Python Syntactic Sugar — Practice

CSCI 40: Computing for the Web

Fall 2026

How current LLMs scored on this practice bank *2026-08-26*

Model	Reasoning	Score
claude-haiku-4.5	low effort	37/38 (97%)
claude-sonnet-5	low effort	34/38 (89%)
claude-fable-5	low effort	38/38 (100%)
claude-opus-4.8	low effort	38/38 (100%)
gpt-5.6-luna	low reasoning	30/38 (79%)
gpt-5.4	low reasoning	35/38 (92%)
gpt-5.6-terra	low reasoning	33/38 (87%)
gpt-5.5	low reasoning	38/38 (100%)
gpt-5.6-sol	low reasoning	35/38 (92%)
gemini-3.1-flash-lite	minimal thinking	25/38 (66%)
gemini-3.5-flash-lite	minimal thinking	34/38 (89%)
gemini-3.6-flash	minimal thinking	28/38 (74%)
gemini-3.7-flash	low thinking	37/38 (97%)

One independent, tool-free prompt per question; deterministic executable ground truth.

Python Syntactic Sugar

Note 1. Syntactic sugar is a way for programming languages to make certain common operations more convenient. Python is famous for having a lot of syntactic sugar that can making learning the language hard. All of the "idioms" you will find in this packet do not allow you to write "more powerful" programs, but they will allow you to write programs that have fewer lines of code to accomplish the same task.
You will never be required in this class to use any particular syntactic sugar to accomplish a task. But these idioms are very widely used in python, and so you need to understand them in order to understand the code that other people (or AIs) write.

1 Pattern Matching

Note 2. We can place containers on the left hand side of an equals sign as long as:

1. every element in the container is a variable (`SyntaxError` if violated),
2. the right hand side of the equals is a container (`TypeError` if violated),
3. both containers have the exact same size (`ValueError` if violated).

Problem 3. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
[a, b] = [1, 2]
print('a=', a)
```

Problem 4. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
[a, b, c] = [1, 2]
print('a=', a)
```

Problem 5. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
[a, b] = 12
print('a=', a)
```

Problem 6. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
[a, b] = '12'
print('a=', a)
```

Note 7. Recall that in python, if you have commas separating values without square brackets [] or curly braces {}, then this creates a *tuple*. (Tuples behave like immutable lists.) A common use case for tuples is pattern matching.

Problem 8. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
a, b = [1, 2]
print('a=', a)
```

Problem 9. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
a = 10
b = 5
a, b = b, a
print('a=', a)
```

Problem 10. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
a = 5
b = 10
for i in range(2, 5):
    a, b = b, a
print('a=', a)
```

Note 11. It is common to combine pattern matching with for loops.

Problem 12. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
total = 0
for a, b in [(1, 2), (3, 4)]:
    total += a
print('total=', total)
```

Problem 13. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
total = 0
for a, b in [(1, 2), (3, 4), 5]:
    total += a
print('total=', total)
```

Problem 14. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
total = 0
for a, b in [(1, 2), (3, 4, 5)]:
    total += a
print('total=', total)
```

2 Special Functions

Note 15. There are a number of special built-in functions for making iterating over containers more convenient.

1. The `zip` function combines two or more containers into a single container whose elements are tuples that contain the values of the original containers.

```
>>> list(zip('hello', 'world'))
[('h', 'w'), ('e', 'o'), ('l', 'r'), ('l', 'l'), ('o', 'd')]
>>> list(zip('hello', 'world', 'hola', 'mundo'))
[('h', 'w', 'h', 'm'), ('e', 'o', 'o', 'u'), ('l', 'r', 'l', 'n'), ('l', 'l', 'a', 'd')]
>>> list(zip(range(1000), 'world'))
[(0, 'w'), (1, 'o'), (2, 'r'), (3, 'l'), (4, 'd')]
```

2. The *enumerate* function creates a new container whose entries are tuples with position 0 equal to the index, and position 1 equal to the value in the original container. It is like doing a zip + range.

```
>>> list(enumerate('hello'))
[(0, 'h'), (1, 'e'), (2, 'l'), (3, 'l'), (4, 'o')]
>>> enumerate('hello')
<enumerate object at 0x7f4521f6c220>
```

   (Notice that I am using the `list` function above only to convert the enumerate type into a list type for display.)

Problem 16. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
result = ''
for a, b in zip('hello', 'world'):
    result += a
    result += b
print('result=', result)
```

Problem 17. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
names = ['alice', 'bob', 'charlie', 'dave', 'eve']
accumulator = []
for i,name in enumerate(names):
    text = name + ' is number ' + str(i)
    accumulator.append(text)
print('accumulator[3]=', accumulator[3])
```

Problem 18. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
names = ['alice', 'bob', 'charlie', 'dave', 'eve']
accumulator = []
for i,name in enumerate(names):
    text = 'number ' + str(i) + ' is ' + name
    accumulator.append(text)
print('accumulator[-1]=', accumulator[-1])
```

Problem 19. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = 'python is *weird*'
accumulator = ''
for i,char in enumerate(sentence):
    if char == '*' and sentence[i-1] == ' ':
        accumulator += '_'
    elif char == '*':
        accumulator += '_!'
    else:
        accumulator += char
print('accumulator=', accumulator)
```

3 Comprehensions

Note 20. Any list or dictionary that contains the keyword `for` inside of it is a *comprehension*. It is very easy to get `IndexErrors` or `TypeErrors` inside a list comprehension, and the error messages can be hard to debug. (Expect me to try to trick you on the quiz this way...)

Problem 21. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
names = ['alice', 'bob', 'charlie', 'dave', 'eve']
greetings = ['hello ' + name for name in names]
greeting = greetings[2]
print('greeting=', greeting)
```

Problem 22. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xs = [ x*x for x in range(10) ]
num = xs[5]
print('num=', num)
```

Problem 23. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xs = [ x*x for x in range(10) if x%2 ]
num = xs[3]
print('num=', num)
```

Problem 24. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xs = [ x**x for x in range(-3,3) ]
num = xs[5]
print('num=', num)
```

Problem 25. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xs = [ x for x in range(-3,3) if x ]
num = xs[3]
print('num=', num)
```

Problem 26. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
small_words = [ word.lower() for word in sentence.split() if len(word) <= 2]
print('len(small_words)=', len(small_words))
```

Problem 27. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
small_words = [ word.lower() for word in sentence.split() if word <= 2]
print('len(small_words)=', len(small_words))
```

Problem 28. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
small_words = [ word.lower() for word in sentence if len(word) <= 2]
print('len(small_words)=', len(small_words))
```

Problem 29. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
names = ['alice', 'bob', 'charlie', 'dave', 'eve']
greetings = [ name[0].upper() + name[1:].lower() for name in names ]
greeting = greetings[1]
print('greeting=', greeting)
```

Problem 30. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
names = ['alice', 'bob', 'charlie', 'dave', 'eve']
greetings = [ names[0].upper() + names[1:].lower() for name in names ]
greeting = greetings[1]
print('greeting=', greeting)
```

Problem 31. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
words = [ word.lower() for word in sentence.split() if 't' in word ]
print('len(words)=', len(words))
```

Problem 32. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
xs = [ x.upper() for x in sentence if 't' in x ]
print('xs[1]=', xs[1])
```

Problem 33. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
sentence = "This␣is␣an␣example␣sentence␣with␣a␣few␣words␣in␣it."
small_words = [ word.lower() for word in sentence.split() if len(word) <= 2]
print('len(small_words)=', len(small_words))
```

Problem 34. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) ] for x in [2, 3, 4] ]
x = xss[-1][-2]
print('x=', x)
```

Problem 35. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) ] for x in [2, 3, 4]  if x%2 == 0 ]
x = xss[-1][-2]
print('x=', x)
```

Problem 36. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) ] for x in [2, 3, 4]  if x%2 == 0 ]
x = xss[-1:][-2][1]
print('x=', x)
```

Problem 37. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) ] for x in [2, 3, 4]  if x%2 == 0 ]
x = xss[-1][-2:][1]
print('x=', x)
```

Problem 38. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) if i%3== 1 ] for x in [4, 5, 6]  if x%2 == 1 ]
x = xss[-1][-1]
print('x=', x)
```

Problem 39. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ [ i for i in range(x) if x%3== 1 ] for x in [4, 5, 6]  if x%2 == 1 ]
print('xss=', xss)
```

Problem 40. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ i for x in [2, 3, 4]  if x%2 == 0 for i in range(x) ]
x = xss[-1][-2]
print('x=', x)
```

Problem 41. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ i for x in [4, 5, 6]  if x%2 == 1 for i in range(x) if i%3== 1 ]
x = xss[-1]
print('x=', x)
```

Problem 42. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
xss = [ i  for x in [4, 5, 6]  if x%2 == 1 for i in range(x) if x%3== 1 ]
print('xss=', xss)
```

4 Comprehensions on Twitter Data

Note 43. Comprehensions are commonly used for processing datasets. Expect 1-2 questions on your quiz using a list comprehension on the twitter dataset like below.

Problem 44. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
tweets = [
    { "source": "Twitter␣Web␣Client"
    , "text": "From␣Donald␣Trump:␣Wishing␣everyone␣a␣wonderful␣holiday␣&␣a␣happy,␣healthy,␣prosper
    , "retweet_count": 28
    },
    { "source": "Twitter␣Web␣Client"
    , "text": "Trump␣International␣Tower␣in␣Chicago␣ranked␣6th␣tallest␣building␣in␣world␣by␣Counci
    , "retweet_count": 33
    },
    { "source": "Twitter␣Web␣Client"
    , "text": "Wishing␣you␣and␣yours␣a␣very␣Happy␣and␣Bountiful␣Thanksgiving!"
    , "retweet_count": 13
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "RT␣@realDonaldTrump:␣Happy␣Birthday␣@DonaldJTrumpJr!\nhttps://t.co/uRxyCD3hBz"
    , "retweet_count": 9529
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "Happy␣Birthday␣@DonaldJTrumpJr!\nhttps://t.co/uRxyCD3hBz"
    , "retweet_count": 9529
    },
    { "source": "Twitter␣for␣Android"
    , "text": "Happy␣New␣Year␣to␣all,␣including␣to␣my␣many␣enemies␣and␣those␣who␣have␣fought␣me␣an
    , "retweet_count": 141853
    },
    { "source": "Twitter␣for␣Android"
    , "text": "Russians␣are␣playing␣@CNN␣and␣@NBCNews␣for␣such␣fools␣-␣funny␣to␣watch,␣they␣don't␣
    , "retweet_count": 23213
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "Join␣@AmerIcan32,␣founded␣by␣Hall␣of␣Fame␣legend␣@JimBrownNFL32␣on␣1/19/2017␣in␣Was
    , "retweet_count": 7366
    }]

trump_tweets = [tweet for tweet in tweets if 'trump' in tweet['text'].lower()]
print('len(trump_tweets)=', len(trump_tweets))
```

Problem 45. The following code (circle one)

terminates without error throws an exception runs forever

If the code terminates without error, write the output. If the code throws an exception, state the exception.

```
tweets = [
    { "source": "Twitter␣Web␣Client"
    , "text": "From␣Donald␣Trump:␣Wishing␣everyone␣a␣wonderful␣holiday␣&␣a␣happy,␣healthy,␣prosper
    , "retweet_count": 28
    },
    { "source": "Twitter␣Web␣Client"
    , "text": "Trump␣International␣Tower␣in␣Chicago␣ranked␣6th␣tallest␣building␣in␣world␣by␣Counci
    , "retweet_count": 33
    },
    { "source": "Twitter␣Web␣Client"
    , "text": "Wishing␣you␣and␣yours␣a␣very␣Happy␣and␣Bountiful␣Thanksgiving!"
    , "retweet_count": 13
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "RT␣@realDonaldTrump:␣Happy␣Birthday␣@DonaldJTrumpJr!\nhttps://t.co/uRxyCD3hBz"
    , "retweet_count": 9529
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "Happy␣Birthday␣@DonaldJTrumpJr!\nhttps://t.co/uRxyCD3hBz"
    , "retweet_count": 9529
    },
    { "source": "Twitter␣for␣Android"
    , "text": "Happy␣New␣Year␣to␣all,␣including␣to␣my␣many␣enemies␣and␣those␣who␣have␣fought␣me␣an
    , "retweet_count": 141853
    },
    { "source": "Twitter␣for␣Android"
    , "text": "Russians␣are␣playing␣@CNN␣and␣@NBCNews␣for␣such␣fools␣-␣funny␣to␣watch,␣they␣don't␣
    , "retweet_count": 23213
    },
    { "source": "Twitter␣for␣iPhone"
    , "text": "Join␣@AmerIcan32,␣founded␣by␣Hall␣of␣Fame␣legend␣@JimBrownNFL32␣on␣1/19/2017␣in␣Was
    , "retweet_count": 7366
    }]

popular_tweets = [tweet for tweet in tweets if tweet['retweet_count'] > 100]
print('len(popular_tweets)=', len(popular_tweets))
```