by converting to numbers:

```
percentage = (int(score) / int(total)) * 100
```

Section B Answers:

4. **13** (Order: 4**2=16, 3*16=48, 48//5=9, 9-2=7, 7+6=13)

5. **1** (x=10, x+=5→15, x**=2→225, x%=7→225%7=1)

6. **False** (a or b = True, not True = False, False and anything = False)

Section C Answers:

7. **Pleasant day** (temperature 25 > 20 but ≤ 30)

8.

```
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
```

9. **Non-vowel letters**: 6 (All letters in "rhythm" are non-vowels)

10. The code is actually correct for exclusive range. For a trick question: if we wanted INCLUSIVE range (10-50 inclusive), it should be `if 10 <= num <= 50:`

Section D Answers:

11. Pattern:

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

12. **Count: 7** (Sums: 0+1+2+3+4+5+6=21, stops when total>20 after adding 6)

13. **1 2 4 5 7** (Skips multiples of 3, stops at 8)

14. **Inefficiency:** Nested loops process 25 iterations (5×5). If order doesn't matter, use:

```
for i in range(len(numbers)):
    for j in range(i, len(numbers)):  # Start from i instead of 0
```

Section E Answers:

15. **a=10, b=13** (Function returns tuple (10, 13))

16.

```
def is_positive_even(n):
    return n > 0 and n % 2 == 0
```

17. `import math` imports entire module (use `math.sqrt()`), while `from math import sqrt` imports only sqrt function (use `sqrt()` directly)

18. **Example:** When you need to perform a specific, repeated calculation that isn't covered by built-in functions, like calculating BMI or converting units.

Section F Answers:

19. "**hello Python!**" (Strips spaces, converts to lowercase, replaces "world" with "Python")

20. "**is f**" (Finds "is" at position 12, takes 4 characters: positions 12-15)

21.

```
sentence = "Apples are awesome and oranges are ok"
words = sentence.split()
count = 0
for word in words:
    if word[0].lower() in "aeiou":
        count += 1
```

Section G Answers:

22 **['yellow', 'red', 'green', 'blue']** (Adds yellow at index 1, moves red to end, sorts reverse alphabetically)

23. **Immutable data**: RGB values shouldn't change accidentally; tuples ensure this

24.

```
squares = [i ** 2 for i in range(1, 6)]
```

25. **matrix[2][2]** or **matrix[-1][-1]**

Section H Answers:

26. "**Not calculated**" and **4** (No "average" key, so get() returns default; grades list has 4 items)

27.

```
random.sample(range(1, 11), 3)
```

28.

```
for key, value in my_dict.items():
    print(key, ":", value)
```

Section I Answers:

29.

a) **Syntax error**

b) **Runtime error**

c) **Syntax error**

30. Improved code:

```
age = 10  # Stores age
is_eligible = "yes"  # Eligibility status

if age > 18:
    print(is_eligible)
```