

Programming: Part I

In this section of notes you will learn how to write simple Python programs using JES.

James Tam

How To Improve Your Program Writing Skills

Practice things yourself.

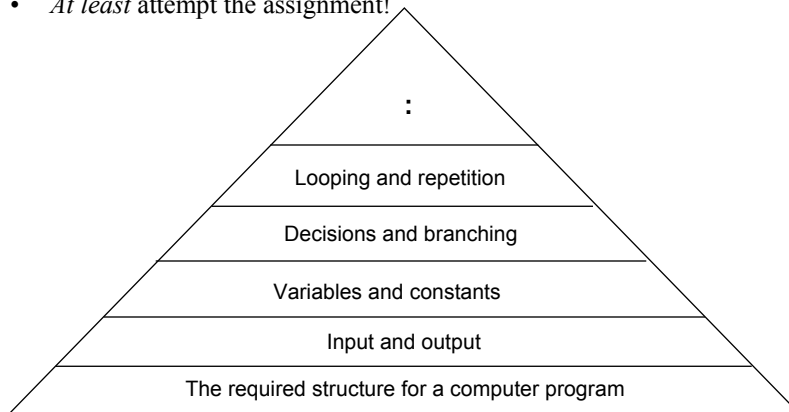
- Write lots programs
 - At the *very least* attempt the assignment.
 - Try to do some additional practice work (some examples will be given in class, some practice assignments will be available on the course web page).
 - Write lots of little ‘test’ programs to help you understand and apply the concepts being taught.
- Trace lots of code
 - Reading through programs that other people have written and understanding how and why it works the way that it does.

James Tam

How To Improve Your Program Writing Skills (2)

Make sure that you keep up with the material

- Many of the concepts taught later depend upon your knowledge of earlier concepts.
- Don't let yourself fall behind!
- *At least* attempt the assignment!



James Tam

How To Succeed In This Course (3)

Look at the material before coming to lecture so you have a rough idea of what I will be talking about that day:

- a) Read the slides
- b) Download and try running the programs
- c) (Advanced) try tracing the programs or figuring out how they work ahead of time.

James Tam

How To Succeed In This Course (4)

Start working on things as early as possible:

- Don't cram the material just before the exam, instead you should be studying the concepts as you learn them throughout the term.
- Don't start assignments the night (or day!) that they are due, they may take more time than you might first think so start as soon as possible.

James Tam

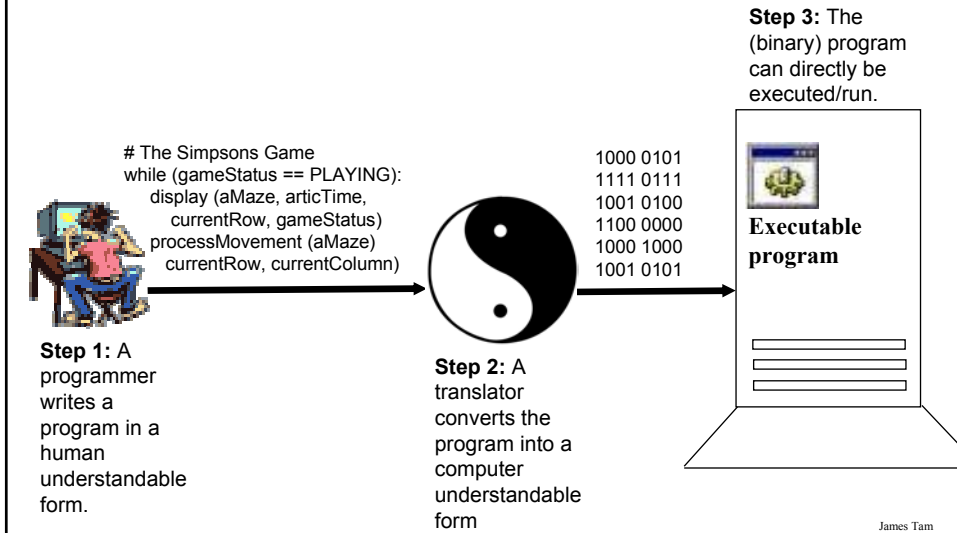
How To Succeed In This Course: A Summary

1. Practice things yourself
2. Make sure that you keep up with the material
3. Look at the material before coming to lecture
4. Start working on things early

James Tam

Translators

The (greatly simplified) process of writing a computer program.



What You Will Be Writing/Translating Your Programs With In This Class

Writing programs this semester:

- Documentation for the programming language: Python (actually it's a modified version of Python called Jython)
(JES: Basic)
 - Quick introduction: <http://www.cs.ucr.edu/~titus/JesIntro.pdf>
- **(Python: Advanced details about the language which JES is based on)**
 - My old CPSC 217 notes (Python):
<http://pages.cpsc.ucalgary.ca/~tamj/2008/217W/index.html>
 - Starting tutorial: <http://docs.python.org/tutorial/>
 - Full online documentation: <http://www.python.org/doc/>
- How to get the software to write/translate your Python programs: JES
 - Free download (JES v3.1) <http://coweb.cc.gatech.edu/mediaComp-plan/94>

The “Why’s” For This Section

Some advantages of the Python language:

- Free
- Powerful
- Practical and widely used (Google, NASA, Yahoo, Activision, Electronic Arts etc.)
- (Pure trivia): it was named after a British comedy group:



Monty Python © BBC

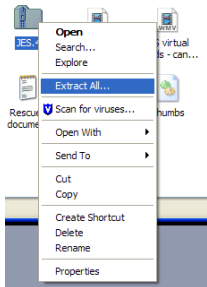
Advantage of the JES version of Python used in this class (Jython)

- The Jython language is based on the Python language
- Some graphical effects can be produced relatively easily

James Tam

Getting Started At Home¹

- Step 1: Download the version appropriate to your computer (Windows, Mac, Linux).
- Uncompress the compressed (zip format) file using the appropriate program (in Windows it’s built into the operating system). Right click on the downloaded file:

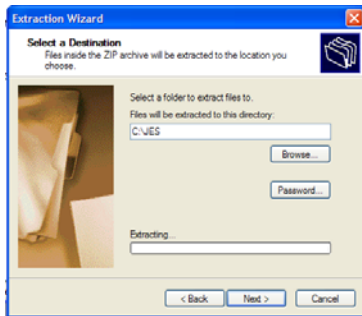


¹ Note: It is NOT required for this course that you install JES at home. No warranties or guarantees of service are provided for this program (i.e., we aren’t responsible if you inadvertently damage your computer during the installation process).

James Tam

Getting Started At Home (2)

Pick a name and location (that you will remember) to extract the files:

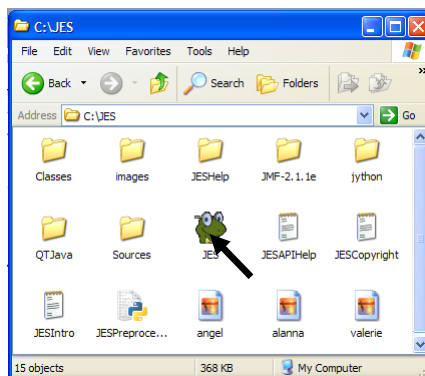


Note: to keep it simple, any data files (e.g., images) that you need for your programs should be stored in this folder/directory.

James Tam

Getting Started At Home (3)

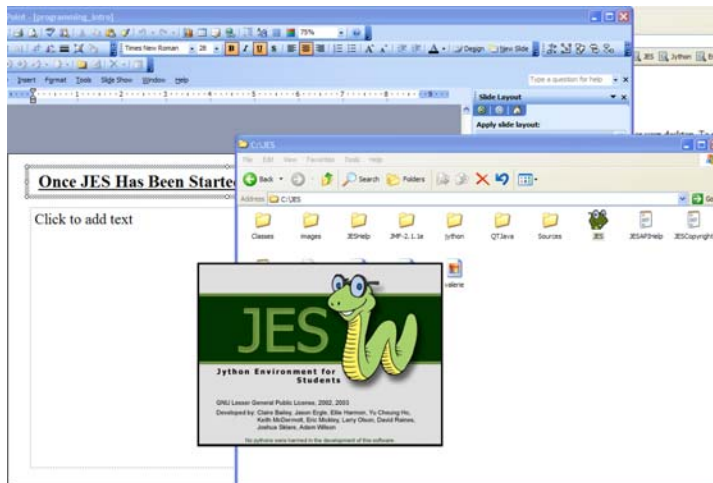
To start the program click on the 'JES' icon:



James Tam

Once JES Has Been Started (Home Or In The Lab)

The splash screen will first load (it may stay on quite a few seconds even with a fast computer)



James Tam

JES

Built in help for using JES and for some programming concepts in JPython

EDIT AREA
Type in your programs here

COMMAND AREA
Type in your commands (e.g., run the program that you just entered) here

File Edit Watcher MediaTools JES Functions Window Layout Help

Load Program Watcher Stop

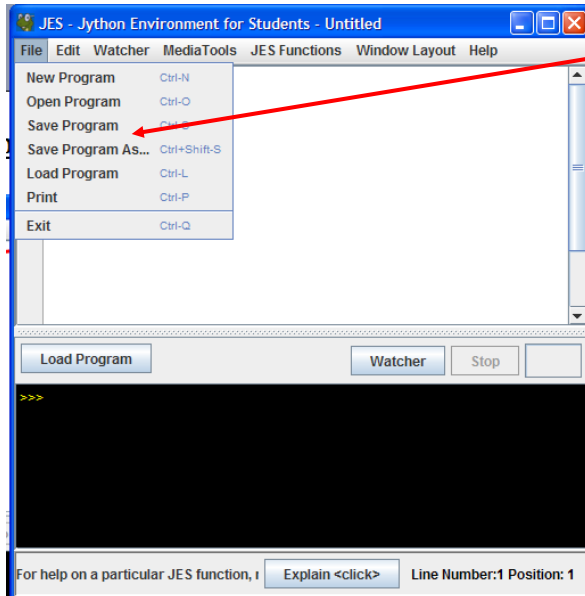
>>>

For help on a particular JES function, move the cursor over it Explain <click> Line Number:1 Position: 1

The image shows the JES IDE interface. The title bar reads "JES - Jython Environment for Students - Untitled". The menu bar includes "File", "Edit", "Watcher", "MediaTools", "JES Functions", "Window Layout", and "Help". The "Help" menu item is circled in red. The main area is a large text editor. Below the editor are buttons for "Load Program", "Watcher", and "Stop". At the bottom, there is a command prompt area with a prompt ">>>". A status bar at the very bottom says "For help on a particular JES function, move the cursor over it Explain <click> Line Number:1 Position: 1". Red arrows point from the annotations to the "Help" menu, the editor area, and the command prompt area.

James Tam

File Operations



Similar to a text editor (e.g., Notepad) and are used in conjunction with the creation of your programs (load, save, print etc.)

James Tam

The Basic Structure Of A Program In JES

Format:

```
def program name ():  
    Body of the program1
```

↑
Indent the body using three spaces (or at least make it consistent throughout).

Example: Available online and is called “start.py”:

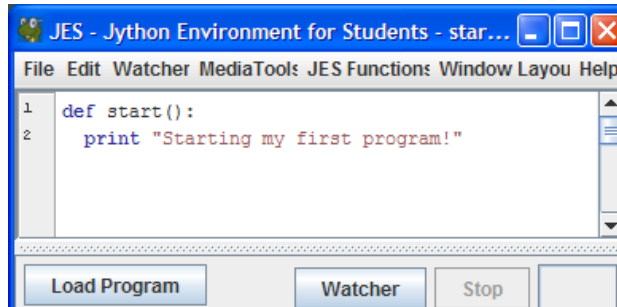
```
def start ():  
    print “Starting my first program!”
```

¹ This is the part that actually completes actions which are program instructions such as: displaying messages onscreen, loading a picture from file, performing a calculation etc.

James Tam

Creating And Running Your First Program

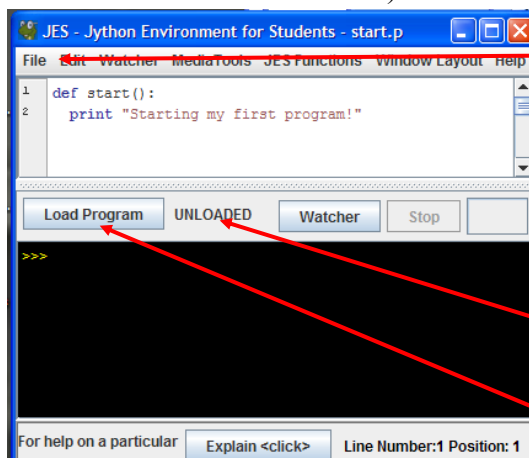
Step 1: Type your program in the editing area:



James Tam

Creating And Running Your First Program (2)

Step 2: Load your program so it can be translated into binary. (Currently your program has been loaded into the editor but not loaded into the JES translator).



The file menu affects the editor and not the translator (e.g., open doesn't load the program)

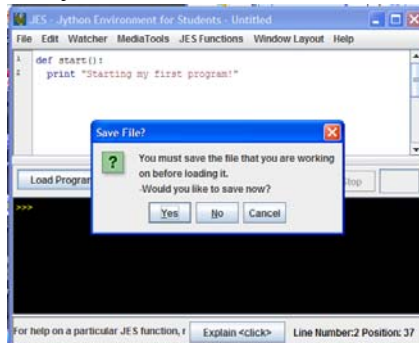
Program hasn't been 'loaded' yet

Load your program here

James Tam

Creating And Running Your First Program (2)

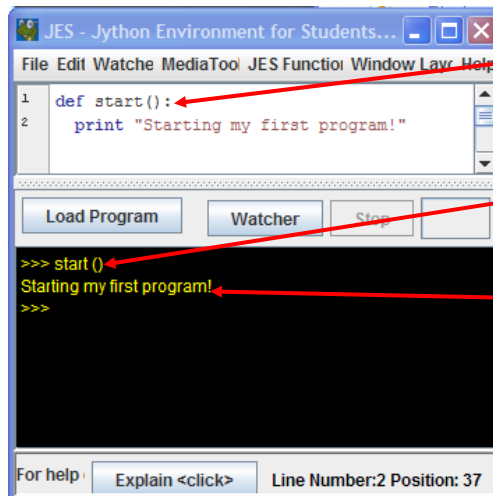
Step 3: When you run your program JES will ask if you want to save it. Save it with a name that corresponds to what your program is supposed to do (e.g., “grades” for a grade calculator) or at least matches the name that you gave it in the editor (e.g., “start” in the example). Make sure that you save it in a location that you will remember and be consistent in the location choice.



James Tam

Creating And Running Your First Program

Step 4: Run your program in the command area



IMPORTANT: the name that you type to run the program must match the program name specified here

Running program called 'start'

The effect of running your program is displayed immediately after you run it.

James Tam

Displaying Output

- A method of communicating information to the user of the program e.g., the result after performing a calculation, instructions for running the program.

Format:

print *"the message that you wish to appear"*

The message is referred to as a 'literal string' because what is between the quotes will 'literally' appear onscreen.

Example:

```
def aProgram ():  
    print "foo"  
    print "bar"
```

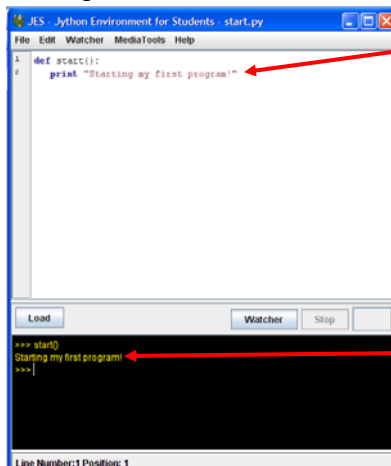
Be consistent in the use of quotes or you'll get an error!

James Tam

Displaying String Output

- Output: Display information or messages in the command area of JES.

Example:



Using Print:

- A command to display information

The effect of print:

- In this case, everything between the quotes will appear in the command area (string = series of characters).

James Tam

Storing Information



- When writing a program there will often be a need to store information e.g., to perform a calculation.
- Information is stored in a computer program by using a variable.
- A variable is a location in memory that will temporarily store information.

This location can store one 'piece' of information

- *At most* the information will be accessible as long as the program runs

Some of the types of information which can be stored in variables:

- Integer
num = 10
- Real numbers
num = 10.5
- Strings
message = "Not happy! >-<"

Picture from Computers in your future by Pfaffenberger B

James Tam

Variables: Arithmetic Operators

Operator	Description	Example
=	Assignment	num = 7
+	Addition	num = 2 + 2
-	Subtraction	num = 6 - 4
*	Multiplication	num = 5 * 4
/	Division	num = 25 / 5
**	Exponent	num = 9 ** 2

James Tam

Output: Variables

Displaying the current contents of a variable is not the same as displaying a string of characters (the latter is dynamic, the exact output can change as the program runs).

Format:

```
print <variable name>
```

Note: there are no quotation marks when displaying the contents of a variable.

Example: Available online and is called “output1.py”

```
def output1 ():  
    num = 10  
    print num
```

James Tam

Variables Must Created Before They Can Be Used!

Assigning a value to a variable reserves a location in memory with this name and puts the specified value at that location.

- num = 123

The contents of a variable cannot be accessed before the variable has been created and a value assigned to it!

```
def anError ():  
    print num  
    num = 123
```

•At this point the variable called 'num' does not yet exist.

•The program cannot recover from this error and will end.

James Tam

Displaying Mixed Output

Strings and the contents of variables can be intermixed with a single print statement.

Format:

```
print "<string>", <variable name>...
```

Example: Available online and is called "profit.py":

```
def profit ():  
    income = 2000  
    expenses = 1500  
    profit = income - expenses  
    print "Income: ", income, " Expenses: ", expenses, " Profit: ", profit
```

James Tam

Variable Naming Conventions

- The name should be meaningful.
- Can't be a word with a predefined meaning in Python e.g., "print".
- Names are case sensitive but avoid distinguishing variable names only by case (bad programming style).
- Variable names should generally be all lower case.
- For variable names composed of multiple words, separate each word by capitalizing the first letter of each word (save for the first word) or by using an underscore. (Be consistent!)

James Tam

Constants

- Memory locations that *shouldn't* change.
- Used to make the program easier to read and understand:
PI = 3.14
- Differentiated from variables by capitalization:
 - Multi-word constants can use the underscore to separate words e.g.,
MAX_SIZE = 10

James Tam

Getting User Input

In Python user input can take two forms:

- Numerical (can be used in calculations):
 - 888, 12.34
- Text (alphabetic, numeric, other keyboard characters):
 - Asdfjas123*6! >-<



Person to a computer



James Tam

Getting Numerical Input

Use the 'input' function

Format:

```
<variable name> = input("<Prompting message>")
```

Example: Available online and is called "input1.py":

```
def input1 ():  
    num = input ("Type in a number: ")  
    print num
```

James Tam

Getting Text Input

Use the 'raw_input' function

Format:

```
<variable name> = raw_input("<Prompting message>")
```

Example: Available online and is called "input2.py":

```
def input2 ():  
    userName = raw_input ("Tell me your name: ")  
    print "hello ", userName
```

James Tam

Working With Picture Variables

- One of the strengths of JES is the ease at which multimedia files (such as images) can be incorporated in a computer program.
- Example: Available online and is called “picture1.py”, requires that you also download and save the image called “lion.jpg” to the folder where you put JES).

```
def picture1():  
    picture = makePicture ("lion.jpg")  
    show (picture)
```

James Tam

Alternative For Getting Input

- In JES it can be done as the program runs.
- Example: How to specify the name of the images as the program runs. (Available online and is called “picture2.py”):

```
def picture2 (file1,file2):  
    picture1 = makePicture(file1)  
    show(picture1)  
  
    picture2 = makePicture(file2)  
    show(picture2)
```

- To run this program you must enter the name of two images as you run the program in the command area
 - E.g., (if you typed the following as you run the program in JES)
 - picture2("angel.jpg", "valerie.jpg")

James Tam

Getting A File Dialog Box

- A file dialog box will allow you to graphically select the image to load into your program.
- Example: Available online and is called “picture3.py”

```
def picture3():  
    filename = pickAFile()  
    myPicture = makePicture (filename)  
    show (myPicture)
```

James Tam

Program Documentation

Used to provide information about a computer program to another *programmer*:

- Often written inside the same file as the computer program (when you see the computer program then you can see the documentation).
- The purpose is to help other programmers understand how the program code was written: how it works, what are some of it's limitations etc.
- (It's purpose is similar to the 'Description' field in the Design view of MS-Access.

James Tam

Program Documentation (2)


- It doesn't get translated into binary.
- It doesn't contain instructions for the computer to execute.
- It's for the reader of the program:
 - What does the program do e.g., tax program.
 - What are its capabilities e.g., it calculates personal or small business tax.
 - What are its limitations e.g., it only follows Canadian tax laws and cannot be used in the US.
 - What is the version of the program
 - If you don't use numbers for the different versions of your program then consider using dates.
 - How does the program work.
 - This is often a description in English (or another human language) that describes the way in which the program operates.
 - The purpose of this description is to help the reader quickly understand how the program works.

James Tam

Program Documentation (3)

Format:

<Documentation>


The number sign '#'
flags the translator that
what's on this line is
documentation.

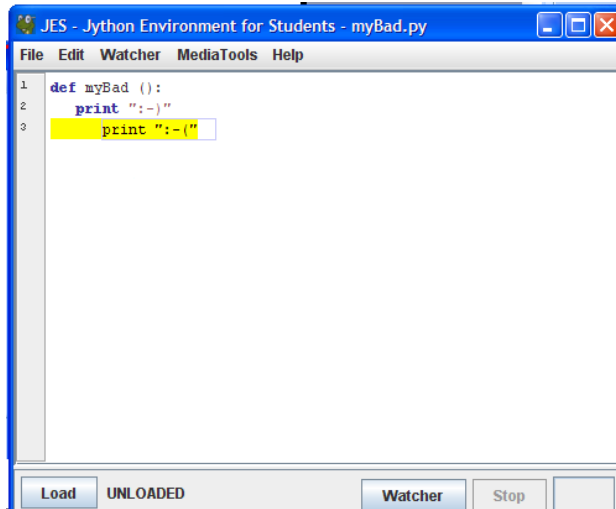
Examples:

- # Tax-It v1.0: This program will electronically calculate your tax return.
- # This program will only allow you to complete a Canadian tax return.

James Tam

Common Errors

1. Inconsistent indenting (remember that you should only indent the 'body' of something).



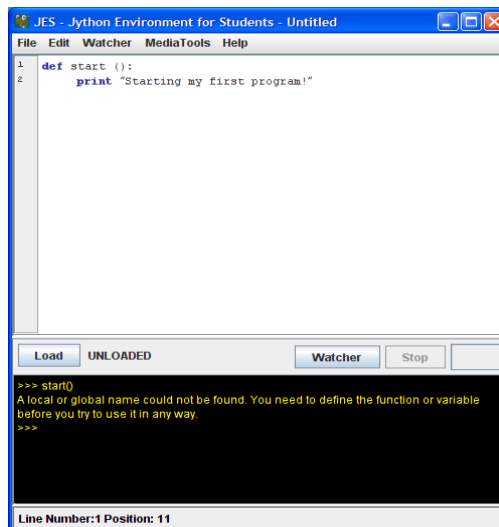
The screenshot shows a window titled "JES - Jython Environment for Students - myBad.py". The code editor contains the following Python code:

```
1 def myBad ():  
2     print ":-)"  
3     print ":-("
```

The third line is highlighted in yellow. The status bar at the bottom shows "Load UNLOADED", "Watcher", and "Stop" buttons. The name "James Tam" is visible in the bottom right corner.

Common Errors (2)

2. Forgetting to load your program before trying to run it.



The screenshot shows a window titled "JES - Jython Environment for Students - Untitled". The code editor contains the following Python code:

```
1 def start ():  
2     print "Starting my first program!"
```

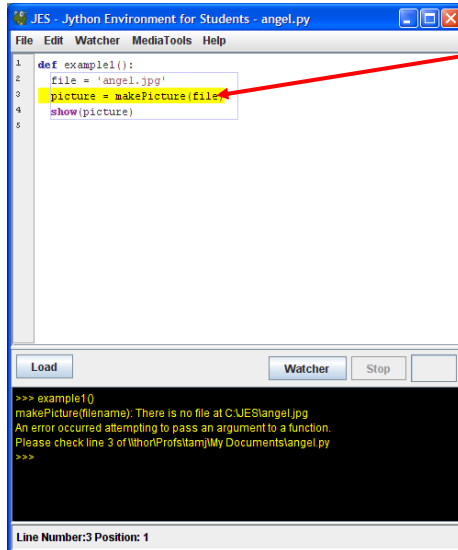
The status bar at the bottom shows "Load UNLOADED", "Watcher", and "Stop" buttons. Below the code editor, a black console window displays the following error message:

```
>>> start()  
A local or global name could not be found. You need to define the function or variable  
before you try to use it in any way.  
>>>
```

The status bar at the bottom of the console window shows "Line Number:1 Position: 11". The name "James Tam" is visible in the bottom right corner.

Common Errors (3)

3. Your program cannot find a file that it needs.



The screenshot shows the JES Python environment with a script named 'angel.py'. The script contains the following code:

```
1 def example1():  
2     file = 'angel.jpg'  
3     picture = makePicture(file)  
4     show(picture)  
5
```

The error message in the console reads: "makePicture(filename). There is no file at C:\JES\angel.jpg. An error occurred attempting to pass an argument to a function. Please check line 3 of \\horProf\slamj\My Documents\angel.py". A red arrow points from the error message to the line of code that defines the file path.

It can't find the file 'angel.jpg' to make the picture

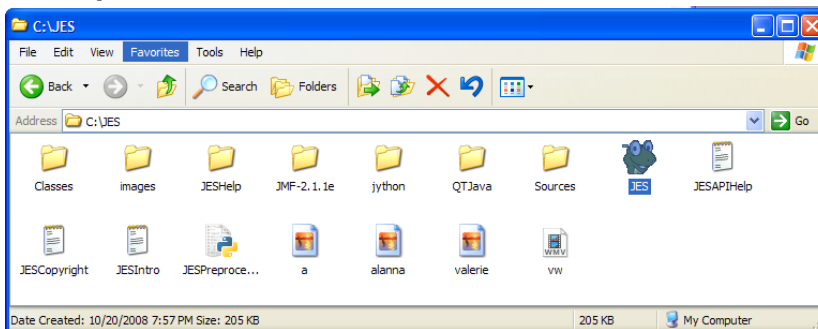
James Tam

Common Errors (3)

3. Problem: Your program cannot find a file that it needs.

Solution: Put all the files (pictures, sounds, videos) in the same folder where you put JES.

Example:



James Tam

Common Errors (4)

4. Forgetting required parts of the program

- Forgetting the word “def” and/or the name of the program

(Erroneous version)

```
print “hello”
```

(Fixed version)

```
def programName ():  
    print “hello”
```

- Forgetting the brackets and/or the colon:

```
def programName  
    print “hola”
```

James Tam

Common Errors (5)

4. Forgetting required parts of the program

- Forgetting the quotes when displaying a string of characters:

```
def programName ():  
    print wei???!!
```

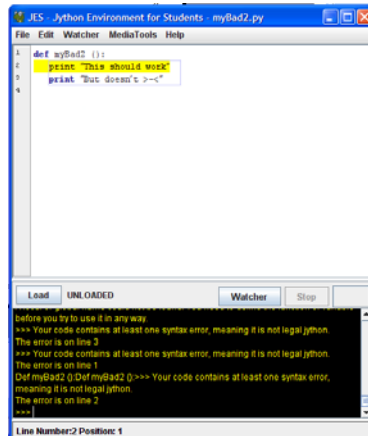
- Mismatching quotes:

```
def programName  
    print “konichiwa!
```

James Tam

Common Errors (6)

5. Typing your JES program using a word processor or other program that allows for powerful text formatting. These errors can be very tough to find and fix because there is no visible error in your program.



The screenshot shows the JES Python Environment for Students interface. The code editor displays the following Python code:

```
def myBad2 ():  
1 print "This should work"  
2 print "But doesn't >-c"  
3  
4
```

The error message in the console reads:

```
before you try to use it in any way.  
*** Your code contains at least one syntax error, meaning it is not legal jython.  
The error is on line 3  
*** Your code contains at least one syntax error, meaning it is not legal jython.  
The error is on line 1  
Def myBad2 ():  
1 print "This should work"  
2 print "But doesn't >-c"  
3  
4  
*** Your code contains at least one syntax error, meaning it is not legal jython.  
The error is on line 2  
***
```

At the bottom of the console, it says "Line Number: 2 Position: 1".

James Tam

You Should Now Know

- How to create and run a program using JES
- How the print statement displays information
- What are variables and how to use them in a program
- What is the purpose of constants and how they differ from variables
- Common arithmetic operators in Jython
- How to load and display images in a program
- How to get input as a program runs
- The purpose of program documentation and how to document a program
- Common programming errors

James Tam