Chapter 2: Goals

- **CS Goals:**
  - Learn about JES. Try it out ✓
  - Create and use variables to store:
    - Values (e.g., how old you are)
    - Objects (e.g., pictures and sounds)
  - Create Functions ✓
  - Understand about different types (encodings) of data ✓

- **Media Goals:**
  - Make and show pictures
  - Make and play sounds

Problem with our functions so far

- Our old version of compute cube:
  ```python
  def computeCube (num):
      answer = num * num * num
      print answer
  ```

- But what if you want to print it differently sometimes
  ```python
  def computeCubePretty (num):
      answer = num * num * num
      print num, "cubed is", answer
  ```
  ```python
  def computeCubePretty (num):
      answer = num * num * num
      print num, "cubed is", answer
  ```

And if you need more versions???

```python
def computeCube (num):
    answer = num * num * num
    print answer
```

```python
def computeCubePretty (num):
    answer = num * num * num
    print num, "cubed is", answer
```

```python
def computeCubeWeird (num):
    answer = num * num * num
    print "wow – that is", answer
```

A better way:
Functions that “return” a value

```python
def computeCube (num):
    octopus = num * num * num
    return octopus
```

How to Trace a Function
1. Get a new piece of scrap paper – write the name of the function at the top
2. Add boxes for the arguments (if there are any)
3. Fill in the values for the arguments
4. Run the function
5. Return as necessary
6. Throw out the scrap paper

Practice!
- Write a function that returns the sum of its three arguments
- Write a function that returns the area of a rectangle given its base and height

NOTE: This slide won’t make sense in print –
Go watch the video!!!
Surprise – Functions in Functions!

```python
def sumOfThree(one, two, three):
    total = one + two + three
    return total

def sumOfSix(a, b, c, d, e, f):
    firstHalf = sumOfThree(a, b, c)
    secondHalf = sumOfThree(d, e, f)
    total = firstHalf + secondHalf
    print "Sum of 6 is: ", total
```

Special boxes for special types

- We already know about three types
  - ints, floats, and strings
- When we trace a program and create a new variable for one of these types, we make a normal box
- Each box holds exactly one thing

Special boxes for special types

- Let’s pretend that we have some more special boxes:
  - A box that holds a date
  - A box that holds a time
  - A box that holds a picture
- What’s special about these boxes?
  - A date has 3 parts: month, day, year
  - A time has 4 parts: hour, minute, seconds, am or pm
  - A picture has a lot of parts: height, width, all the pixels
Special types are similar to regular types

• Each box can only hold one thing at a time
• It’s just that thing may have a bunch of parts
• (Pretend) Example: Date box
  • Only holds one date
  • But that’s made up of 3 parts
  >>> birthday = 2/4/1980
  >>> firstDayOfClass = 1/22/2013
• How should we store these dates?

Pretend Example (continued)

• Each box can only hold one thing at a time
• It’s just that thing may have a bunch of parts
• (Pretend) Example: Date box
  • Only holds one date
  • But that’s made up of 3 parts
  >>> birthday = 2/4/1980
  >>> firstDayOfClass = 1/22/2013
• How should we store these dates?

Command area

name (str)
“Mary”

>>> name = “Mary”
>>> birthday = 2/4/1980
>>> firstDayOfClass = 1/22/2013

>>> print birthday
02/04/1980
>>> print firstDayOfClass
01/22/2013

>>> firstDayOfClass = 9/3/2013
>>> print firstDayOfClass
09/03/2013

NOTE: This slide won’t make sense in print –
Go watch the video!!!

Thinking about functions & types

• We now have to pay attention to
  • The types of values we use as arguments our functions
  • The types of values that are returned from our functions!!

```python
def computeCube (num):
    cube = num * num * num
    return cube

def tripler (thing):
    myTriple = thing * 3
    return myTriple

def niceDay ():
    return “Have a nice day!”

def myPi ():
    return 3.14159265358979
```

int or float or string
Jython has some built-in functions that may be useful

```python
>>>(letterVal = ord("B")
66
>>> x = abs(letterVal)
>>> y = abs(-25)
>>> print "x is" x, "and y is", y
x is 66 and y is 25
```

A function that will be very useful!

```python
>>> myPicture = pickAFile()
```

- What do we know so far about pickAFile()?
  - It's a function
  - It returns a value
- More info you wouldn't know if someone didn't tell you:
  - When you run pickAFile, it pops up a file picker
  - It returns a string that is the full filename of your file
  - The full filename includes the path and the name

```
pickAFile example
```
Jython & Pictures (wooo hoo!!)

- Jython will let you ...
  - Show a picture on the screen
  - Change parts of a picture
  - Lots of other stuff...
- It’s a computer so ... It’s stupid
  - You need to tell it all the steps it has to do
  - At first this will seem like a lot of steps, but I promise it will feel OK in a little while

How to display a picture

1. Make sure you have a (jpg) picture on your hard drive (or flash drive or h drive)
2. Figure out the path to your file (hey, we know how to do that!)
3. Use another function to tell Jython to grab the file at that location and put it into a special picture object (like our boxes for floats, ints, and strings, but more special)
4. Use another function that tells Jython to display the special picture object on the screen

A function for picking & showing picture files

```python
def pickAndShow():
    myfile = pickAFile()
    mypict = makePicture(myfile)
    explore(mypict)
```

NOTE: This slide won’t make sense in print – Go watch the video!!!
Displaying pictures from the command area

- Feel free to use your own picture and your own names

```python
>>> turkey = pickAFile()
>>> turkeyPic = makePicture(turkey)
>>> show(turkeyPic)
```

- Reminder – In my example:
  - `turkey` is the name I use for my file
  - `turkeyPic` is the name I use for my picture
  - I can use these names for pictures of elephants too (it's just a bad choice of name!)

Summary: Picture Functions

- `makePicture(filename)` creates and returns a picture object, from the JPEG file at the filename
- `explore(picture)` displays a picture in a window
- `show(picture)` also displays a picture in a window
- We’ll learn functions for manipulating pictures later, like `getColor()`, `setColor()`, and `repaint()`

Summary Sound Functions

- `makeSound(filename)` creates and returns a sound object, from the WAV file at the filename
- `explore(sound)` allows you to play all or some of the sound
- `play(sound)` makes the sound play (but doesn’t wait until it’s done)
- `blockingPlay(sound)` waits for the sound to finish
- We’ll learn more later like `getSample()` and `setSample()`

JES Review:
Blocking is indicated for you in JES

- Statements that are indented the same, are in the same block.
- Statements that are in the same block as where the line where the cursor is are enclosed in a blue box.
The Most Common JES Bug: Forgetting to Load

- Your function does NOT exist for JES until you load it
  - Before you load it, the program is just a bunch of characters.
  - Loading encodes it as an executable function
- Save and Save As
  - You must Save before Loading
  - You must Load before you can use your function

An “Unloaded” function doesn’t exist yet.

Notes me vs. book

- Book uses `show()` function, I use `explore()`
- Book uses `play()` function, I use `explore()`
- Try them both!

Book’s recipe for playing picked sound files

```python
def pickAndPlay():
    myfile = pickAFile()
    mysound = makeSound(myfile)
    play(mysound)
```

Note: `myfile` and `mysound`, inside `pickAndPlay()`, are completely different from the same names in the command area.

Book’s function for picking & showing picture files

```python
def pickAndShow():
    myfile = pickAFile()
    mypict = makePicture(myfile)
    show(mypict)
```
What if you forget your variable names in command area? showVars()

A function for a specific sound or picture

```python
def playSound():
    myfile = "FILENAME"
    mysound = makeSound(myfile)
    play(mysound)

def showPicture():
    myfile = "FILENAME"
    mypic = makePicture(myfile)
    show(mypic)
```

Put r in front of Windows filenames:
`r"C:\mediasources\pic.jpg"`
or use forward slashes
`"C:/mediasources/pic.jpg"`

What to do about Windows filenames?
- Python doesn’t like you to use “\” in filenames, like "C:\mediasources\barbara.jpg"
- What to do?
  - Option #1: Put r in front of Windows filenames: `r"C:\mediasources\pic.jpg"`
  - Option #2: Use forward slashes. Python will translate it for you: `"C:/mediasources/pic.jpg"

JES Review: What can go wrong?
- Did you use the exact same names (case, spelling)?
- All the lines in the block must be indented, and indented the same amount.
- Variables in the command area don’t exist in your functions, and variables in your functions don’t exist in the command area.
- The computer is stupid
  - It will only do exactly what you tell it to do
Demonstrating using Pictures

```python
>>> myfile = pickAFile()
>>> print myfile
H:\classes\intro sci f10\images\turkey.jpg
>>> mypicture = makePicture(myfile)
>>> print mypicture
Picture, filename H:\classes\intro sci f10\images\turkey.jpg height 144 width 144
>>> show(mypicture)
```
What you should do between classes

• Go through all the slides again and DO THE EXAMPLES!
• Go through the book and DO THE EXAMPLES!
• Do the reading again and DO THE EXAMPLES AGAIN!
• Change the examples slightly and TRY THEM AGAIN
• Do whatever other homework I assign
• Actually do (and make sure you understand) the reading I assign

• Problems??
  • ASK QUESTIONS!! (Class, office hours, appointment)

Pictures for JES Have to be SMALL!!

• Use any program you like to shrink pictures BEFORE you put them into JES
• Don’t have software? See class web site for how to download GIMP (free!)