Lists and Looping

Introduction to Computer Programming

Dr. Paul Vrbik

October 19, 2018

Looping Over Lists

We can loop over lists in the same manner we did with strings.

```
>>> xs = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’]
>>> for x in xs:
...     print(x)
‘a’
‘b’
‘c’
‘d’
‘e’
```

Question

Write a function that removes all instances of x from a list xs.

```
def remove_x( x, xs:list ) -> list:
    """ Removes all instances of x from the list xs.
    >>> remove(2, [1,2,2,3] )
    [1, 3]
    >>> remove(0, [])
    []
    """
```

Definition (Range)

Python's `range` keyword allows us to quickly build an iterator for use by for-loops.

It has general form:

```
range([start], stop[, step])
```

which is similar to list slicing.

```
>>> range(10)
range(0, 10)

>>> type(range(10))
<class ‘range’>

>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> list( range(2, 7) )
[2, 3, 4, 5, 6]


>>> list(range(2, 7, 3))
[2, 5]


>>> list(range(2, 7, -1))
[]


>>> list(range(7, 2, -1))
[7, 6, 5, 4, 3]
```

Question

Write a function that does not use `range` that generates `range(a, b, c)` for valid inputs of `a`, `b`, `c`.

```
def list_range(a:int, b:int, c:int) -> List[int]:
    """Returns list(range(a,b,c)) without using range.
    Assumes a, b, c are nonzero integers.
    >>> list_range(0, 4, 1)
    [0, 1, 2, 3]
    >>> list_range(3, 11, 4)
    [3, 7]
    """
```

`range` is quite useful when paired with for-loops.

```
>>> xs = [1, 2, 3, 4, 5, 6, 7]
>>> for k in range(0, len(xs), 2):
...     print(xs[k])
```

Question

Write a function which takes two integer lists of equivalent length and returns their dot product.

```
def dot_product(xs:List[int], ys:List[int]) -> int:
    """Returns mathematical dot-product of xs and ys
     taken as vectors.
    Assumes len(xs) == len(ys) > 0
    >>> dot_product([1, 2], [3, 4])
    11
    """
```

Question

Write a function which returns a list of the first k-squares.

```
def squares(k:int) -> List[int]:
    """Returns a list of the first k squares.
    >>> squares(0)
    []
    >>> squares(4)
    [0, 1, 4, 9]
    """
```

List Comprehension

There is set builder notation in Python. It is called list comprehension.

Mathematics

$\{k^2 : k \in \mathbb{N}\} = \{0, 1, 4, \ldots\}$

Python

```
>>> [ k**2 for k in range(4) ]
[0, 1, 4, 9]
```

Question

Write a function which disassembles a string into a list of characters comprising the string.

```
def disassemble(cs:str) -> List(str):
    """Returns a list of characters comprising cs,
    maintaining order.
    >>> disassemble("")
    []
    >>> disassemble("abcdef")
    [‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
    """
```

Nested Lists

A list can have another list as an element

```
>>> xs = [ [1,2], [5,6,7] ]
>>> len(xs)
2
>>> len( xs[-1] )
3
```

Nested Loops

Recall that loops can be nested.

```
>>> X = [ [1, 2], ["a", "b", "c"], "hello" ]
>>> for xs in X:
...    for x in xs:
...        print(x, end=’ ’)
1 2 a b c h e l l o >>>
```

Question

Assume students uniquely identified by integers are broken into teams. Write a function to check that all team members are enrolled students.

Answer

```
def enrolled_teams(teams:List[List[int]],
                   enrolled_students:List[int]) -> bool:
```

Matrices

Notice that a matrix can be represented by a list-of-lists in Python.

$$\mathbb{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \equiv \texttt{[[1, 2, 3], [4, 5, 6], [7, 8, 9]]}$$

```
>>> AA = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
>>> AA[-1]
[7, 8, 9]

>>> AA[1:3]
[[4, 5, 6], [7, 8, 9]]

>>> AA[1]
[4, 5, 6]
>>> AA[1][-1]
6
>>> AA[1,-1]
TypeError: list indices must be integers or slices, not tuple
```

Question

Write a function that fills a $N \times N$ list of lists with the $N \times N$ multiplication table.

```
def mult_table( N:int ) -> List[List[int]]:
    """Fills a N x N list of lists with the N x N
    multiplication table.
    >>> mult_table( 1 )
    [[1]]
    >>> mult_table( 3 )
    [[1, 2, 3], [2, 4, 6], [3, 6, 9]]
    >>> mult_table( 0 )
    []
    """
```

Question

Write a function that returns the matrix transpose of $\mathbb{A}$. Recall

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}^\intercal = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$$

That is the columns and rows are swapped.

Answer

```
def mult(AA::List[List[int]],
         BB::List[List[int]]) -> List[List[int]]:
```

Question

Write a function that returns the matrix product $\mathbb{A} \times \mathbb{B}$ assuming correct dimensions. Recall

$$(\mathbb{A} \times \mathbb{B})_{ij} = \mathrm{row}_i(A) \cdot \mathrm{col}_j(B)$$

```
def mult(AA::List[List[int]],
         BB::List[List[int]]) -> List[List[int]]:
```