COMP 110

Introduction to Lists

Lists

Examples of lists:

- To-do list
- Assignment Due Dates
- Grocery List

A list is a **data structure**–something that lets you reason about multiple items.

***Lists can be an arbitrary (but finite) length! (Not a fixed number of items.)*

Declaring the type of a list

```
<list name>: list[<item type>]

grocery_list: list[str]
```

Declaring the type of a list

```
<list name>: list[<item type>]

grocery_list: list[str]
```

str, int, float, etc.

Initializing an empty list

With a constructor:

The constructor **list()** is a *function* that returns the literal **[]**

- <list name>: list[<item type>] = list()
- grocery_list: list[str] = list()

With a literal:

- <list name>: list[<item type>] = []
- $\underbrace{\text{grocery\_list: list[str]}}_{\text{declare variable}} = \underbrace{\text{[]}}_{\text{initialize list}}$

"create a var called grocery_list, a list of strings, which will initially be empty"

Initializing an empty list

With a constructor:

- <list name>: list[<item type>] = list()
- grocery_list: list[str] = list()

With a literal:

- <list name>: list[<item type>] = []
- grocery_list: list[str] = []

The constructor **list()** is a *function* that returns the literal **[]**

Bringing it back to something we know, you can create an empty string using the constructor **str()** or the literal **“”**

Initializing an empty list

With a constructor:

- <list name>: list[<item type>] = list()
- grocery_list: list[str] = list()

With a literal:

- <list name>: list[<item type>] = []
- grocery_list: list[str] = []

The constructor **list()** is a *function* that returns the literal **[]**

Bringing it back to something we know, you can create an empty string using the constructor **str()** or the literal **“”**

Let’s try it!
Create an empty list of floats with the name my_numbers.

Adding an item to a list

```
<list name>.append(<item>)

grocery_list.append(“bananas”)
```

Adding an item to a list

<list name>.append(<item>)

grocery_list.append(“bananas”)

- Method: a function that *belongs* to the **list** class
- Like calling append(grocery_list, “bananas”)

Adding an item to a list

```
<list name>.append(<item>)

grocery_list.append("bananas")
```

- Method: a function that *belongs* to the **list** class
- Like calling append(grocery_list, "bananas")

Let's try it!

Add the value 1.5 to my_numbers.

Initializing An Already Populated List

```
<list name>: list[<item type>]  = [<item 0>, <item 1>, … , <item n>]

grocery_list: list[str] = [“bananas”, “milk”, “bread”]
```

Initializing An Already Populated List

<list name>: list[<item type>] = [<item 0>, <item 1>, … , <item n>]

grocery_list: list[str] = [“bananas”, “milk”, “bread”]

Let’s try it!
Create a list called game_points that stores the following numbers: 102, 86, 94

Indexing

grocery_list: list[str] = [“bananas”, “milk”, “bread”]

grocery_list[0]

***Starts at 0, like with strings!*

Indexing

grocery_list: list[str] = [“bananas”, “milk”, “bread”]

grocery_list[0]

***Starts at 0, like with strings!*

Let’s try it!
In game_points, use subscription notation to print out 94.

Modifying by Index

```
grocery_list: list[str] = [“bananas”, “milk”, “bread”]

grocery_list[1] = “eggs”
```

Modifying by Index

```
grocery_list: list[str] = [“bananas”, “milk”, “bread”]

grocery_list[1] = “eggs”
```

Let’s try it!
In game_points, use subscription notation to change 86 to 72.

Modifying by Index

Let’s try it!
In game_points, use subscription notation to change 86 to 72.

```
grocery_list: list[str] = [“bananas”, “milk”, “bread”]

grocery_list[1] = “eggs”
```

Question: Could you do this type of modification with a string? Try it out!

Length of a List

```
grocery_list: list[str] = [“eggs”, “milk”, “bread”]

len(grocery_list)
```

Length of a List

```
grocery_list: list[str] = [“eggs”, “milk”, “bread”]

len(grocery_list)
```

Let’s try it!
Print the length of game_points.

Remove an Item From a List – “pop off!”

grocery_list: list[str] = [“eggs”, “milk”, “bread”]

grocery_list.pop(2)

Index of item you want to remove

Remove an Item From a List

grocery_list: list[str] = [“eggs”, “milk”, “bread”]

grocery_list.pop(2)

Index of item you want to remove

Let’s try it!
Remove 72 from game_points.