import random
length = 20
nums = []
for _ in range(length):
    nums.append(random.randint(0, 100))
squares = []
for i in range(length):
    squares.append(nums[i]**2)

from random import randint
length = 20
nums = [0] * length
for i in range(length):
    nums[i] = randint(0, 100)
squares = [0] * length
for i in range(length):
squares[i] = nums[i]**2
Solutions to Quiz 1

Try it using list comprehensions.

```python
from random import randint
length = 20
nums = [randint(0, 100) for _ in range(length)]
squares = [i**2 for i in nums]
```

Quiz Observations

• `for i in nums:
  squares[i] = i*i`

• `for i in range(length):`  
  ```python
  squares[i] = i*i
  ```

• `nums[20]`

• `for e in squares: e = e**2`

Solutions to Quiz 1

```python
from random import randint
length = 20
nums = [0] * length
for _ in range(length):
    nums.append(randint(0, 100))
squares = [0] * length
for i, e in enumerate(nums):
    squares[i].append(e**2)
```

Cardinal Sin

camelCase
Multiple Lists

You can use multiple existing lists to create the new list:

```python
pairs = [(x, y) for x in xcoords for y in ycoords]
```

What are the elements of `pairs`?
- each is an (x, y) pair in the cross-product of `xcoords` and `ycoords`
- this works even if the two input lists have different lengths

What if you want elements paired by position?
```python
pairs = [(xcoords[i], ycoords[i]) for i in range(len(xcoords))]
```

Can use min if the input lists have different lengths.

Using a function

Let `coords` be a list containing 2-element lists of GPS coordinates:

`[[[lat1, lon1], [lat2, lon2]], [[lat3, lon3], [lat4, lon4]], ...]`

Create a list of distances between the pair of cities in each sublist.

```python
def distance(city1, city2):
    # compute and return distance
dists = [distance(x, y) for x, y in coords]
```

More filtering

Let `data` be a list containing lists of instrument readings:

`[[d0_0, d0_1, ..., d0_n], [d1_0, d1_1, ..., d1_m], ...]`

Create a list of min reading for each sublist.

```python
mins = [min(L) for L in data]
```

Above gives error if a sublist is empty – can’t `min([])`

```python
mins = [min(L) for L in data if len(L) > 0]
```
“Double iteration”

Let data be a list containing lists of instrument readings:

\[
\text{[[d0_0, d0_1, … d0_n], [d1_0, d1_1, … d1_m], …]}\]

Combine all of the elements into a single list:

\[
\text{all_data = [x for L in data for x in L]}\]

If we only want non-negative values:

\[
\text{all_data = [x for L in data for x in L if x >= 0]}\]