1) What can replace `???` so that the sum of the numbers in `L` is printed?

```
total = 0
for x in range(len(L)):
   ???

print(total)
```

A. `total = x`
B. `total = L[x]`
C. `total += x`
D. `total += L[x]`

2) This program is meant to print the sum of the numbers in `L` (a list of numbers). But unfortunately it has a bug. When the program runs, on what line does it crash?

```
i = 0
total = 0                  # line 1
while i <= len(L):         # line 2
   total = L[i] + total    # line 3
   i = i + 1               # line 4

print(total)
```

A. line 1
B. line 2
C. line 3
D. line 4

3) Which operation **can** be done on a Python string without creating a new string?

A. remove a character
B. insert a character
C. change a character
D. none of the above

4) Which two lines print the same thing?

```
A = True
B = False
C = True
print(not A and (not B))       # line 1
print(C or (not B or A))       # line 2
print(C and not (not B or A))  # line 3
```

A. lines 1 and 2
B. lines 1 and 3
C. lines 2 and 3
D. none of the above

5) This program is meant to print different messages based on the value of the already-defined `int` variable `m`. But unfortunately it has a bug. Which line has an error that causes it not to run?

```
# assume m is an int variable
# that is already defined

if m <= 0:           # line 1
    print("error")   # line 2
elif 0 < m < 10:     # line 3
    print('small')   # line 4
elif:                # line 5
    print('big')     # line 6
```

A. line 1
B. line 3
C. line 5
D. at least one of lines 2, 4, or 6

Programming Questions

For each of the following programming questions:

- Your code will be marked according to correctness, efficiency, and readability.
- Do **not** use Python features not covered in the course (such as list comprehensions, `with` statements, `match` statements, type hints, etc.). Stick to basic Python features as presented in the course.
- Do **not** `import` anything unless the question specifically allows it.

Shortest String

Write a function called `get_shortest_string(words)` that returns (not prints!) the shortest string in `words` (a non-empty list of strings). The `words` list should **not be changed** in any way. If there is a tie for the shortest string, then any one of the shortest can be returned.

For example:

- `get_shortest_string(['mice', 'dog', 'horse', 'cat'])` should return either `'cat'` or `'dog'`
- `get_shortest_string(['hot', '', 'cold])` should return `''` (the empty string)

Do **not** use the built-in functions `min`, `max`, `sort`, or `sorted` in your answer.

Sample Solution

```
def get_shortest_string(words):
    """Assumes words is not empty.
    """
    shortest = words[0]
    for word in words:
        if len(word) < len(shortest):
            shortest = word
    return shortest
```