ECS 10

11/16

Announcements

- MIDTERM Mon Nov 19
- Special Office Hours Mon, 10-12, 3015 Kemper
- Open book, open notes. Bring sample programs from class, your programs, etc.
- Bring a Scantron 2000
- On myUCDavis: solutions to Prog 4, programming problem on sample midterm.
- Wds Nov 21 will be review, no new material.

Topics

- Lists, indexing, lists of lists
- for loops
- Functions, local and global variables
- Files
- Slicing strings or lists
- String methods (split, replace, strip)
- Exceptions
- Everything from Midterm 1

Getting Input

```
choosing = True
while choosing:
    a = raw_input("Enter h or t: ")
    choosing = not (a == "h" or a == "t")
print "You chose ",a
```

- **choosing** is a Boolean variable.
- **while** loop runs so long as **choosing** is **True**
- Gets set to **False** if **a=="h" or a=="t"**

List Indexing

```
L = [["D",245],["R",342],["L",43],["P",12]]
newVotes = ["D",13]
for pair in L:
    if pair[0] == newVotes[0]:
        pair[1] = pair[1]+newVotes [1]
print L
```

- **L** is a list of lists.
- **pair** becomes each party-total pair in turn.
- Adds new votes into correct total

List of Strings

```
L = []
for i in range(0,9):
        L = L + [str(i)]
print L
```

- **L** begins as the empty list
- Function **range(0,9)** has the value [0,1,2,…8,9]
- Function **str(i)** converts integer **i** to a string
- We have to concatenate two lists, so **[str(i)]**

Function

```
def removeComma(s):
        sOut = ""
        for char in s:
        if char != ",":
              sOut = sOut+char
        s = "Gone"
        return sOut
# Main program - s begins as undefined
strIn = raw_input("Please give input: ")
newStr = removeComma(strIn)
```

Arguments – Input to Function

- def removeComma(s):
  - s is the parameter or argument
  - Takes on values of expressions in parenthesis when the function is called
- newStr = removeComma(strIn)
  - Here, it will be the value of inStr, which holds the input string the user typed

Return Value – Output of Function

- return sOut
  - Values following the return statement are returned to the main program
- newStr = removeComma(strIn)
  - Here, the value of sOut is returned by the function, and gets assigned to the variable newStr in the main program

Local Variables

```
# A program that crashes on the last line
def summer(n):
   total = 0
   outLine = ""
   for i in range(n):
      total = total + i
   return total
# Main program
print "Function returns",summer(5)
print total # Try to print local variable
```

Local Variables

- Any variables first defined in a function are local and are NOT defined in the main program.
- Function arguments are local.
- It is possible to have two variables in a program with the same name, one in a function and one in the main program. This is VERY CONFUSING; don't do it. Make up all new variable names for your functions.

Files

```
# Basic file reading loop
inFile = open("myFile.txt","r")
giftStr = inFile.readline()
while giftStr != "":
   print "Read line: ",giftStr
   giftStr = inFile.readline()
```

- Files are read from beginning to end; no going backwards.
- readline() method returns next line.
- .txt or .csv files contain only strings!

Slicing

```
# Example of slicing
string = "pineapple"
print string[-5:]
squareList = [1,4,9,25]
print squareList[1:-2]
```

- Prints “apple” and then “[4]”

String Processing

```
string = "    Nov 16\tNASDAQ\t 2634.93\t+0.63%\n"
string = string.strip()
words = string.split("\t")
change = words[-1]
if change[0] == "+":
   direction = "up"
else:
   direction = "down"
change = change[1:]
change = change.replace("%","")
print "NASDAQ ",direction,"by",change,"percent"
```

String Methods

- **string = string.strip()**
  - **Removes leading and trailing whitespace**
- **words = string.split("\t")**
  - **Returns list of substrings**
- **change = change.replace("%","")**
  - **Replaces all copies of one substring with another**

Exceptions

```
def isFloat(s):
   try:
        float(s)         # Try to do the conversion
   except:
        return False  # Conversion failed!
   else:
        return True   # Conversion succeeded
```