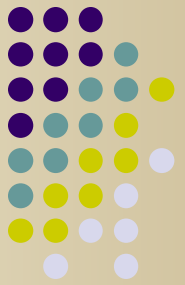
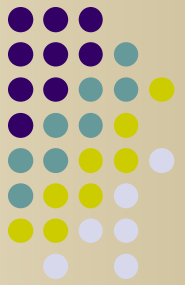
ENGR/CS 101 CS Session Lecture 12



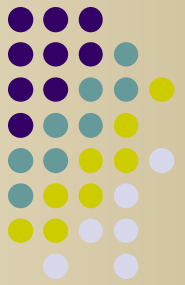
- Log into Windows/ACENET (reboot if in Linux)
- Use web browser to go to session webpage
  http://csserver.evansville.edu/~hwang/f14-courses/cs101.html
- Right-click on lecture12.py link. Save link/target to folder where your other CS 101 programs are.
- Right-click on numbers.dat link. Save link/target to same folder.
- Start Python, open lecture12.py file

Outline



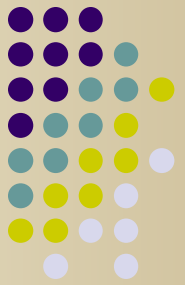
- Questions about Homework 2?
- Arrays
  - Python lists
- Problem: Finding the range of a collection of numbers
  - Finding the maximum value
  - Finding the minimum value

Storing Collections



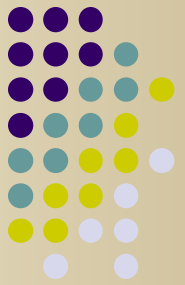
- Last class we discussed searching and sorting algorithms. Today we will discuss how the data storage might be arranged.
- First, we want the data to be stored in a collection that is arranged to allow us easy access to each element.
- Also, we want to be able to identify an element of the collection by using the position (or index) of the element in the collection.

Arrays



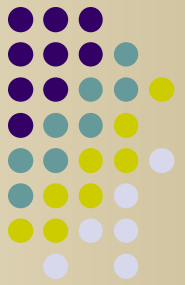
- The simplest collection in most programming languages is the ***array***.
- Formally, an array is an ***ordered*** collection of ***homogeneous*** elements that is accessed by an ***index***.

Arrays



- Ordered means that the relative position of each element matters.
- Homogeneous means all elements are of the same type
- Indexed means that we identify positions using an integer.

Arrays



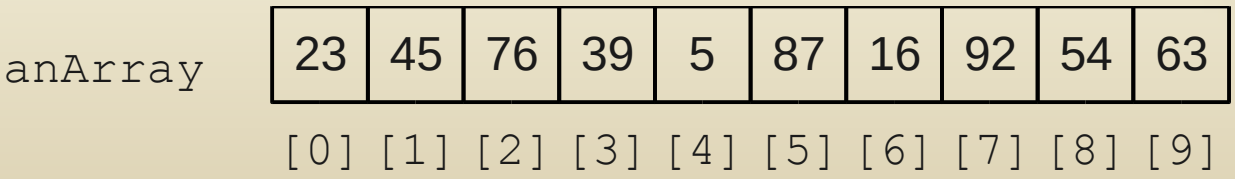
- To access an array element, you give the array name and the index of the element. Almost all programming languages (including Python) use:

  `<name>[<index>]`

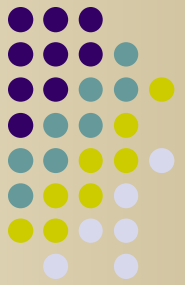
- This is a compound name that can be used on either side of an assignment.

Arrays

- The indexes of an array usually start at 0 or 1. For technical reasons, most modern programming languages (including Python) start indexing at 0.
- Here is a picture of an array of 10 integers:

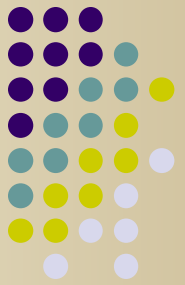


Python Lists



- In Python, the basic collection is called a ***list***.
- Like an array, a Python list is ordered and indexed. Unlike an array, the elements are ***heterogeneous***. I.e., they may be of different types.

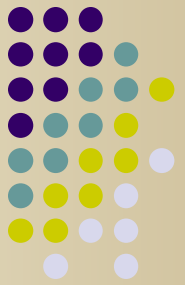
Python Lists



- A list is written as comma-delimited values enclosed in square brackets. A list with no elements is called the ***empty list***.

```
>>> mylist = [3, "cat", 4.5, True]
>>> mylist
[3, "cat", 4.5, True]
>>> mylist[2]
4.5
>>> mylist[2] = 6.5
>>> mylist
[3, "cat", 6.5, True]
```

Python Lists



- Python lists have a number of built-in operations including

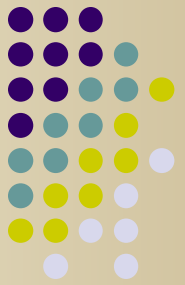
  `+` , concatenate two lists together

  * , concatenate a repeated number of times

  `len()` , function that returns the number of elements in the list

  `[<startIdx>:<endIdx>]` , slicing operation for extracting a part of a list starting at index **`<startIdx>`** and ending at index **`<endIdx>-1`**.

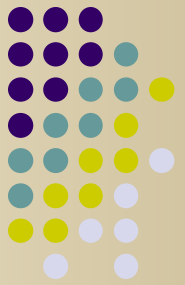
Python Lists



- Here are some examples

```
>>> mylist+mylist
[1, 'cat', 6.5, True, 1, 'cat', 6.5, True]
>>> mylist*3
[1, 'cat', 6.5, True, 1, 'cat', 6.5, True,
1, 'cat', 6.5, True]
>>> len(mylist)
4
>>> len (mylist*4)
16
>>> mylist[1:2]
['cat']
```

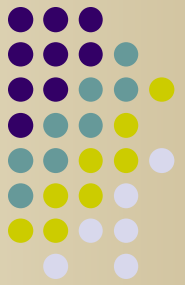
Python Lists



- It turns out that Python also has built-in ways of searching a list (**`in`** operation) and sorting a list (**`sort`** method).

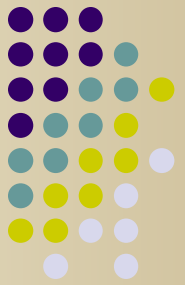
```
>>> mylist = [12, 4, 63, 36, 28]
>>> 36 in mylist
True
>>> mylist.sort()
>>> mylist
[4, 12, 28, 36, 63]
```

Problem: Finding the Range of a Collection of Numbers



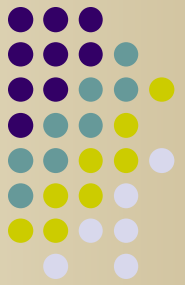
- ***Dispersion*** is a measure of how spread out a collection of data values are. The simplest measure is to find the ***range*** of the data values, which is the difference between the maximum and minimum values of the collection.
- We would like a program that given a file of numbers, computes and displays the range of the numbers stored in the file.

Main Program Design



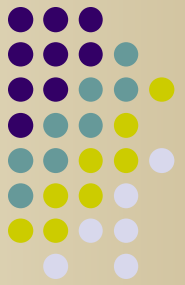
- Design
  1. Create an empty list
  2. Ask the user for the name of a data file
  3. Open the file for reading
  4. For each line in the input file
     a. Convert the line to an integer
     b. Append the integer to the list
  5. Compute and display the range of the data in the list

In-class Program



- **`lecture12.py`** is the start of program we will use to show how to compute various statistical quantities on a collection of random numbers.
- The provided code handles asking the user for a file name, opening the file and accessing each line. (Steps 2-4)
  - The **`pass`** statement is used as a placeholder for code bodies. This allows the file to be compiled.

Creating a Python List



- In order to compute statistics on the data, we need to store the data into a list.
- First we create an empty list.

```
items = []
```

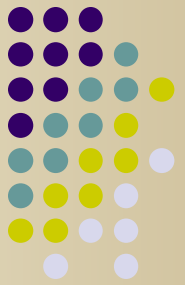
- Then in the loop, we convert each line into an integer

```
number = int(line)
```

- And we append each number to the list

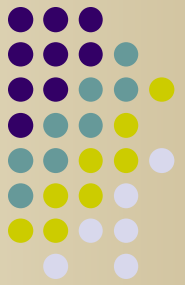
```
items.append(number)
```

Finding the Maximum Value



- The first thing we need is to find the maximum value in the list. We will write a function that does this.
- Analysis
  - Received: list to be searched
  - Returned: maximum value

Finding the Maximum Value

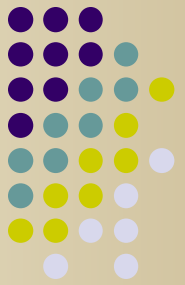


- The basic idea is to go through the list keeping track of the current maximum value.
- Start by assuming the first element is the maximum value.
- Use a for-loop and an if-statement to check the rest of the elements.

findMax Function Code

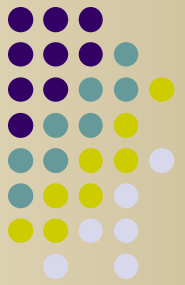
```
def findMax (aList):
    # assume first item is max
    maxSoFar = aList[0]

    # look at the rest of the items
    for item in aList[1:]:
        if item > maxSoFar:
            # a new max
            maxSoFar = item
    return maxSoFar
```

In-class Exercise



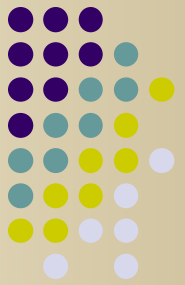
- In the main program, write a function call to **`findMax`** using the **`items`** list and display the result. Run the program and make sure you are getting the correct answer, then delete this code.
- Write a function **`findMin`** that finds and returns the minimum value in a received list. Hint: how does this differ from finding the maximum value?

In-class Exercise



- In the main program, write a function call to **`findMin`** using the **`items`** list and display the result. Again, make sure you are getting the correct answer, then delete the code.
- Write a function **`computeRange`** that receives a list, computes the range of the values using the **`findMax`** and **`findMin`** functions, and returns the range of the values.

In-class Exercise



- Finish the main program by calling **`computeRange`** and displaying the range of the data values.