CL05 - Lists Practice

Announcements

Weekly review session: Sun 4-5 pm in SN 011

Tuesday Recap:

LS 12 Lists

LS13 Relative Reassignment Operators (e.g. i += 1)

LS14 Importing Modules *(don’t need to know mem diagram for quizzes/final)*

LS15 (optional) elif

Lists in Memory

```
1   x: list[str] = ["Comp", "110"]
2   x[1] = "210"
3   y: list[str] = x
4   print(y)
```

Memory Diagrams

```
1   a: str = "24"
2   b: str = a
3   a += "6"
4   print(b)
```

```
1   a: list[int] = [2,4]
2   b: list[int] = a
3   a.append(6)
4   print(b)
```

```python
def remove_first(xs: list[int]):
    xs.pop(0)

course: list[str] = ["comp", "110"]
remove_first(course)
```

```
1   def dup(xs: list[int]):
2       start_len: int = len(xs)
3       i: int = 0
4       while i <= start_len - 1:
5           xs.append(xs[i])
6           i += 1
7
8   groceries: list[str] = ["apples", "eggs"]
9   print(dup(groceries))
10  print(groceries)
```

```python
def dup(xs: list[int]):
    start_len: int = len(xs)
    i: int = 0
    while i <= start_len - 1:
        xs.append(xs[i])
        i += 1

groceries: list[str] = ["apples", "eggs"]
print(dup(groceries))
print(groceries)
```

```python
def odds_list(min: int, max: int) -> list[int]:
    """returns list of odds between min and max"""
    odds: list[int] = list()
    x: int = min
    while x <= max:
        if x % 2 == 1:
            odds.append(x)
        x += 1
    return odds

global_odds: list[int] = odds_list(2,10)
print(global_odds)
```

```python
def odds_list(min: int, max: int) -> list[int]:
    """returns list of odds between min and max"""
    odds: list[int] = list()
    x: int = min
    while x <= max:
        if x % 2 == 1:
            odds.append(x)
        x += 1
    return odds

global_odds: list[int] = odds_list(2,10)
print(global_odds)
```

Coding Example (if we have time)

- Let's implement a function where we can call with 2 arguments:
  - A “needle" int value we are searching for
  - A " haystack " list of values we are searching in

The return value of the function should be True if in the haystack at least once and False otherwise

The name of the function will be contains