Structures: Lists: Basics

CPSC 217: Introduction to Computer Science for Multidisciplinary Studies I
July 2024

Jonathan Hudson, Ph.D.
Instructor
Department of Computer Science
University of Calgary

June 4, 2024

UNIVERSITY OF CALGARY

What is a List?

- A collection of values
  - Values
    - May all have the same type, or
    - May have different types
  - Each item is referred to as an element
  - Each element has an index (ORDERED)
    - Unique integer identifying its position in the list
  - A list is one type of data structure
    - A mechanism for organizing related data

Creating a List

- Format:

```
<list name> = [<value 1>, ..., <value n>]
```

- Examples:

```
names = []
```
→ defines an empty list

```
nums = [10.0, 9.0, 8.5, 5.0, 7.5]
letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g']
names = ['Marc', 'Jim', 'Ken']
mixed = [1.0, 1, "this", True]
```

- By defining the list memory is allocated for it

* Works on Lists?

Repetition Operator (*)

- Just like strings, you can use asterisk to repeat a list

```
>list = [0]*5
```


Produces a list of size 5 with all elements = 0

```
>newList = list*5
```


Produces a new list of size 25 with all elements = 0

Indices

Accessing Elements

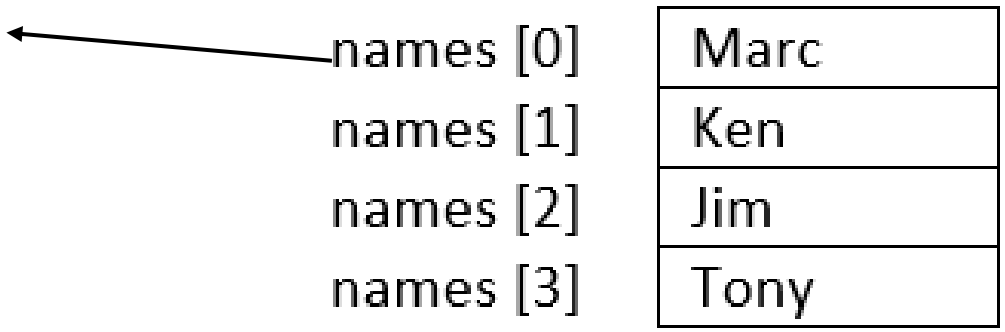
- Each list element has two unique indices, a positive one and a negative one:
  - Positive indices range from 0 to the length of the list minus one (*len(list)-1*)
  - Negative indices range from *-len(list) to –1*

0	1	2	3	4	5	6	7
A	B	C	D	E	F	G	H
-8	-7	-6	-5	-4	-3	-2	-1

Accessing Elements - Accessing a Single Element

- To access one element, use the name of the list, followed by the index of that element in square brackets
  - Use this one element just like any other variable

names[**index**] → returns the value stored at location **index**.



- names refers to the whole list
- len(names)→4
- names.index('Ken') → 1

Loop on List

Accessing Elements - Iterating Over List Items

- A for loop can be used iterates over the list values:

```
stuff = [1, "ICT", 3.14]
for item in stuff:
    print(item)
```

Accessing Elements - Iterating Over List Indices

- Sometimes we need a loop where the control variable varies over the indices rather than the values

```
stuff = [1, "ICT", 3.14]
for i in range(0, len(stuff))
  print(stuff[i])
```

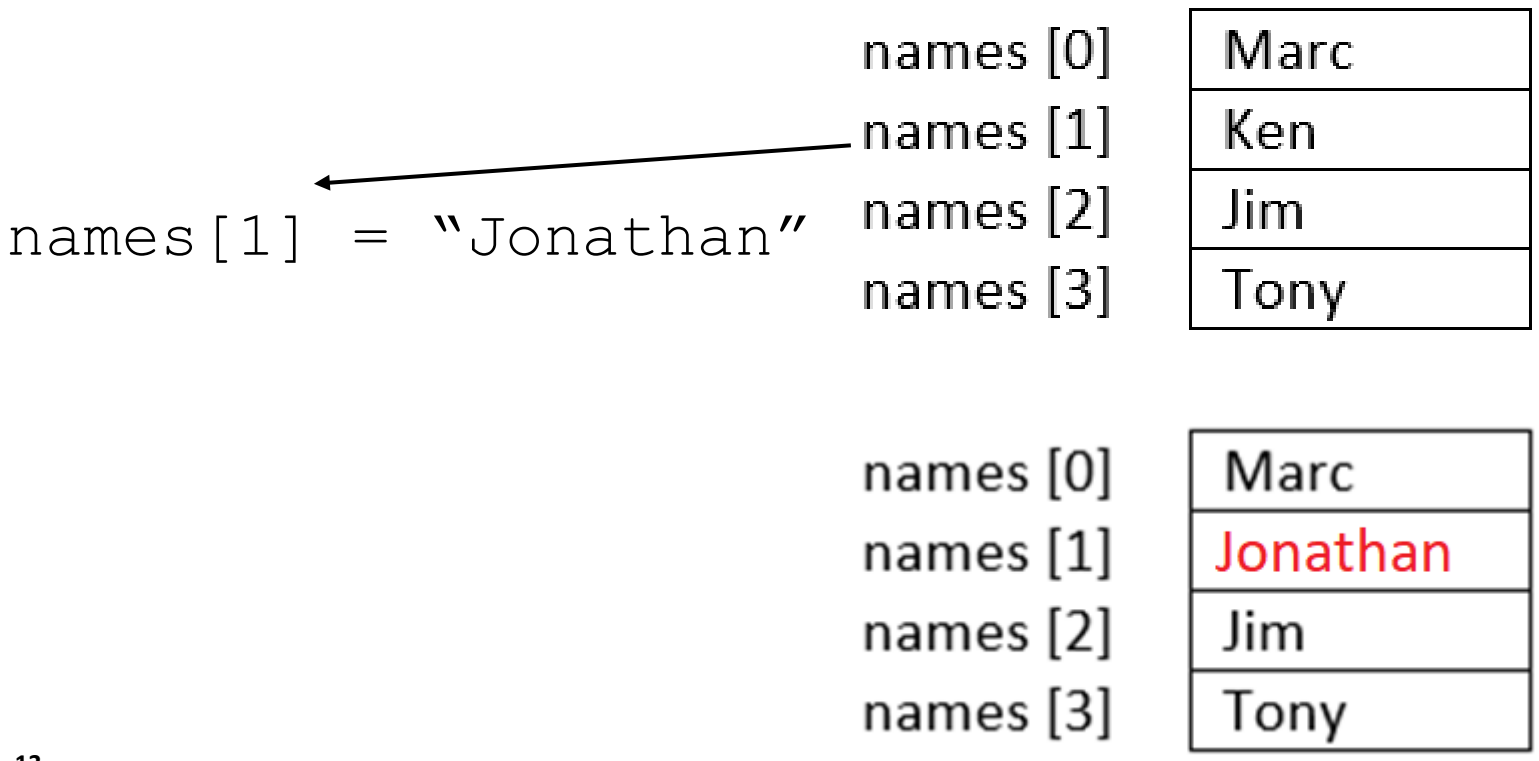
List length changes as elements are added/removed.
So, use *len()* function to determine the length of list.

Modifying List

Modifying Elements

- Lists are mutable, so their elements can be changed as follows:

```
names[index] = new_data
```



Adding Elements

- Lists are mutable, so we can add more elements to them.
- There are three ways to add elements to a list
  - `append(x)`: adds a single element to the end of the list

    ```
    names.append('Daniel')
    ```

  - `insert(i, x)`: inserts a single element into a list at index i, shifts elements at index 3+ up

    ```
    names.insert(3, 'Chris')
    ```

  - `extend(L)`: extends the list by appending the given second list to it

    ```
    names.extend(['Eric', 'Frank'])
    ```

Adding Elements

- Example:

```
names = []

name = input("Enter a name:")
names.append(name)

names_str = [“Joe”,”James”]
names.extend(names_str)

print(names)
```

Printing List

Printing List

- There are many ways to print the content of a list.
- Two common ways are:
  - using *print()*

```
print('names = %s` , (names))
```

- Using a loop → allows us to print the list in a customized format:

```
for i in range(0, len(names), 1):
    print("names[%d] = %s" % (i, names[i]))
```

2D Lists

2D Lists

- A list of lists (images,movies,tables,matrices -> all 2D data)
- [does not have to be rectangular]

A matrix

1	2	3
4	5	6
7	8	9

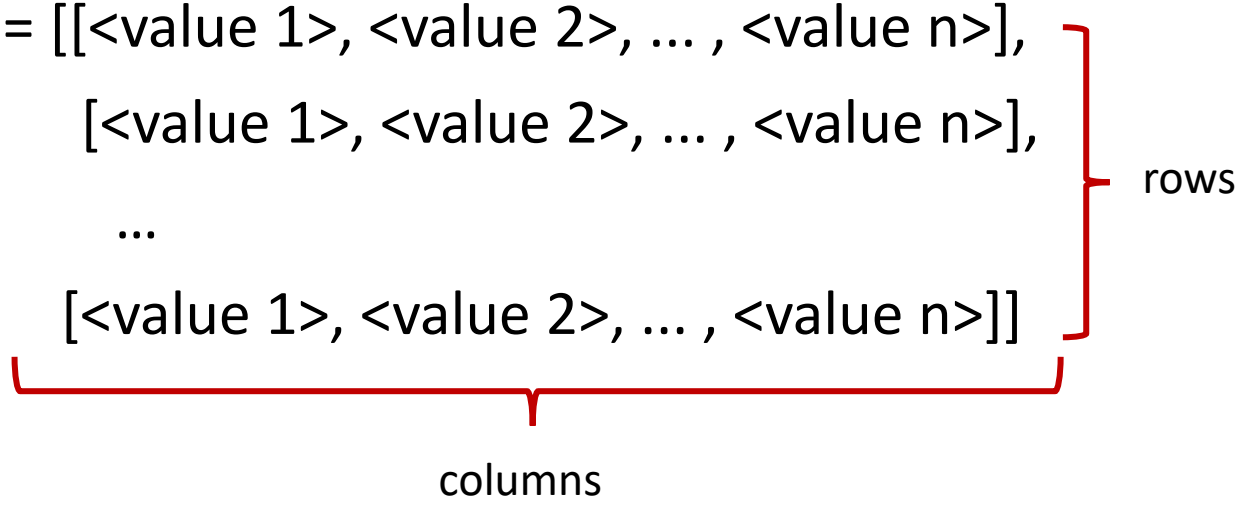
A table

T01	Sandeep Zechariah
T02	Hooman Khosravi
T03	Kanishka Singh
T04	Khobaib Zaamout

2D Lists

- Format:

<list name> = [[<value 1>, <value 2>, ... , <value n>],
[<value 1>, <value 2>, ... , <value n>],
...
[<value 1>, <value 2>, ... , <value n>]]



Accessing 2D Lists

```
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
print(matrix1)
print(matrix1[0])
print(matrix1[1])
print(matrix1[2])
```

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

```
row = matrix1[0]
print(row[0])
print(row[1])
print(row[2])
```

```
1
2
3
```

Accessing 2D Lists

```
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
print(matrix1[0][0])
print(matrix1[0][1])
print(matrix1[0][2])
```


1

2

3

2D List: Example

Example: Boggle

- Generate a random board for Boggle
  - 4x4 board
  - Store the board in a 2D list
  - Each space on the board contains one randomly selected letter
  - Display the board
  - Sample Board:

S	N	K	O
V	R	E	R
I	D	I	N
N	E	G	U

Example: Boggle

```
from pprint import pprint
from random import choice

NUM_ROWS = 4
NUM_COLS = 4

board = [] # Create a new, empty board
for row in range(NUM_ROWS): # Add the correct number of rows to the board
  board.append([""]* NUM_COLS) # Append a row of size NUM_COLS

pprint(board) #pretty print the board

# Set each element in the board to a random letter
for row in range(NUM_ROWS):
  for col in range(NUM_COLS):
    board[row][col] = choice("ABCDEFGHIJKLMNOPQRSTUVWXYZ")

pprint(board) # Pretty print the board
```

2D-List Creation

- Creating the following matrix programmatically:

```
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

2D-List Creation

- Creating the following matrix programmatically:

```
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

```
matrix1 = []
for i in range (1, 10, 3):
        row = [i , i + 1, i + 2]
        matrix1.append(row)

print(matrix1)
```

```
matrix2=[]

ROWS=3
COLS=3

for row in range(ROWS):
    matrix2.append([])
    for col in range(COLS):
        matrix2[row].append(counter)

print(matrix2)
```

2D-List Printing

- Using *print (matrix)*
- Using loops:

```
for row in matrix:
    print(row)
```

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

Print one row per iteration

Print one row per iteration

```
for row in matrix:
    output = ''
    for num in row:
        output += str(num) + ' '
    print (output)
```

```
1 2 3
4 5 6
7 8 9
```

Onward to … more complicated lists.

Jonathan Hudson
jwhudson@ucalgary.ca
https://cspages.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY