Lists and Strings

George Mason University
Today’s topics

• Review of Chapter 5: Lists and Strings
• Go over examples and questions
• lists and strings in Python
List (and strings) review

• How do we declare a list? An empty list?
• What operations can we apply to lists?
• How are lists represented in memory?
  – primitive types
  – complex types
• What happens when we assign a variable to a list? (in memory)
• What is a character?
• How is a string similar to a list? Different?
• What is the substring operation?
Let’s go over the exercises
Lists in python

```python
def function():
    hybrids = ["puggle", "liger", "mule"]
    hybrids.append("beefalo")
    hybrids[1] = "wolf dog"

print function()
```

- `.append()` is a way to add a single element to the end of a list
length in python

```python
def function():
    #YOUR CODE GOES BELOW HERE, MAKE SURE YOU INDENT!
    hybrids = ["puggle", "ligier", "mule"]
    counter = 0
    while counter < len(hybrids):
        print hybrids[counter]
        counter = counter + 1

    #DO NOT WRITE CODE BELOW THIS LINE

print function()
```

• `len( )` is a function to return the length of a list or string
python memory visualizer

• let’s use the python memory visualizer to trace through an example:

http://pythontutor.com/visualize.html#code=x+%3D+1%0Asmall+%3D+%5Bx,+2,+3%5D%0Amedium+%3D+%5B+43,+76,+180,+57%5D%0Alarge+%3D+%5B234%5D%0AbigList+%3D+%5B+medium,+small,+large%5D%0Aprint+small%0Aprint+bigList%0Asmall%5B0%5D+%3D+-1%0Aprint+small%0Aprint+bigList%0Ax+%3D+7%0Aprint

Note the difference between the storage of x, a primitive, and the lists.
Debugging with `id( )`

```python
def function():
    #YOUR CODE GOES BELOW HERE, MAKE SURE YOU INDENT!
    small = [1, 2, 3]
    medium = [43, 76, 180, 57]
    large = [234]
    bigList = [medium, small, large]
    print "small: " + str(small)
    print "small memory address: " + str(id(small))
    print "bigList: " + str(bigList)
    counter = 0
    while counter < len(bigList):
        print "bigList["+str(counter)+"] memory address: " + str(id(bigList[counter]))
        counter = counter + 1
    print "bigList: memory address" + str(id(bigList))
    #DO NOT WRITE CODE BELOW THIS LINE

print function()
```

• `id( )` can be used to print out the memory address of a complex type

• `str( )` is needed to add non-strings to strings
Debugging with \texttt{id()}
Note you’ll need to check if the element is in the list or string before trying to get its index.
• *in* is a keyword and can be used to check if the element is in the list or string before trying to get its index
substrings and sub-lists with [: :]

```python
def function():
    # YOUR CODE GOES BELOW HERE, MAKE SURE YOU INDENT!
    string = "CS112 Intro to programming"
    print(string)
    print(string[7:15])

    list = [5, 7, 9, 11, 15]
    print(list)
    print(list[1:4])
    print(list[1:])
    print(list[:4])
    print(list[::])

    print(function())
```

```
Drs-MacBook-Air:cs112 drcica$ python template.py
CS112 Intro to programming
Intro to
[5, 7, 9, 11, 15]
[7, 9, 11]
[7, 9, 11, 15]
[5, 7, 9, 11]
[5, 7, 9, 11]
Drs-MacBook-Air:cs112 drcica$
```
def function():
    # YOUR CODE GOES BELOW HERE, MAKE SURE YOU INDENT!
    string = "CS112:\n\tIntro to programming"
    print string

print function()
Other useful functions

- \texttt{min(list)}
- \texttt{max(list)}
- \texttt{list.remove(1.5)}
- \texttt{sortedList = sorted(list)}
Strings in Python

• ‘cat’ and “cat” are the same
  – can use single or double quote
  – default is single quote
  – useful for “ ’ ” versus ‘ ” ’
Questions?