CSC108H Winter 2024 Worksheet 16 : List Operations and Methods, Function range

1. Consider this code:

```
a = [1, 0]
```

All of the following code fragments cause `a` to refer to `[1, 0, 8]`.

Circle all of the code fragment(s) that create a new list.

(a) `a.append(8)` [✓ (b) `a = a + [8]`]
(c) `a.insert(len(a), 8)` [✓ (d) `a = [a[0], a[1], 8]`]

Circle all of the code fragment(s) that modify the original list.

[✓ (a) `a.append(8)`] (b) `a = a + [8]`
[✓ (c) `a.insert(len(a), 8)`] (d) `a = [a[0], a[1], 8]`

2. Consider this code:

```
a = [1, 0, 8]
b = a.sort()
```

After the code above is executed, which of the following expressions evaluate to `True`? Circle those expression(s).

(a) `a == [1, 0, 8]` [✓ (b) `a == [0, 1, 8]`]

(c) `b == [1, 0, 8]` (d) `b == [0, 1, 8]`

3. Consider this code

```
a = [0, 1, 2]
b = a
b[2] = 100
```

After the code above is executed, which of the following expressions evaluate to `True`? Circle those expression(s).

(a) `a == [0, 1, 2] and b == [0, 1, 100]` (b) `a == [0, 1, 2] and b == [0, 100, 2]`

[✓ (c) `a == [0, 1, 100] and b == [0, 1, 100]`] [✓ (d) `id(a) == id(b)`]

4. Which of the following code fragments **does not** print `'na'` 12 times? Circle those expression(s).

[✓ (a)
```
for i in range(12):
    print('na')
```
]

(b)
```
for i in range(1, 24, 2):
    print('na')
```

[✓ (c)
```
for i in range(1, 12):
    print('na')
```
]

(d)
```
for i in range(6, 12):
    print('na')
    print('na')
```

CSC108H Winter 2024 Worksheet 17 : For Loops Over Lists

For each function, complete the examples in the docstring and then complete the function body.

1.
   ```
   def collect_below_threshold(nums: list[int], threshold: int) -> list[int]:
       """Return a new list consisting of those numbers in nums that are below threshold,
       in the same order as in nums.

       >>> collect_below_threshold([1, 2, 3, 4], 3)
       [1, 2]
       >>> collect_below_threshold([1, 2, 108, 3, 4], 50)
       [1, 2, 3, 4]
       >>> collect_below_threshold([], 7)
       []
       """
        new_nums = []
        for i in nums:
            if i.isdigit() and i < threshold:
                new_nums.append(i)
        return new_nums
   ```

2.
   ```
   def scale_midterm_grades(grades: list[int], multiplier: int, bonus: int) -> None:
       """Modify each grade in grades by multiplying it by multiplier and then
       adding bonus. Cap grades at 100.

       >>> grades = [45, 50, 55, 95]
       >>> scale_midterm_grades(grades, 1, 10)
       >>> grades
       [55, 60, 65, 100]
       """
        for i in range(len(grades)):
            grades[i] = min((grades[i] * multiplier + bonus), 100)
   ```