ASTR 2600
Introduction to Scientific Coding

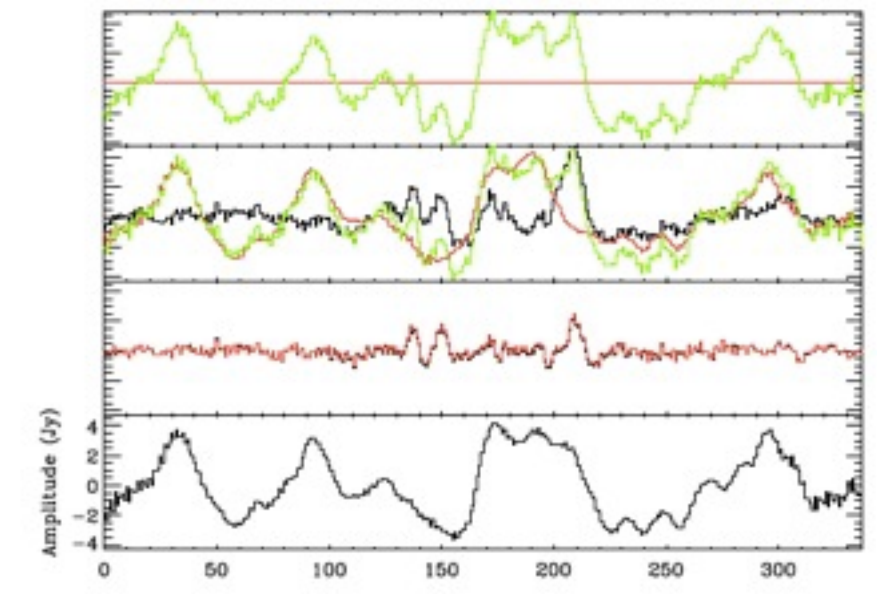
Monday, January 14th, 2013

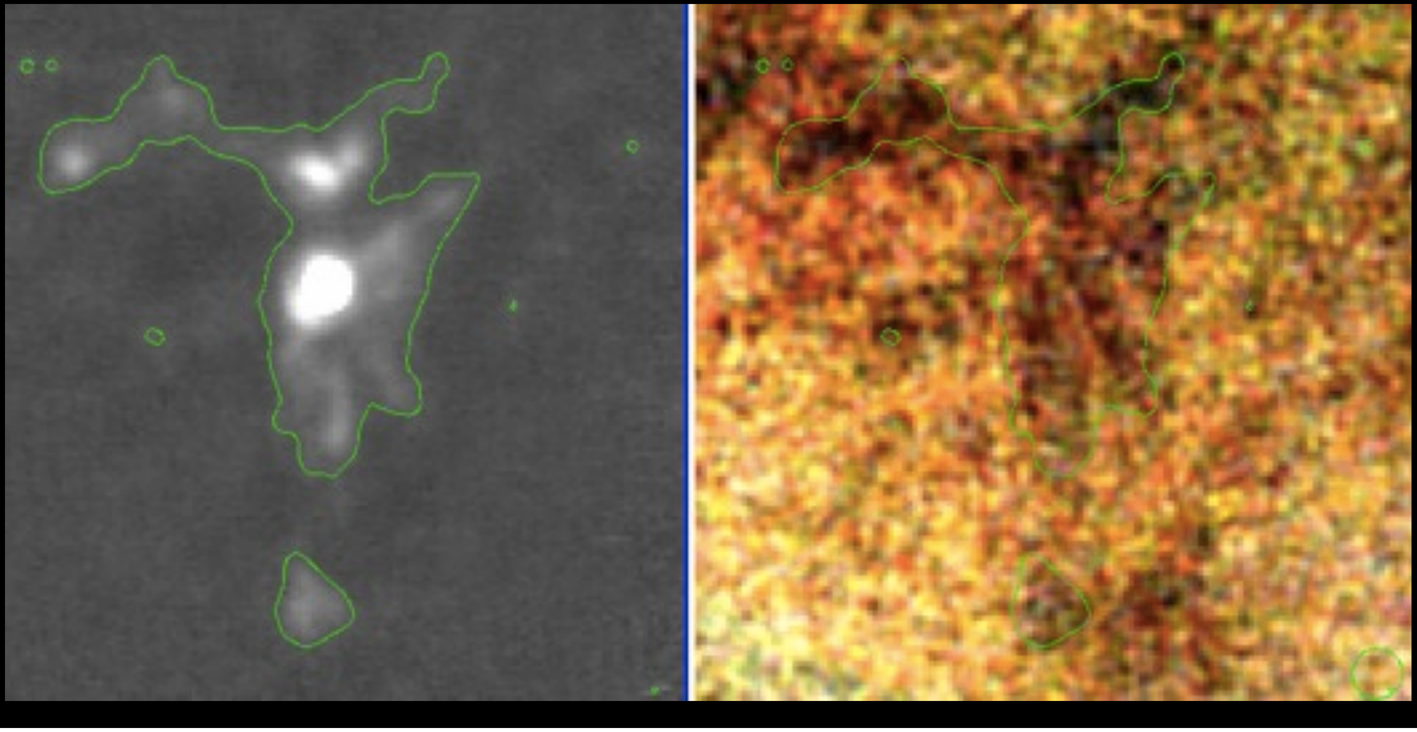
Programming and Astronomy

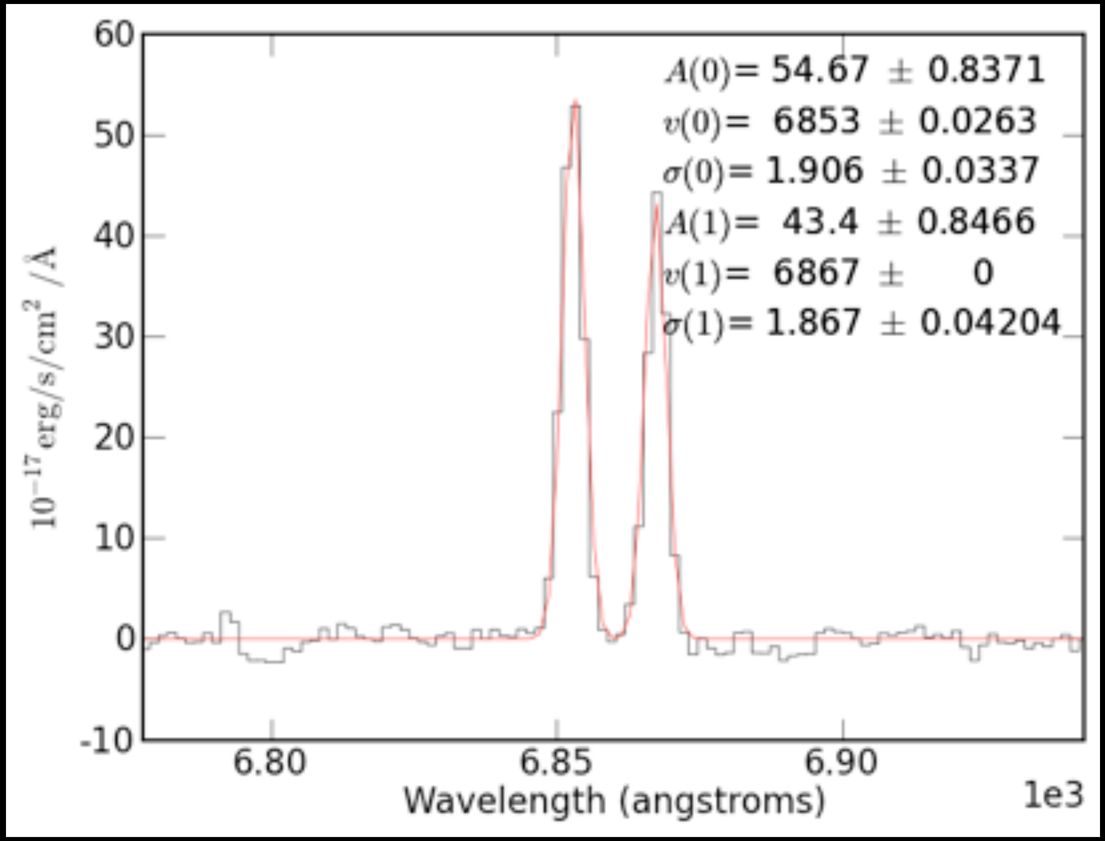
- To do astronomy, or nearly any physics, you need to know how to use computers
- But most astronomers get no formal training in programming!
- This course is designed to fill that gap

Programming and Astronomy

Data Processing

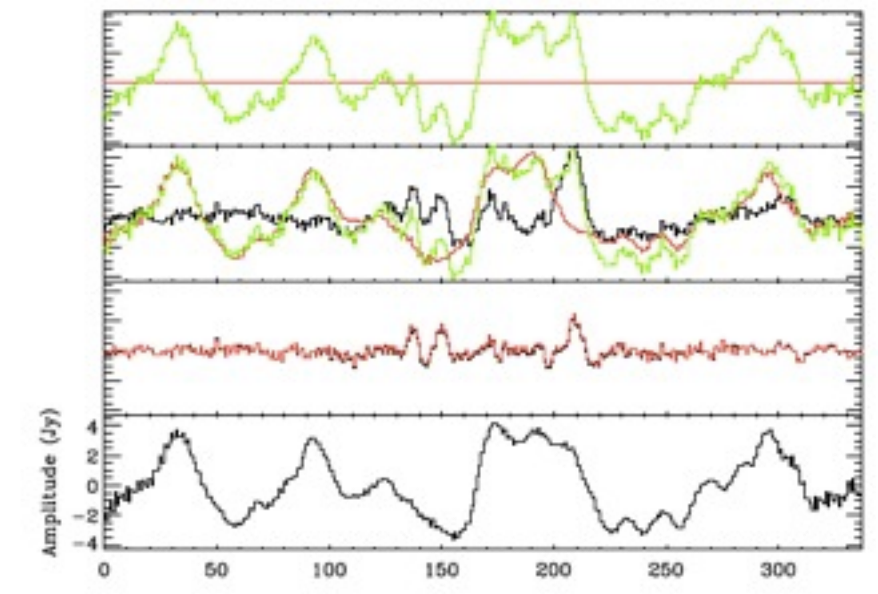


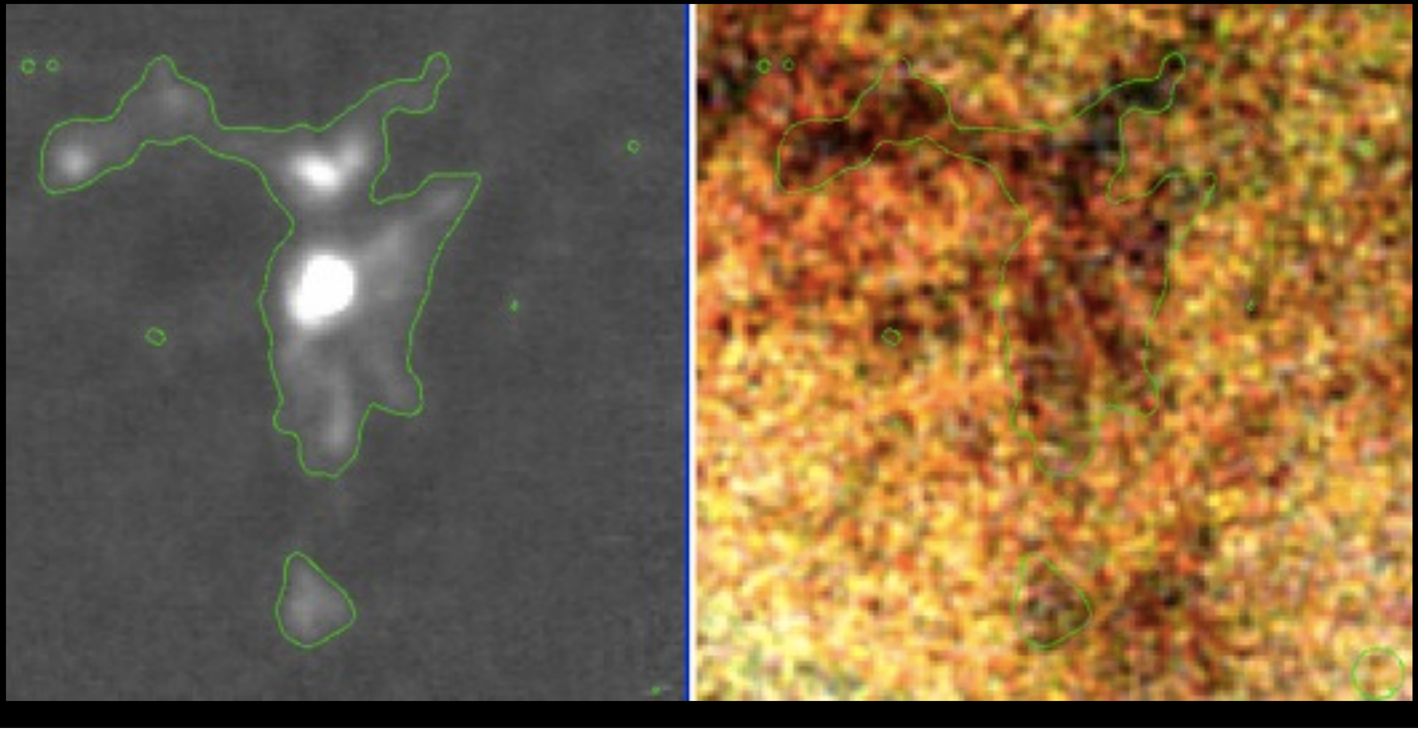


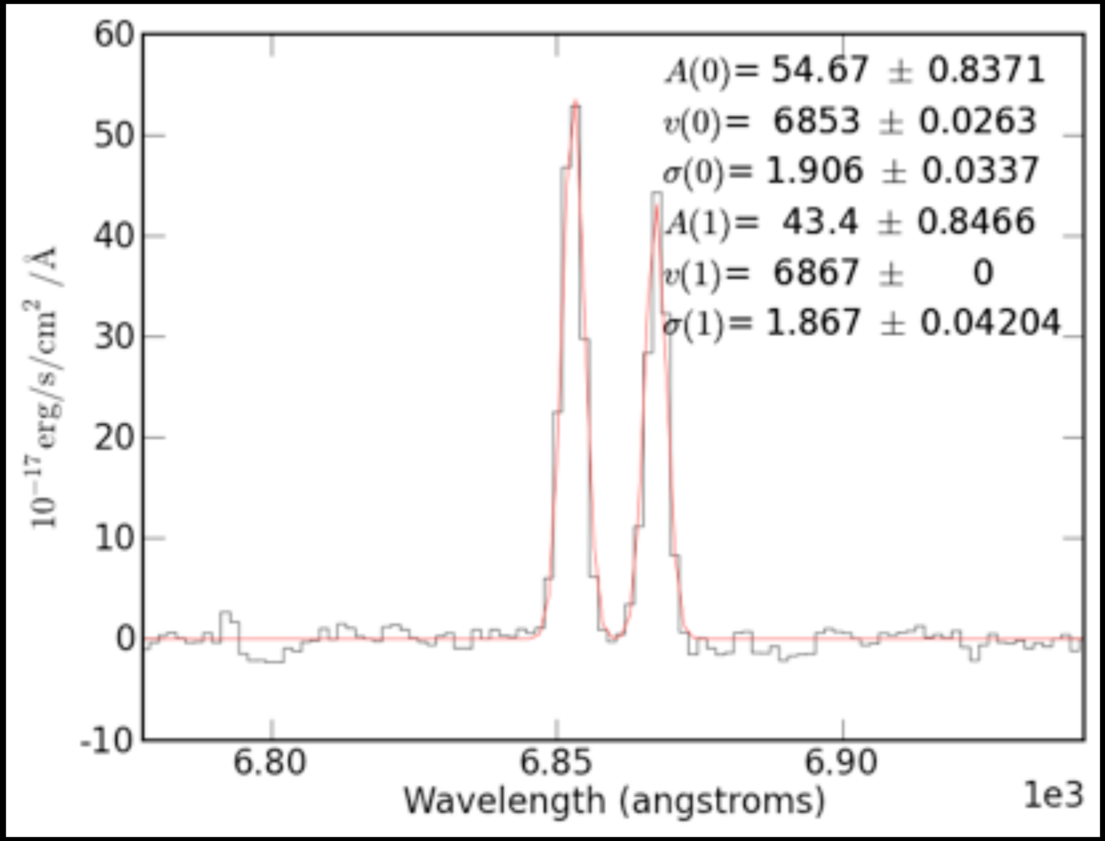


Programming and Astronomy

Plotting!

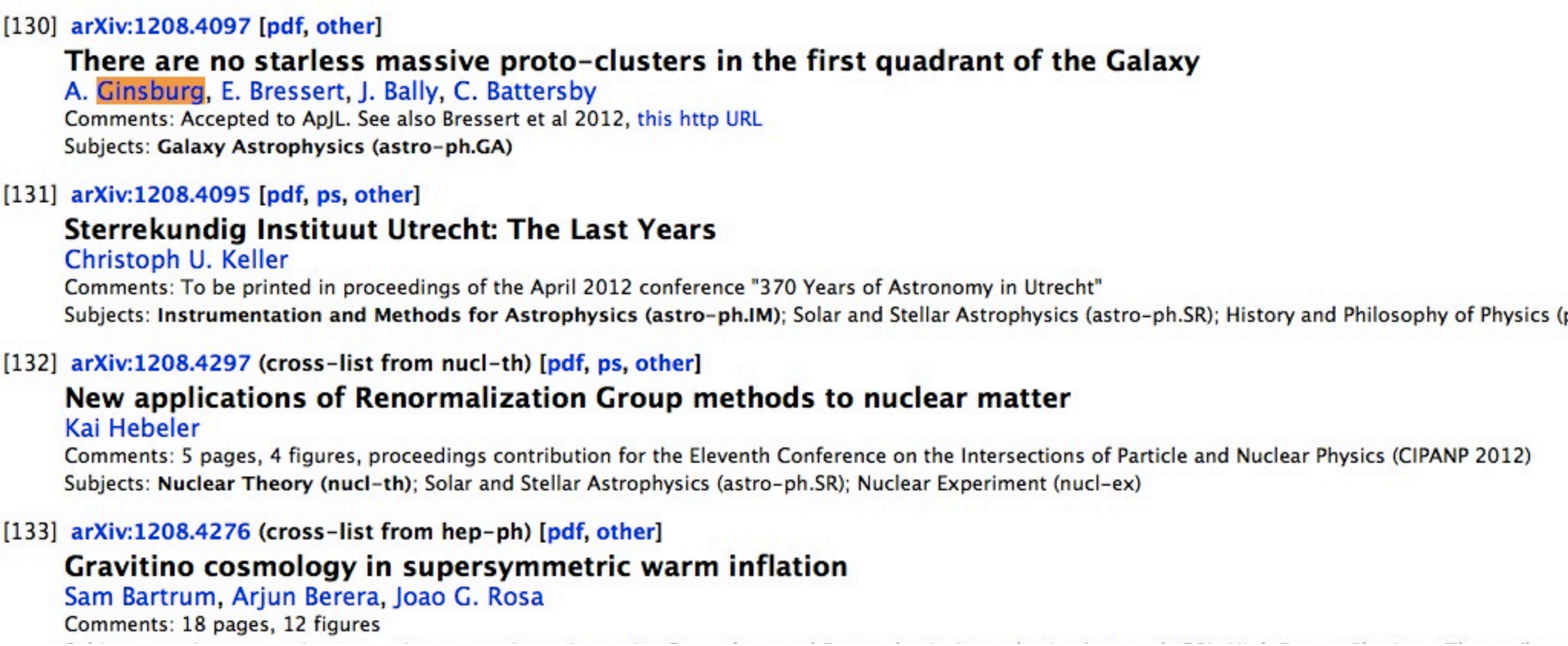






Programming and Astronomy

Sharing results



[130] arXiv:1208.4097 [pdf, other]

There are no starless massive proto-clusters in the first quadrant of the Galaxy

A. Ginsburg, E. Bressert, J. Bally, C. Battersby

Comments: Accepted to ApJL. See also Bressert et al 2012, this http URL

Subjects: **Galaxy Astrophysics (astro-ph.GA)**

[131] arXiv:1208.4095 [pdf, ps, other]

Sterrekundig Instituut Utrecht: The Last Years

Christoph U. Keller

Comments: To be printed in proceedings of the April 2012 conference "370 Years of Astronomy in Utrecht"

Subjects: **Instrumentation and Methods for Astrophysics (astro-ph.IM)**; Solar and Stellar Astrophysics (astro-ph.SR); History and Philosophy of Physics (p

[132] arXiv:1208.4297 (cross-list from nucl-th) [pdf, ps, other]

New applications of Renormalization Group methods to nuclear matter

Kai Hebeler

Comments: 5 pages, 4 figures, proceedings contribution for the Eleventh Conference on the Intersections of Particle and Nuclear Physics (CIPANP 2012)

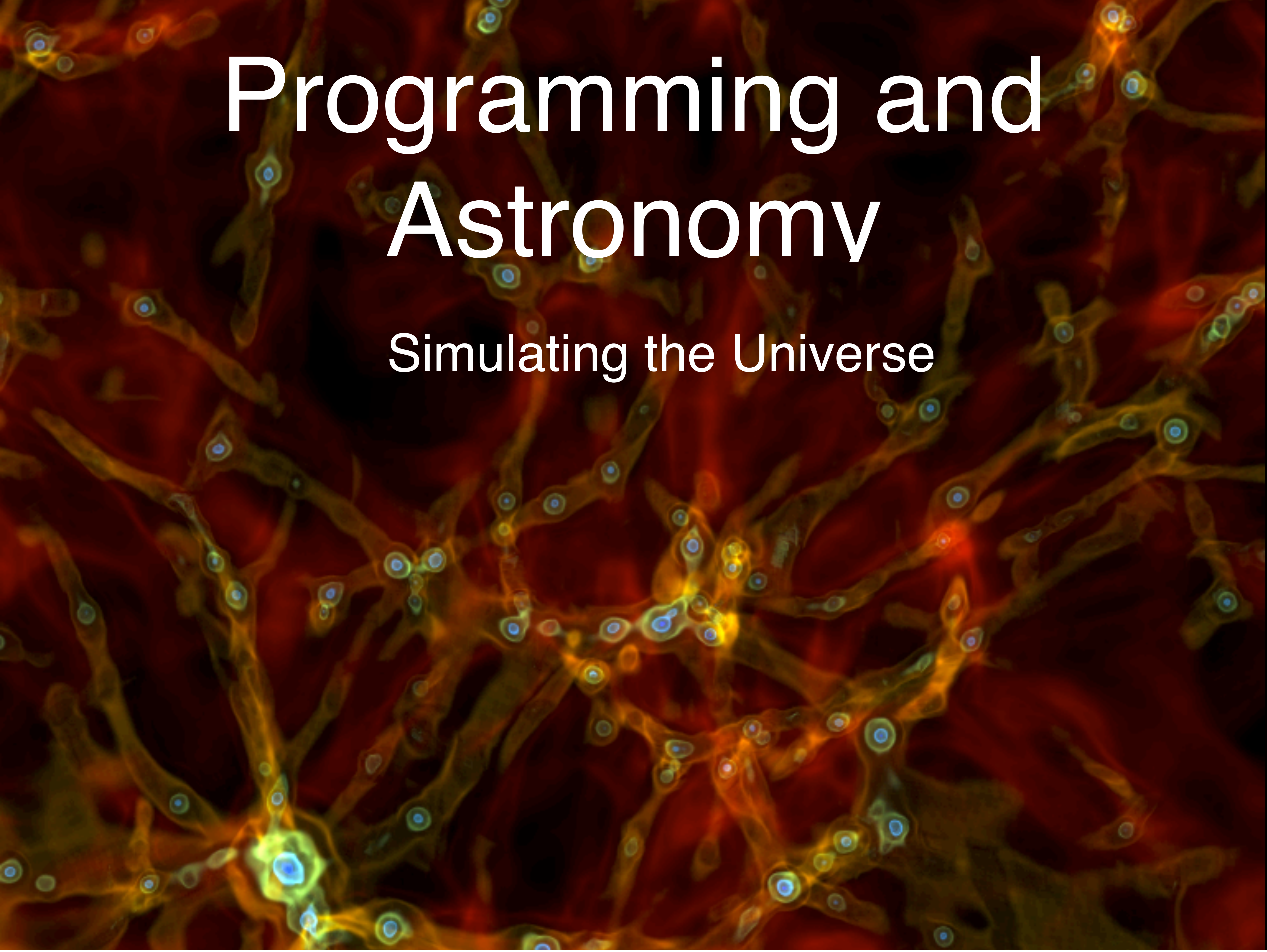
Subjects: **Nuclear Theory (nucl-th)**; Solar and Stellar Astrophysics (astro-ph.SR); Nuclear Experiment (nucl-ex)

[133] arXiv:1208.4276 (cross-list from hep-ph) [pdf, other]

Gravitino cosmology in supersymmetric warm inflation

Sam Bartrum, Arjun Berera, Joao G. Rosa

Comments: 18 pages, 12 figures



Programming and Astronomy

Simulating the Universe

Programming and Astronomy

- What languages do astronomers use?
  - Fortran
  - C++
  - IDL
  - Python*
  - IRAF
  - Matlab
  - Javascript
  - ...

*this is what I use, and you can attempt any assignment in this course in python if you are motivated

This course

- Fundamentals of programming
- IDL: the Interactive Data Language
- Python
- UNIX and the command line
- Version Control

Course Outline

- Unix and IDL command-line interaction
- Data types, equations, built-in IDL features
- Graphics

Course Outline

- Arrays
- Numerical derivatives and integration
- Procedures and Functions
- File input and output

Course Outline

- Writing programs
- Flow control
- Software design
- Animation
- Object-Oriented programming

Course Outline

- Curve fitting
- Interpolation
- Regression
- Recursion
- N-body simulation

Syllabus

- Exercises:
  - Tasks to do in IDL
  - Completion credit
    - 0,1,2 grade scale
  - 1-week deadline

Syllabus

- Whuduzitdo? = What does it do
  - Tasks to do *by hand* (no computer!)
  - Completion credit
    - 0,1,2 grade scale
  - Next class time deadline

Syllabus

- Tutorials
    - Tasks to do in class (and finish later?)
    - Completion credit
        - 0,1,2 grade scale
    - Often, you can’t do the homework without finishing these first

Syllabus

- Homework
  - Tasks to do on the computer
  - Correctness credit (0-100 grade scale)
    - Code must run! Code that crashes will result in ZERO credit!
  - 1-week deadline
    - late work accepted up to 2 weeks after deadline at 3 point per business day markdown

Syllabus

- Overall grade is 65% homework
- There may be a test at the end of the semester that resembles “What does it do” assignments
  - or we’ll do a final project
- Clicker questions worth 10% (mostly participation)

Syllabus

- Office hours - TBD
- Will also have “remote office hours”
    - on github, you can ask questions
    - e-mail is the best way to contact Adam

Pause for questions

This slide intentionally left blank

(even though it’s not really)

First assignment

- Assignment 0
  - includes exercise, whuduzitdo
- Reading: Chapter 1, pages 1-28
  - This is a lot of reading; if you can do the exercises after today's lecture, just skim so you know how to use the text as a reference

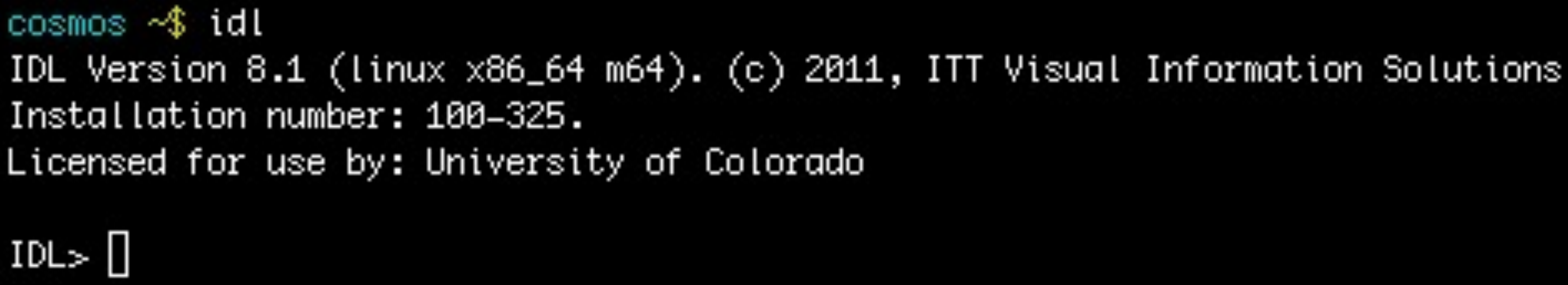
Logging in to cosmos

- We will do our work on “workstations” connection to a “server” called cosmos.colorado.edu
- Log in with your CU identikey username and password

Open a Terminal

- If you don’t have a terminal open already, go to the “start menu” and under “system tools” click Terminal
  - it may also be called Konsole

Intro to IDL: The Command Line

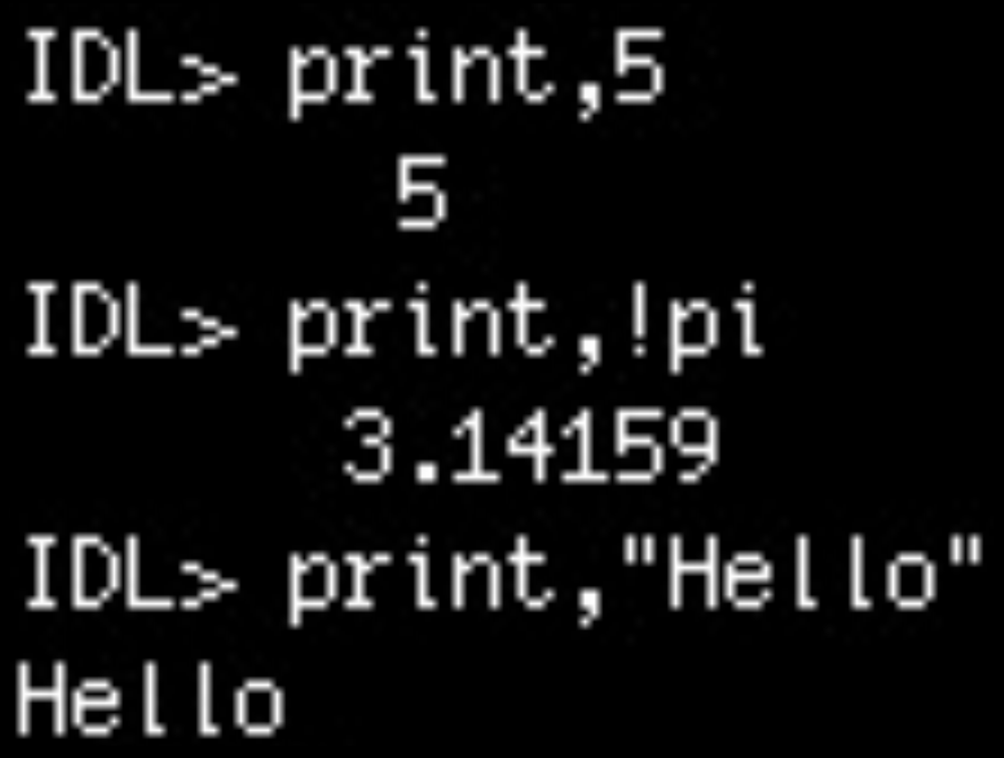


```
cosmos ~$ idl
IDL Version 8.1 (linux x86_64 m64). (c) 2011, ITT Visual Information Solutions
Installation number: 100-325.
Licensed for use by: University of Colorado

IDL> []
```

Your terminal window will look something like this

Commands: `print`

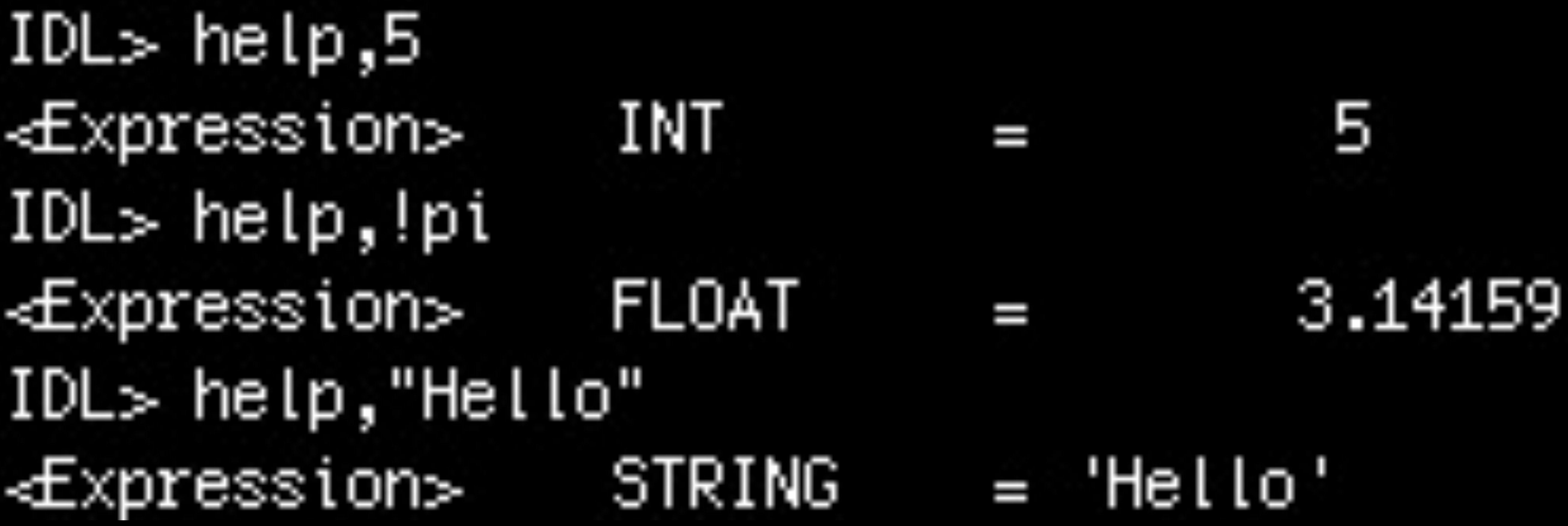


```
IDL> print,5
       5
IDL> print,!pi
      3.14159
IDL> print,"Hello"
Hello
```

- The `print` command: any time you want to see something “printed” on the next line (not to the printer)

Commands: `help`

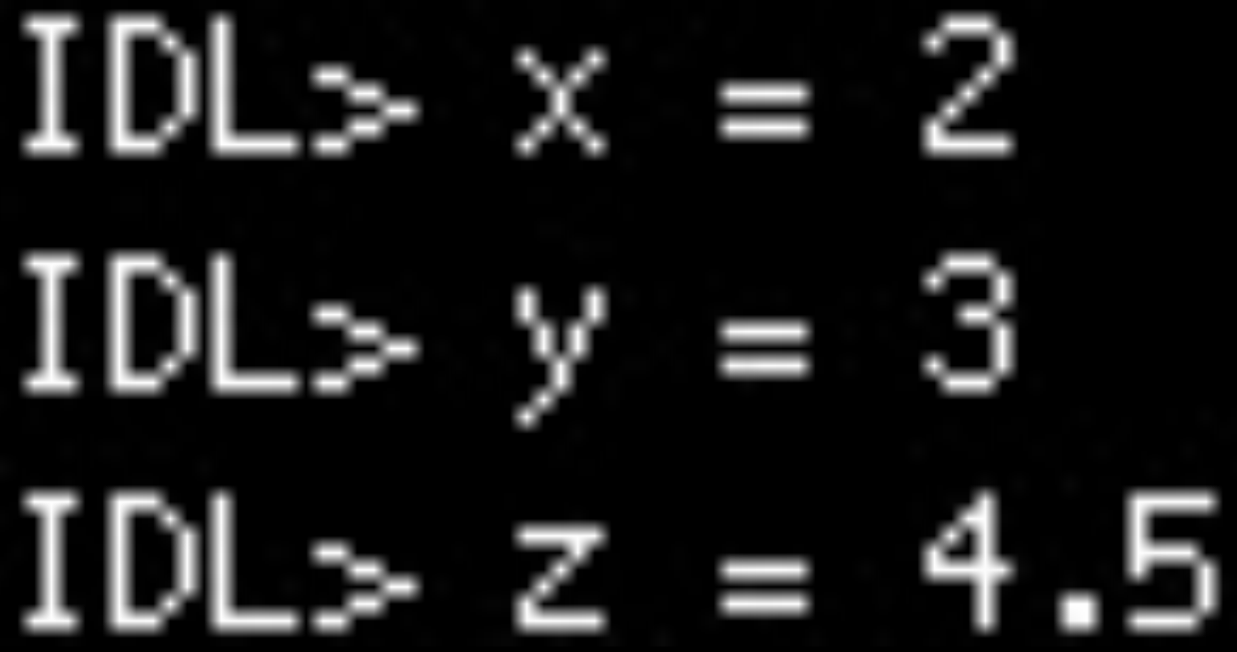
- Shows properties of its arguments



```
IDL> help,5
<Expression>    INT         =         5
IDL> help,!pi
<Expression>    FLOAT       =       3.14159
IDL> help,"Hello"
<Expression>    STRING      = 'Hello'
```

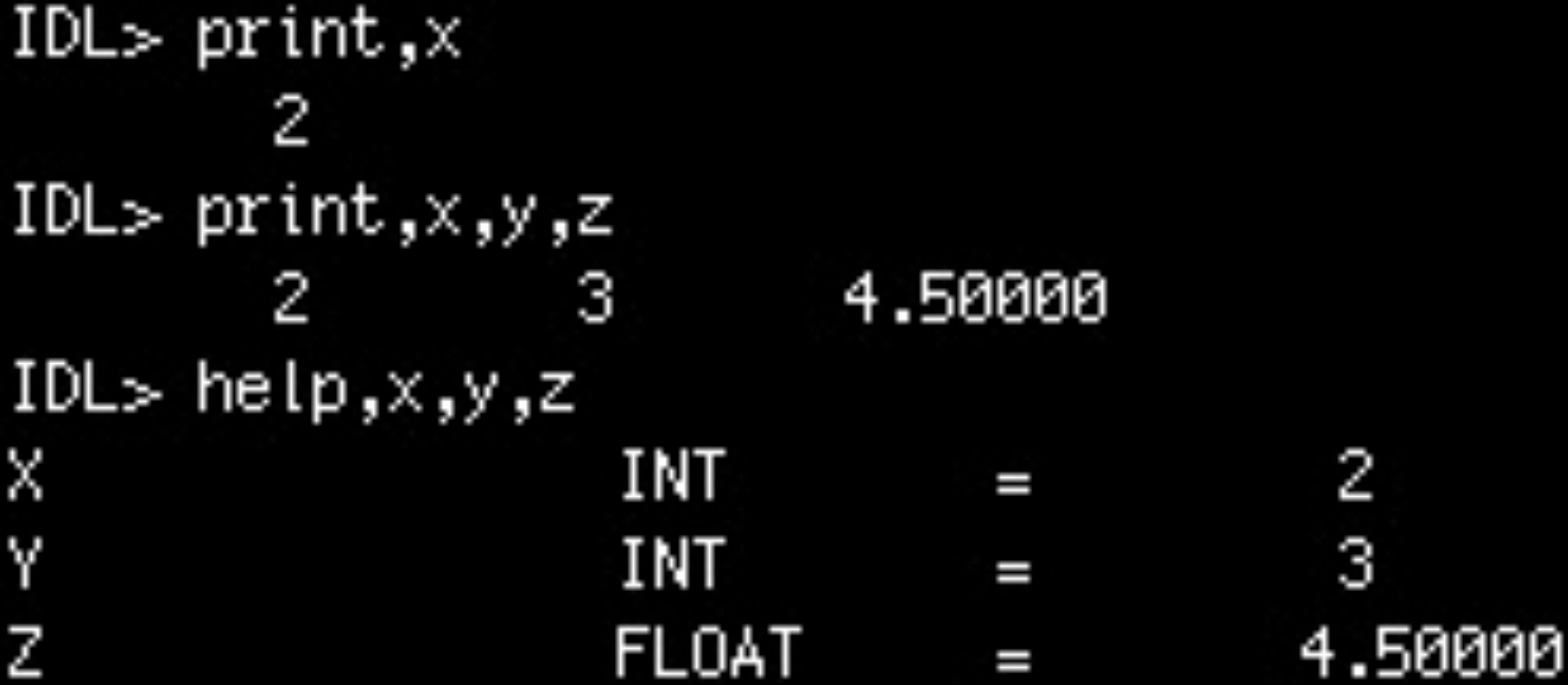
Variables

- “Assign” values to variables



```
IDL> x = 2
IDL> y = 3
IDL> z = 4.5
```

Use help, print



```
IDL> print,x
       2
IDL> print,x,y,z
       2       3      4.50000
IDL> help,x,y,z
X               INT       =        2
Y               INT       =        3
Z               FLOAT     =      4.50000
```

Operator Precedence

Math question! Evaluate 6/2+3*4-5^2

(this is what “Whuduzitdo?” questions are like)

A) -1

B) -10

C) -6/61

D) 6/5

E) I don’t know

Operator Precedence

- IDL behaves as expected:
- (), ^, */, +-

Operator Precedence

Math question! Evaluate 6/2+3*4-5^2

With no added parentheses, B was correct

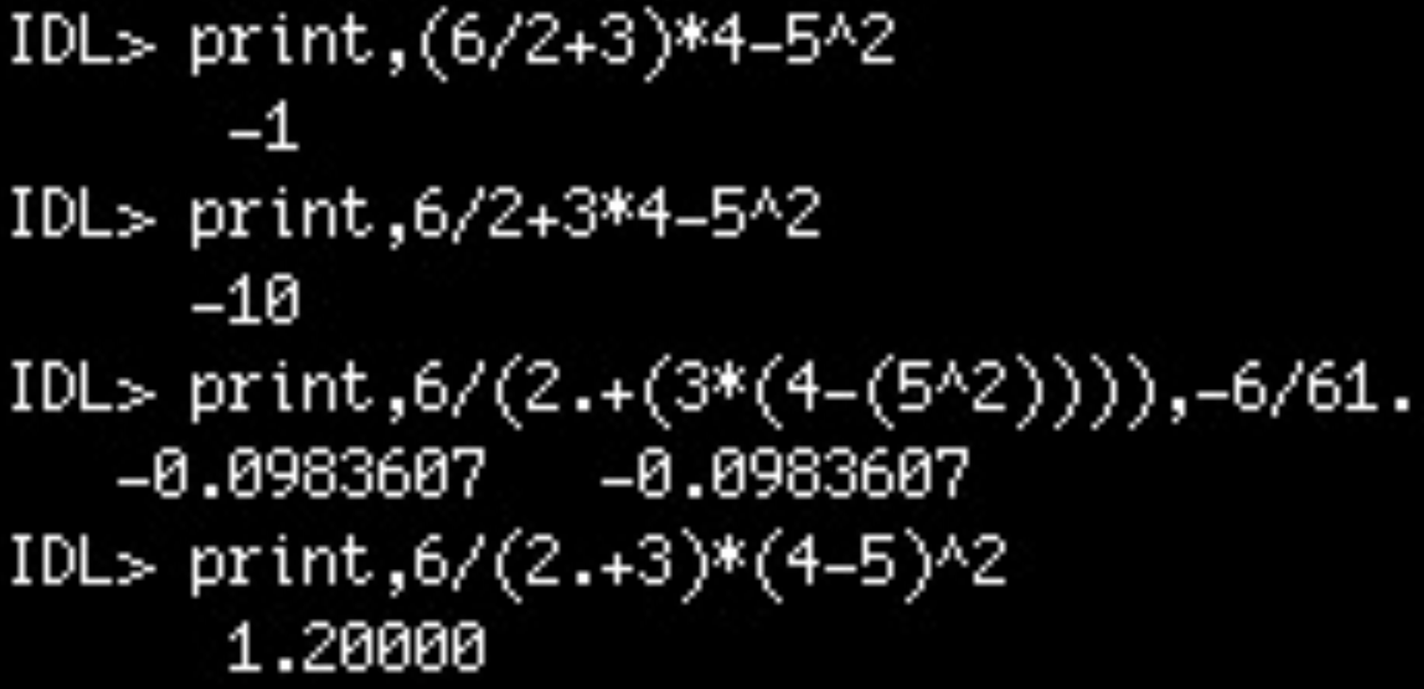
A) -1

B) -10

C) -6/61

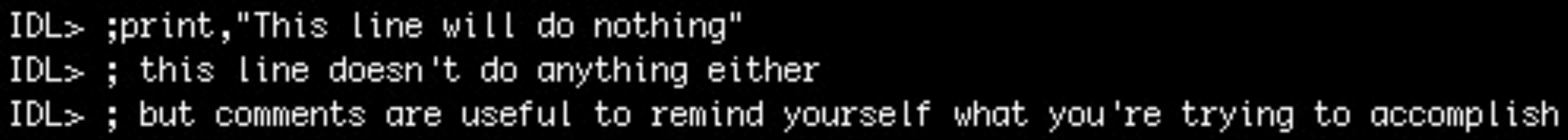
D) 6/5

E) I don’t know



```
IDL> print,(6/2+3)*4-5^2
      -1
IDL> print,6/2+3*4-5^2
     -10
IDL> print,6/(2.+(3*(4-(5^2)))),-6/61.
  -0.0983607  -0.0983607
IDL> print,6/(2.+3)*(4-5)^2
      1.20000
```

Comments

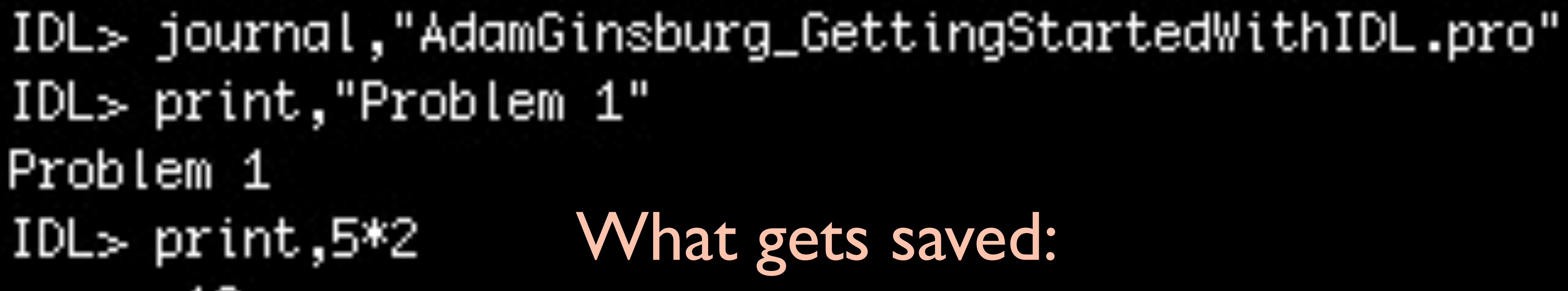


```
IDL> ;print,"This line will do nothing"
IDL> ; this line doesn't do anything either
IDL> ; but comments are useful to remind yourself what you're trying to accomplish
```

- Semicolon ; preceding a line tells IDL to ignore the line

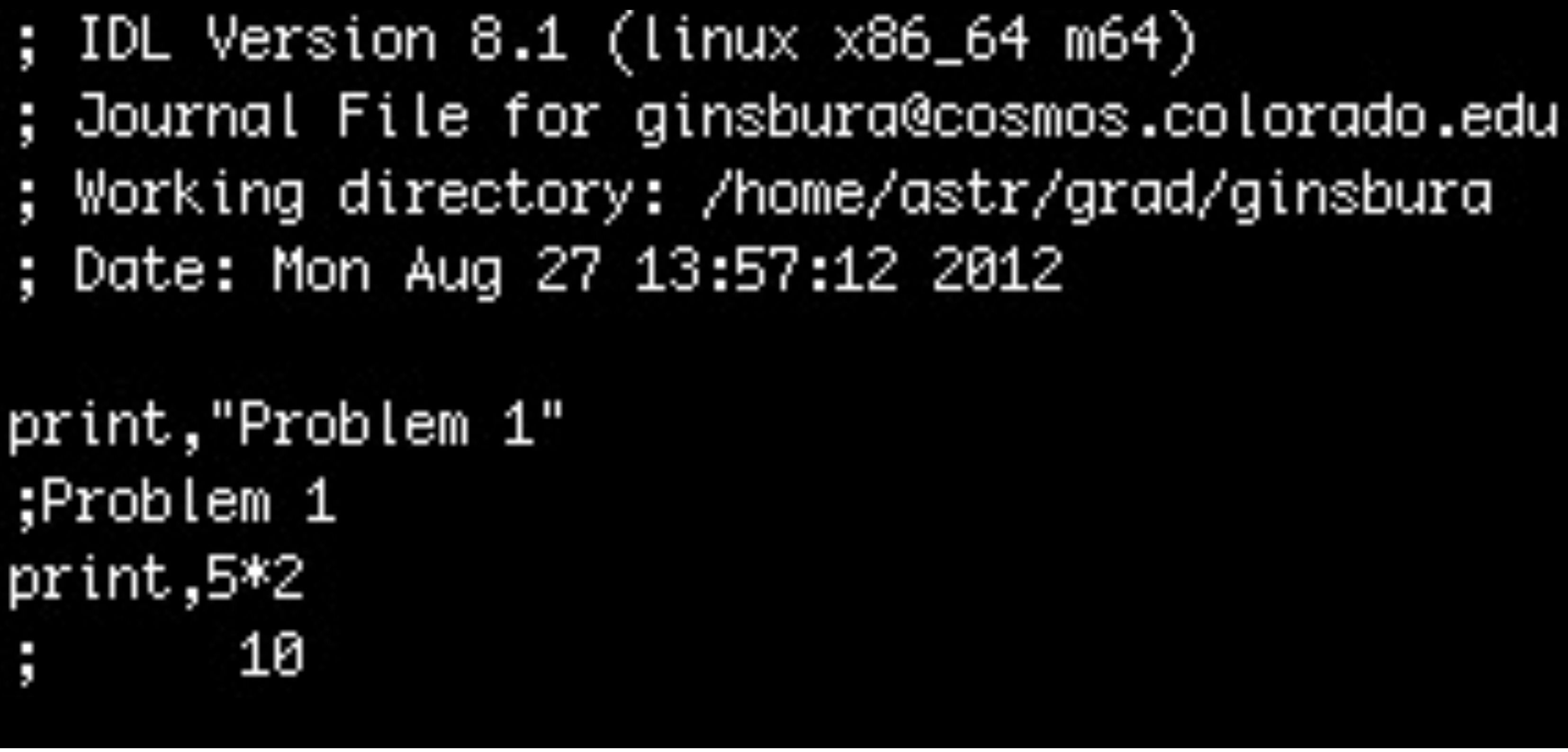
Journal: Saving your work

On the command line:



```
IDL> journal,"AdamGinsburg_GettingStartedWithIDL.pro"
IDL> print,"Problem 1"
Problem 1
IDL> print,5*2
      10
```

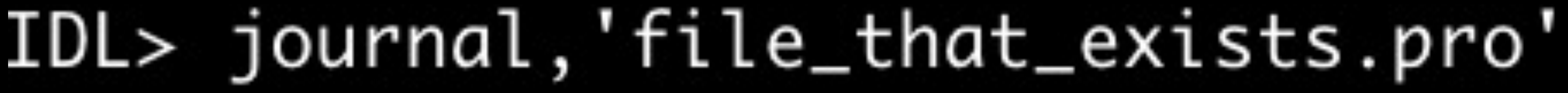
What gets saved:



```
; IDL Version 8.1 (linux x86_64 m64)
; Journal File for ginsbura@cosmos.colorado.edu
; Working directory: /home/astr/grad/ginsbura
; Date: Mon Aug 27 13:57:12 2012

print,"Problem 1"
;Problem 1
print,5*2
;      10
```

!!WARNING!!



```
IDL> journal,'file_that_exists.pro'
```

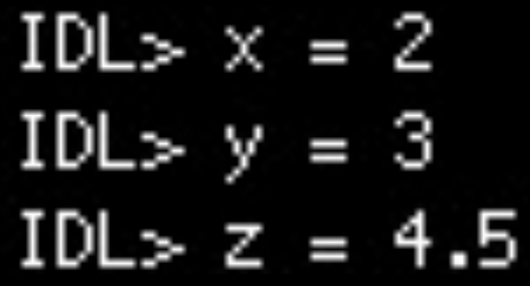
- Will overwrite `file_that_exists.pro`!
- Be cautious! If you want to continue where you left off, use:
  - `journal,"AdamGinsburg_ex0_part2.pro"`
- In-class “tutorial” today will cover backups

Where to get help

- The textbook
- The internet! But beware, there are "IDL" hits on google that are not the Interactive Data Language
    - I prefer searching here for IDL help: http://idlastro.gsfc.nasa.gov/
- Online help: Type ? at the command prompt, e.g. `IDL> ?print`

Variables cont’d

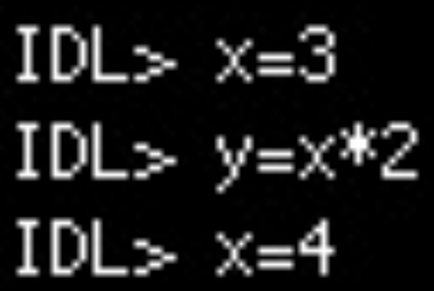
- Already covered “setting” variables



```
IDL> x = 2
IDL> y = 3
IDL> z = 4.5
```

- What about changing their values?

Modifying Variables



```
IDL> x=3
IDL> y=x*2
IDL> x=4
```

What is `y` now?

A) 3

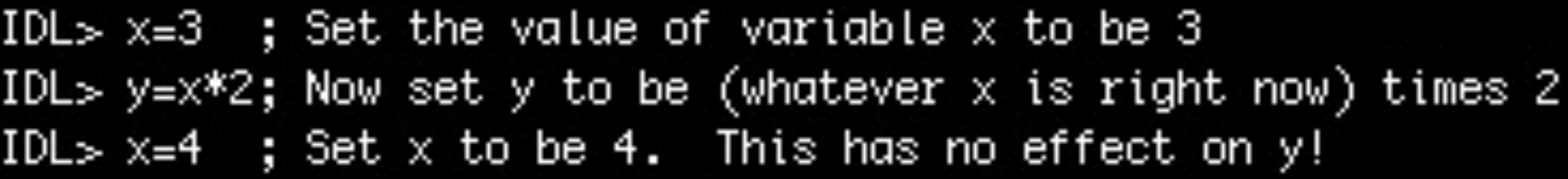
B) 4

C) 6

D) 8

E) None of the above / I don’t know

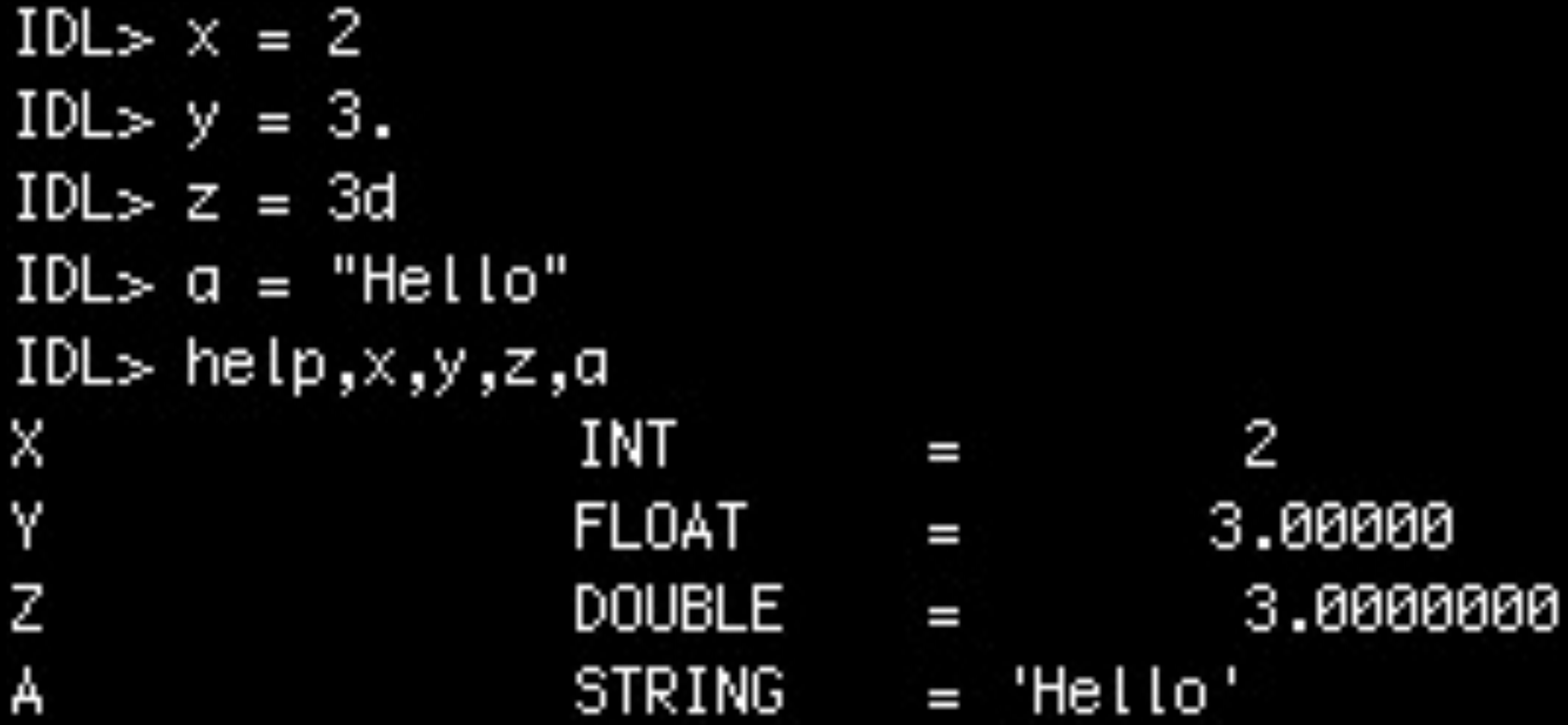
IDL vs Spreadsheets



```
IDL> x=3  ; Set the value of variable x to be 3
IDL> y=x*2; Now set y to be (whatever x is right now) times 2
IDL> x=4  ; Set x to be 4.  This has no effect on y!
```

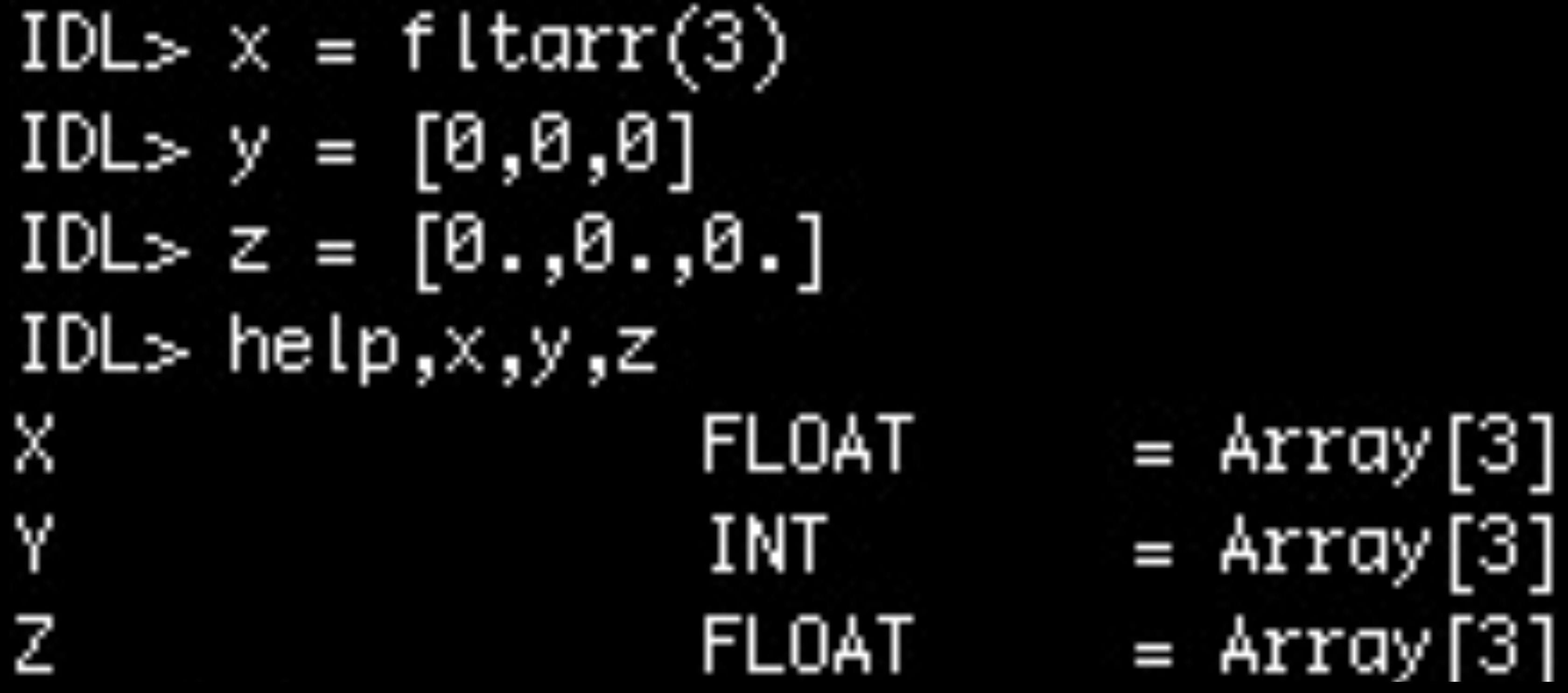
- In a spreadsheet, “cells” are all dynamically updated
- That’s not how IDL works, or programming languages in general.
- Why? In a spreadsheet, you can “see” all values at the same time. In IDL, you have to “print” to see the value

Variables:Types



```
IDL> x = 2
IDL> y = 3.
IDL> z = 3d
IDL> a = "Hello"
IDL> help,x,y,z,a
X               INT       =        2
Y               FLOAT     =       3.00000
Z               DOUBLE    =        3.0000000
A               STRING    = 'Hello'
```

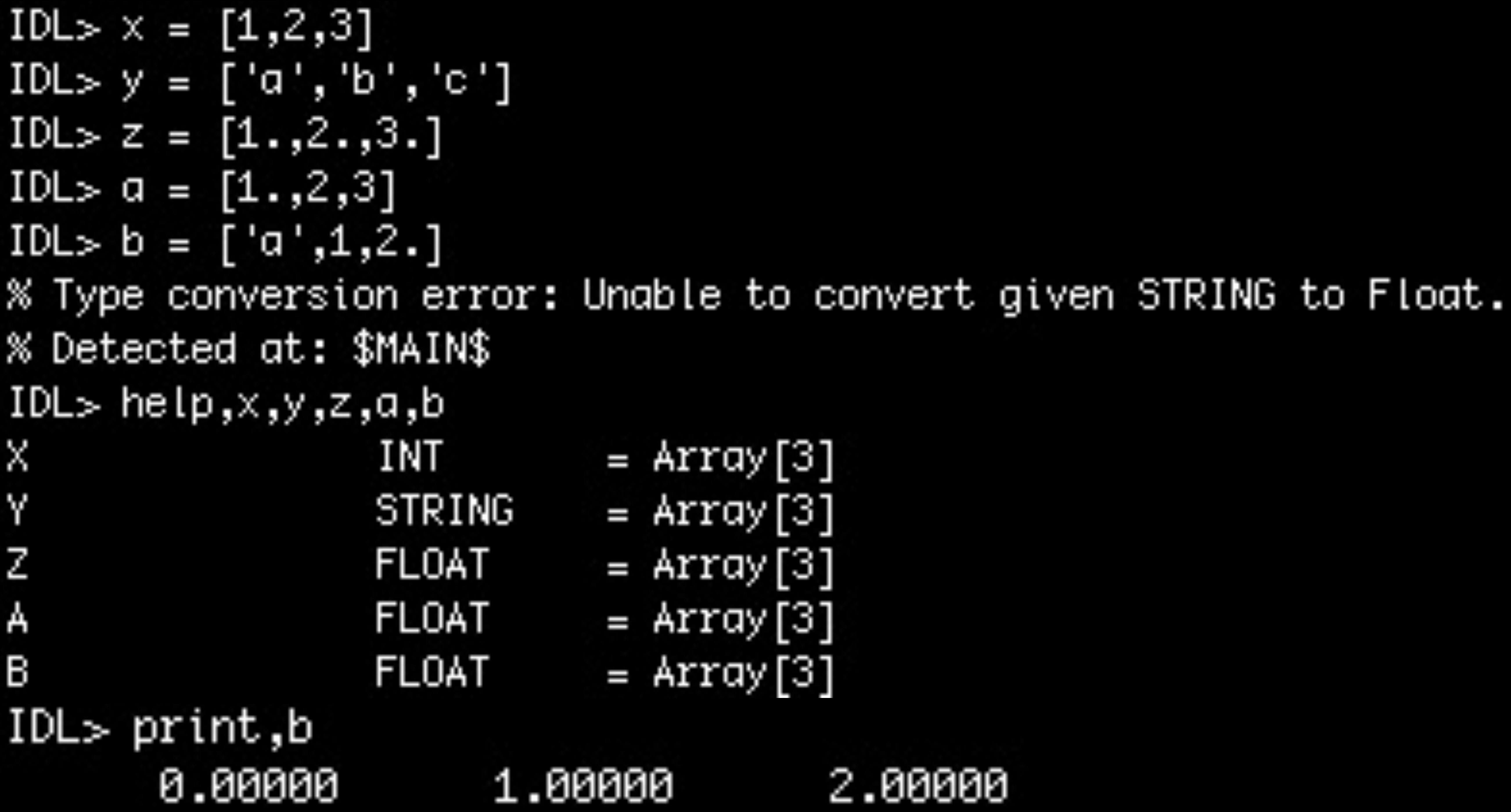
Arrays



```
IDL> x = fltarr(3)
IDL> y = [0,0,0]
IDL> z = [0.,0.,0.]
IDL> help,x,y,z
X               FLOAT     = Array[3]
Y               INT       = Array[3]
Z               FLOAT     = Array[3]
```

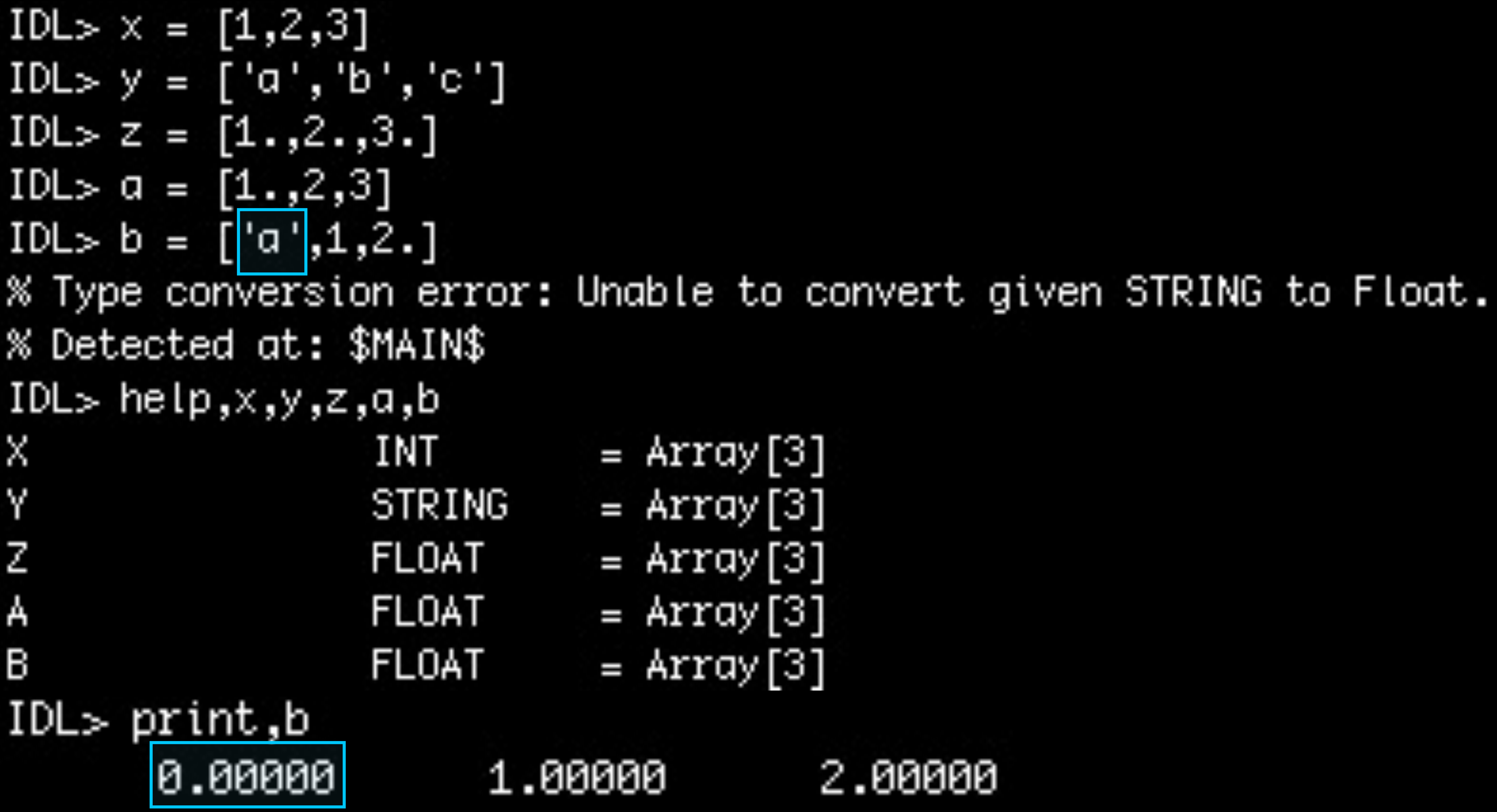
- Lists of numbers... but better

Variables and Arrays



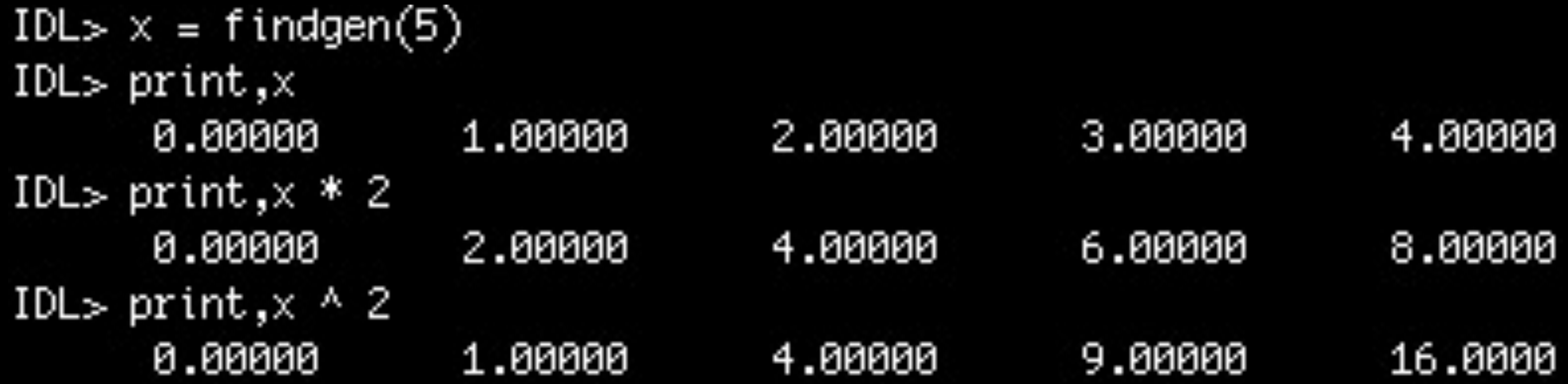
```
IDL> x = [1,2,3]
IDL> y = ['a','b','c']
IDL> z = [1.,2.,3.]
IDL> a = [1.,2,3]
IDL> b = ['a',1,2.]
% Type conversion error: Unable to convert given STRING to Float.
% Detected at: $MAIN$
IDL> help,x,y,z,a,b
X               INT       = Array[3]
Y               STRING    = Array[3]
Z               FLOAT     = Array[3]
A               FLOAT     = Array[3]
B               FLOAT     = Array[3]
IDL> print,b
      0.00000      1.00000      2.00000
```

Variables and Arrays



```
IDL> x = [1,2,3]
IDL> y = ['a','b','c']
IDL> z = [1.,2.,3.]
IDL> a = [1.,2,3]
IDL> b = ['a',1,2.]
% Type conversion error: Unable to convert given STRING to Float.
% Detected at: $MAIN$
IDL> help,x,y,z,a,b
X               INT       = Array[3]
Y               STRING    = Array[3]
Z               FLOAT     = Array[3]
A               FLOAT     = Array[3]
B               FLOAT     = Array[3]
IDL> print,b
      0.00000      1.00000      2.00000
```

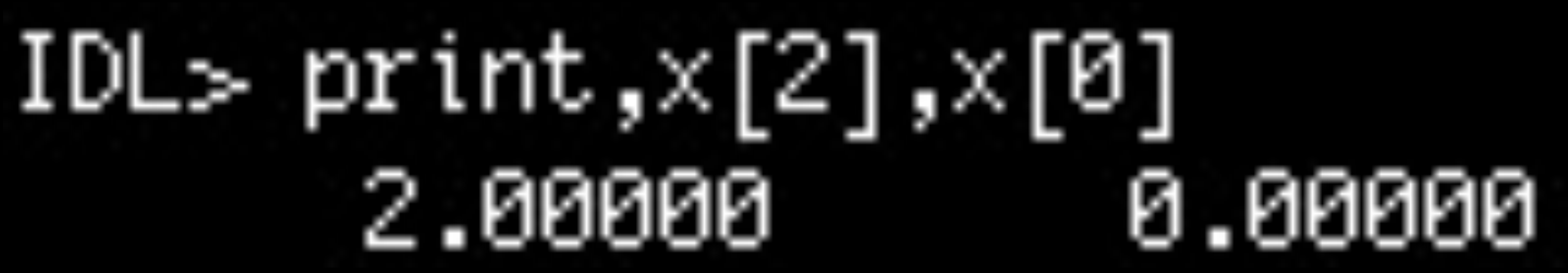
Arrays



```
IDL> x = findgen(5)
IDL> print,x
      0.00000      1.00000      2.00000      3.00000      4.00000
IDL> print,x * 2
      0.00000      2.00000      4.00000      6.00000      8.00000
IDL> print,x ^ 2
      0.00000      1.00000      4.00000      9.00000      16.0000
```

- “Elementwise” arithmetic

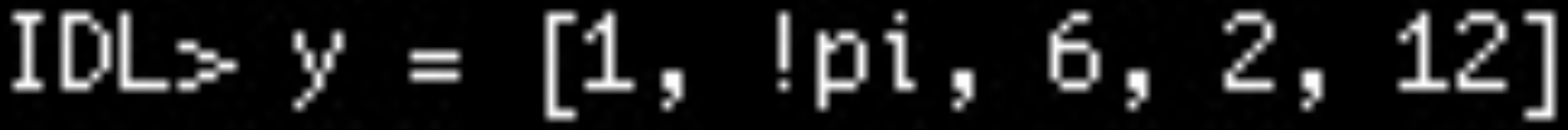
Arrays: Indexing



```
IDL> print,x[2],x[0]
      2.00000      0.00000
```

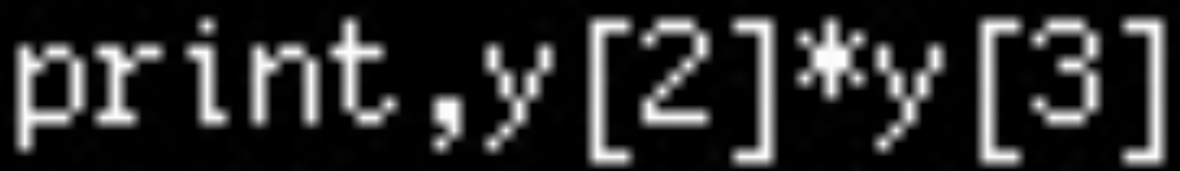
- Zero-based indexing (i.e. the “first” element is “element zero”)
    - Confusing but you’ll get used to it

Indexing



```
IDL> y = [1, !pi, 6, 2, 12]
```

Evaluate:



```
print,y[2]*y[3]
```

A) `12.0000`

B) 6 pi = `18.8496`

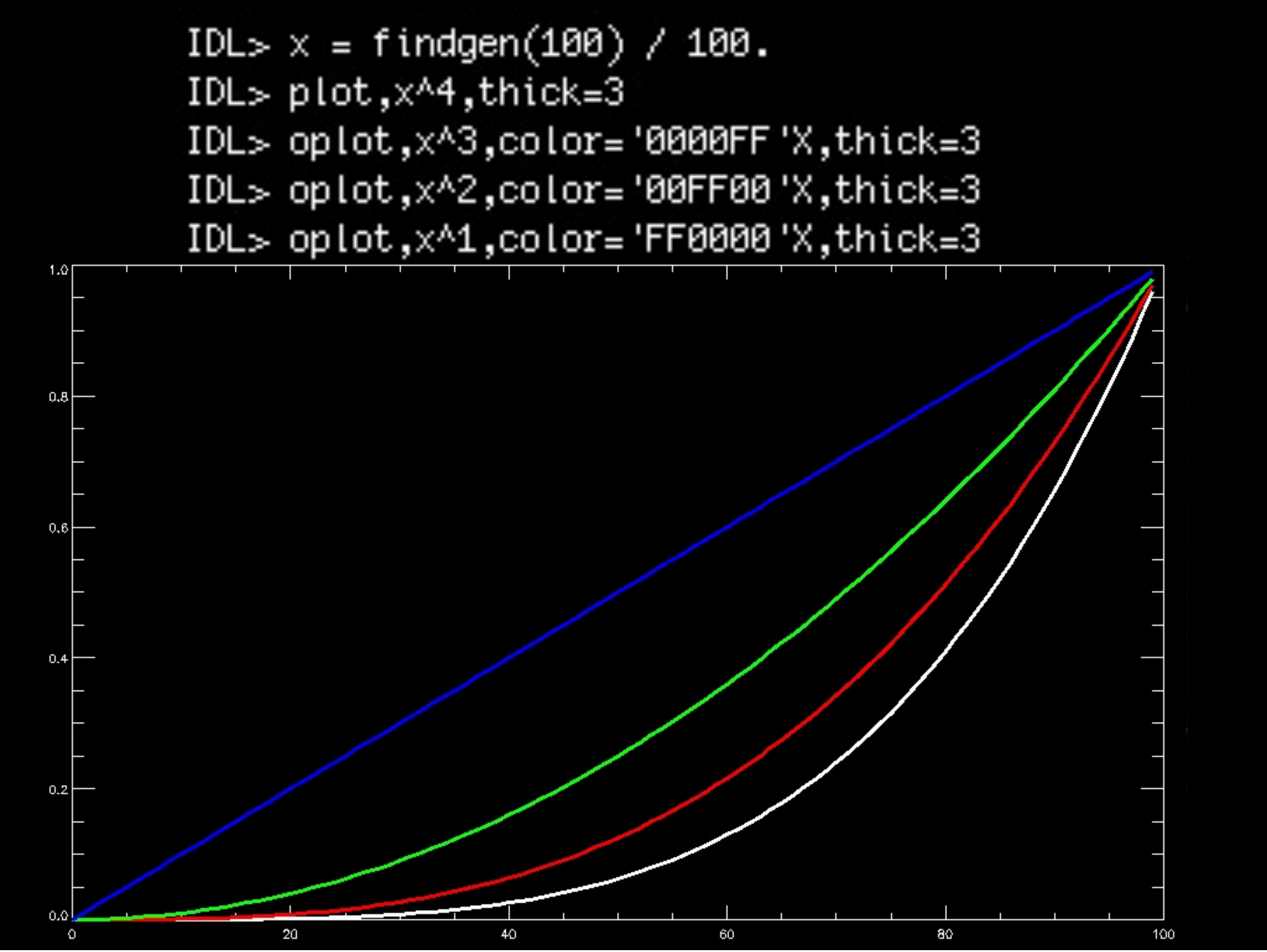
C) pi = `3.14159`

D) `24.0000`

E) I don’t know

Arrays: Plotting

- Arrays are an advanced topic, but I introduce them early so we can use them for plotting
- The "helper" functions `indgen`/ `findgen` produce arrays from 0 to n-1, e.g. `indgen(5) = [0,1,2,3,4]`
  - `intarr(5) = [0,0,0,0,0]`



```
IDL> x = findgen(100) / 100.
IDL> plot,x^4,thick=3
IDL> oplot,x^3,color='0000FF'X,thick=3
IDL> oplot,x^2,color='00FF00'X,thick=3
IDL> oplot,x^1,color='FF0000'X,thick=3
```

Math Functions

- Natural logarithm: `alog`
- Log base 10: `alog10`
- Trig functions: `sin, cos, tan`
- Exponential: `exp`

To the lab!

- “Getting Started with IDL” handout
- Github signup handout
- Homework handout
    - Including class survey