C.E.C.I

Consortium des Equipements
de Calcul Intensif
en Fédération Wallonie-Bruxelles

Introduction to Scripting Languages

damien.francois@uclouvain.be

October 2017

UCL Université catholique de Louvain

INSTITUT DE CALCUL INTENSIF ET DE STOCKAGE DE MASSE

Goal of this session:

“Advocate the use of scripting languages and help you choose the most suitable for your needs”

Agenda

1. Interpreters vs compilers
2. Octave, R, Python
3. GUIs & Literate programming
4. Packages/Libraries/Modules
5. When it is too slow
6. Bridges

Interpreters vs Compilers

- A **compiler** reads the whole code and produces a separate binary file that can be executed by the CPU.

  C/C++, Fortran, Java, Go, Haskel, ...

- An **interpreter** reads each line of code and executes it by calling the corresponding functionalities in its own code.

  Bash, Python, PHP, Javascript, Ruby, ...

Interpreters vs Compilers

- The ugly truth...
  - Many interpreters will pre-compile the code
  - Some compilers compile not to CPU-specific machine instructions but to bytecode
  - The bytecode interpreters sometimes re-compile the bytecode just before execution (JIT compiling)
  - Interpreters exist for C and C++
  - Compilers exist for Python
  - The interpreter can be compiled or himself interpreted

Interpreters vs Compilers

Compilers

- can apply code-wise powerful optimization
- practically have no run-time overhead

→ Speed

Interpreters

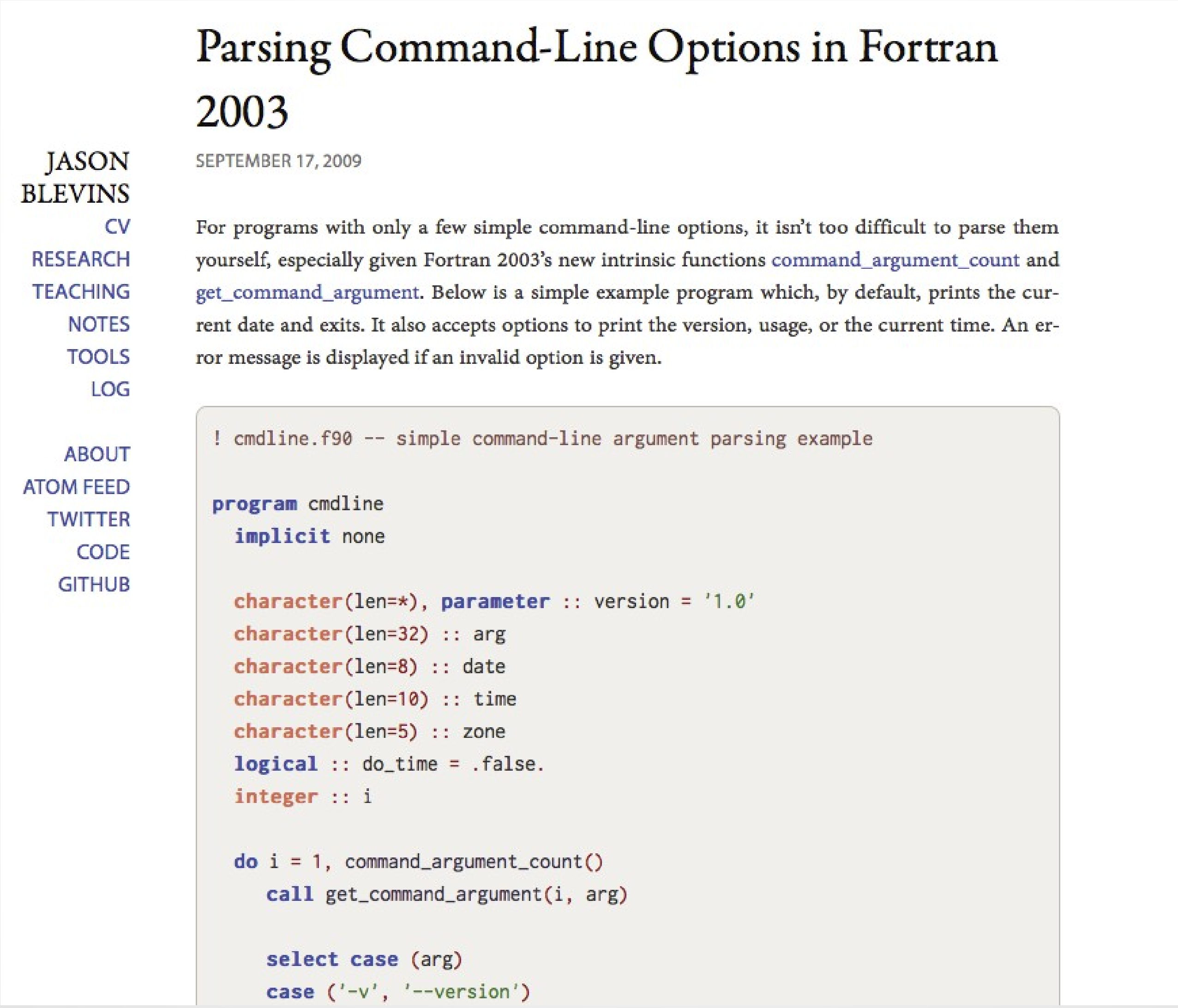
- allow easy code introspection
- offer high-level language constructs and tools

→ Ease of use

Interpreted languages

- Easier to **learn**
  - Many implementation details hidden
  - Can try and test code portions rapidly and easily
- Easier to **exchange**/reuse
  - The scripts are cross-platform by design
  - Often built-in package management
- Faster development
  - More **convenient programming** and shorter programs
    - Offers many simplifications and shortcuts – no need to micromanage memory
    - Built-in support for mundane tasks (handle files, dates, plots, Nas, NANs, etc.)
  - **Easier to debug** and profile
    - GUI

Ex.1: argument parsing in Fortran



Parsing Command-Line Options in Fortran 2003

SEPTEMBER 17, 2009

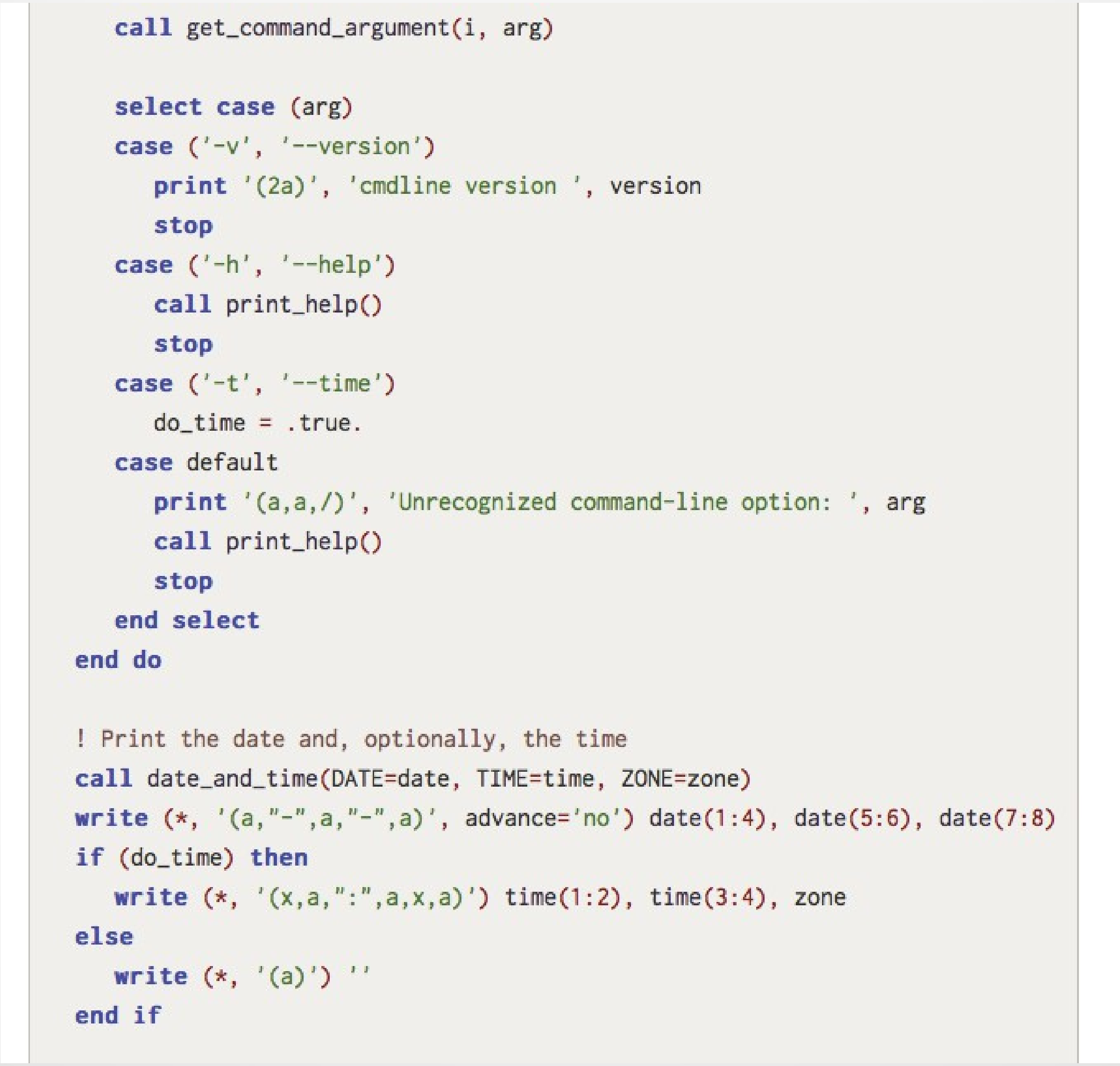
JASON BLEVINS

For programs with only a few simple command-line options, it isn't too difficult to parse them yourself, especially given Fortran 2003's new intrinsic functions command_argument_count and get_command_argument. Below is a simple example program which, by default, prints the current date and exits. It also accepts options to print the version, usage, or the current time. An error message is displayed if an invalid option is given.

```fortran
! cmdline.f90 -- simple command-line argument parsing example

program cmdline
  implicit none

  character(len=*), parameter :: version = '1.0'
  character(len=32) :: arg
  character(len=8) :: date
  character(len=10) :: time
  character(len=5) :: zone
  logical :: do_time = .false.
  integer :: i

  do i = 1, command_argument_count()
     call get_command_argument(i, arg)

     select case (arg)
     case ('-v', '--version')
```

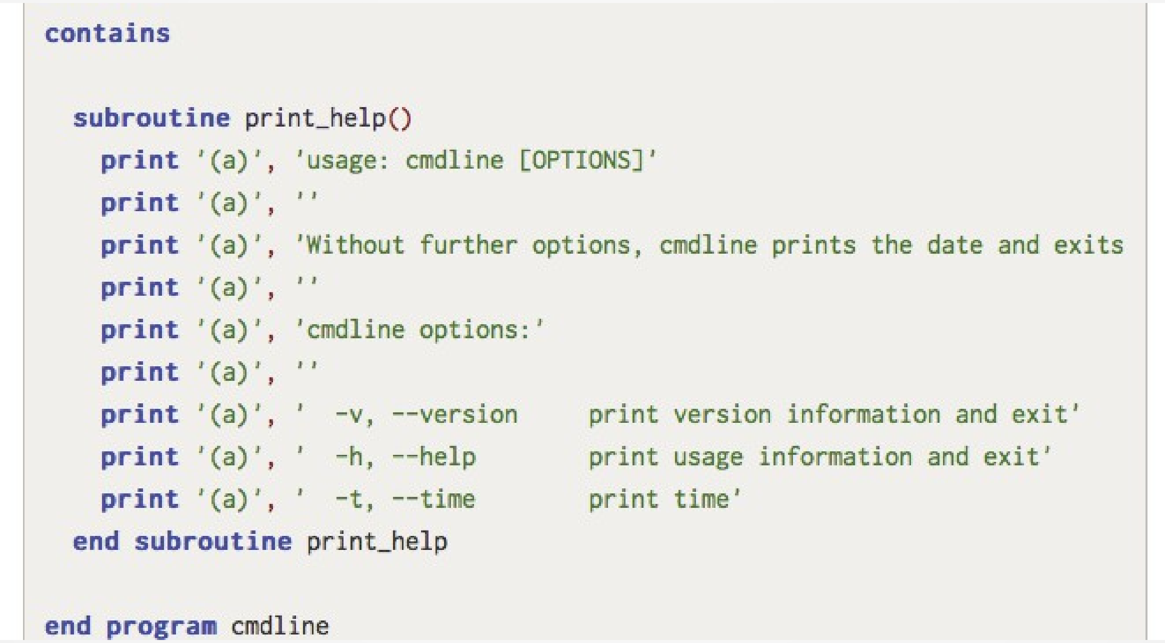
https://docs.python.org/3/library/argparse.html

Ex.1: argument parsing in Fortran



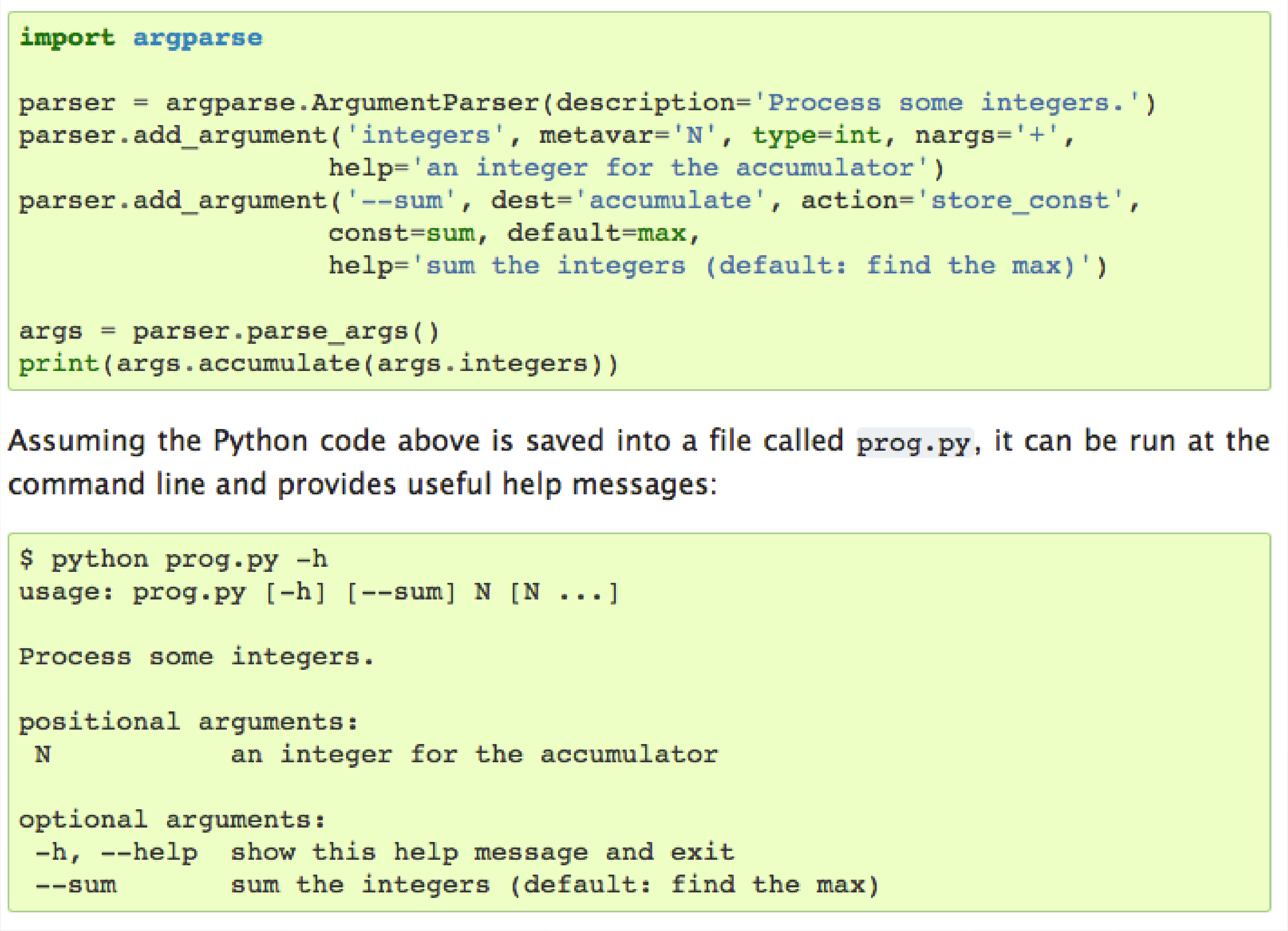
```
   call get_command_argument(i, arg)

   select case (arg)
   case ('-v', '--version')
      print '(2a)', 'cmdline version ', version
      stop
   case ('-h', '--help')
      call print_help()
      stop
   case ('-t', '--time')
      do_time = .true.
   case default
      print '(a,a,/)', 'Unrecognized command-line option: ', arg
      call print_help()
      stop
   end select
end do

! Print the date and, optionally, the time
call date_and_time(DATE=date, TIME=time, ZONE=zone)
write (*, '(a,"-",a,"-",a)', advance='no') date(1:4), date(5:6), date(7:8)
if (do_time) then
   write (*, '(x,a,":",a,x,a)') time(1:2), time(3:4), zone
else
   write (*, '(a)') ''
end if
```

Ex.1: argument parsing in Fortran



```fortran
contains

  subroutine print_help()
    print '(a)', 'usage: cmdline [OPTIONS]'
    print '(a)', ''
    print '(a)', 'Without further options, cmdline prints the date and exits
    print '(a)', ''
    print '(a)', 'cmdline options:'
    print '(a)', ''
    print '(a)', '  -v, --version     print version information and exit'
    print '(a)', '  -h, --help        print usage information and exit'
    print '(a)', '  -t, --time        print time'
  end subroutine print_help

end program cmdline
```

Ex.1: argument parsing in Python



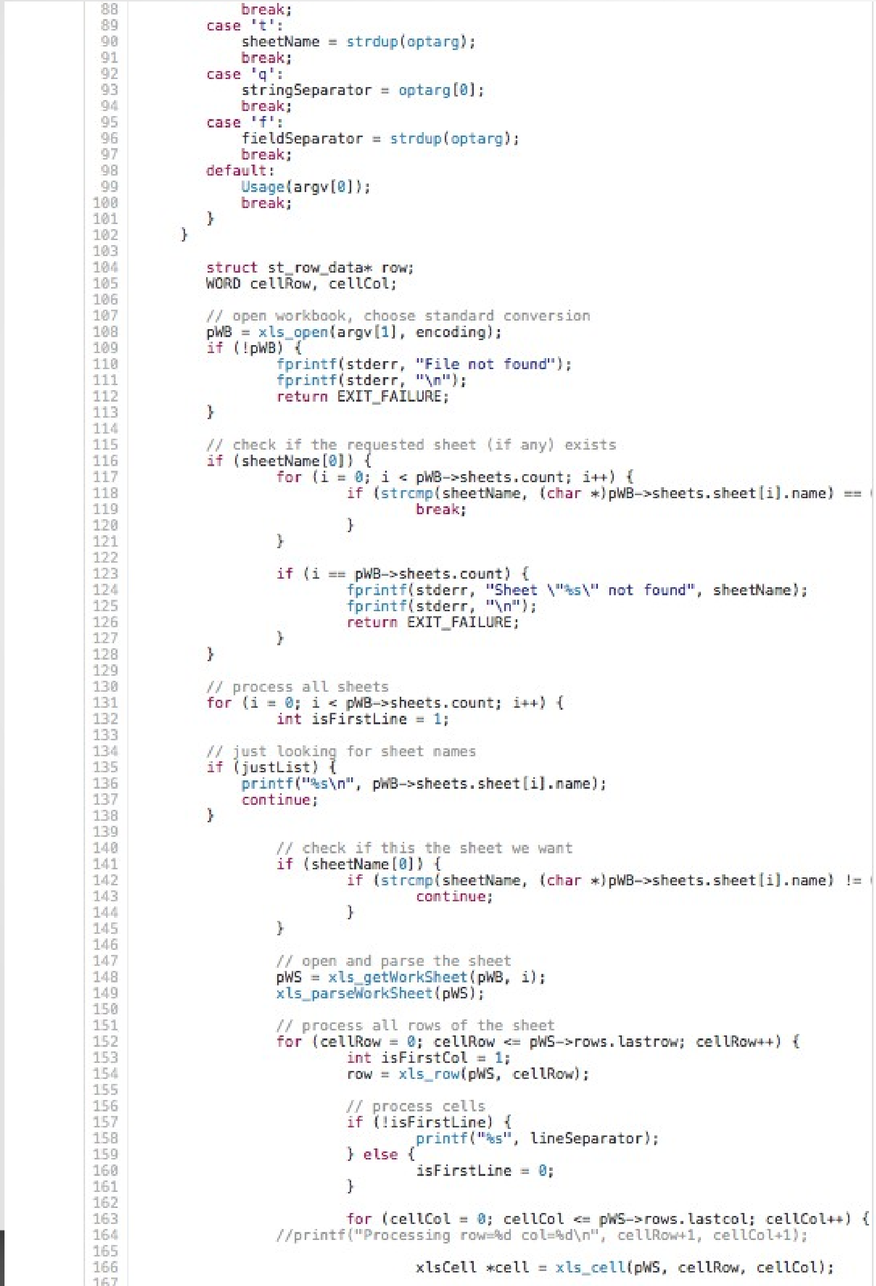
```
import argparse

parser = argparse.ArgumentParser(description='Process some integers.')
parser.add_argument('integers', metavar='N', type=int, nargs='+',
                   help='an integer for the accumulator')
parser.add_argument('--sum', dest='accumulate', action='store_const',
                   const=sum, default=max,
                   help='sum the integers (default: find the max)')

args = parser.parse_args()
print(args.accumulate(args.integers))
```

Assuming the Python code above is saved into a file called `prog.py`, it can be run at the command line and provides useful help messages:

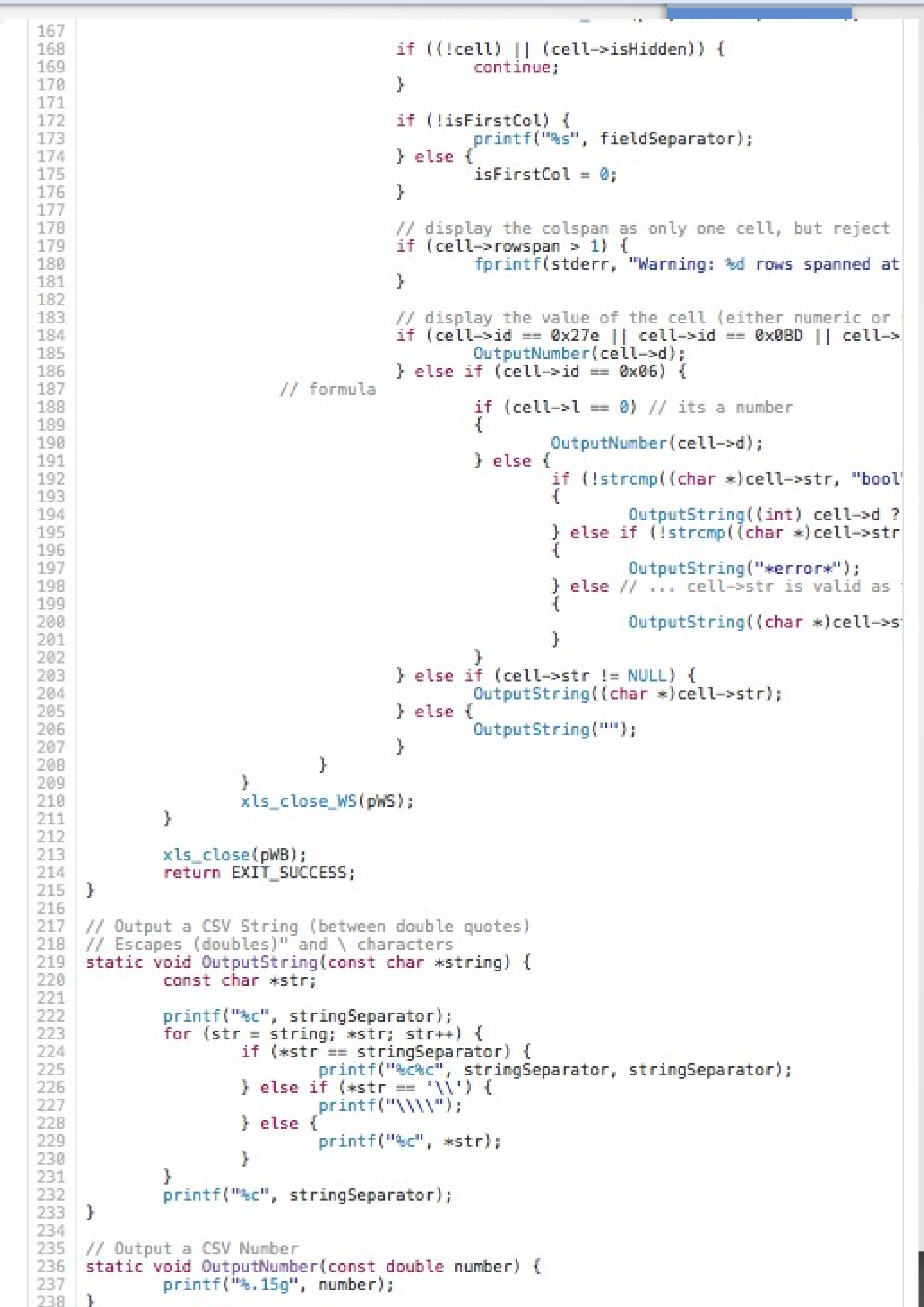
```
$ python prog.py -h
usage: prog.py [-h] [--sum] N [N ...]

Process some integers.

positional arguments:
 N           an integer for the accumulator

optional arguments:
 -h, --help  show this help message and exit
 --sum       sum the integers (default: find the max)
```

https://docs.python.org/3/library/argparse.html

Ex.2: Use XLS file in C

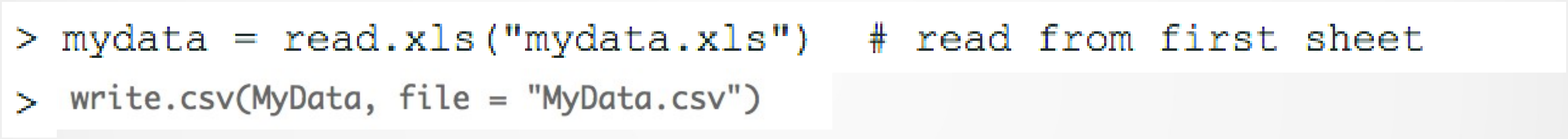


```
            break;
    case 't':
        sheetName = strdup(optarg);
        break;
    case 'q':
        stringSeparator = optarg[0];
        break;
    case 'f':
        fieldSeparator = strdup(optarg);
        break;
    default:
        Usage(argv[0]);
        break;
    }
}

    struct st_row_data* row;
    WORD cellRow, cellCol;

    // open workbook, choose standard conversion
    pWB = xls_open(argv[1], encoding);
    if (!pWB) {
            fprintf(stderr, "File not found");
            fprintf(stderr, "\n");
            return EXIT_FAILURE;
    }

    // check if the requested sheet (if any) exists
    if (sheetName[0]) {
            for (i = 0; i < pWB->sheets.count; i++) {
                    if (strcmp(sheetName, (char *)pWB->sheets.sheet[i].name) ==
                            break;
                    }
            }

            if (i == pWB->sheets.count) {
                    fprintf(stderr, "Sheet \"%s\" not found", sheetName);
                    fprintf(stderr, "\n");
                    return EXIT_FAILURE;
            }
    }

    // process all sheets
    for (i = 0; i < pWB->sheets.count; i++) {
            int isFirstLine = 1;

    // just looking for sheet names
    if (justList) {
        printf("%s\n", pWB->sheets.sheet[i].name);
        continue;
    }

            // check if this the sheet we want
            if (sheetName[0]) {
                    if (strcmp(sheetName, (char *)pWB->sheets.sheet[i].name) !=
                            continue;
                    }
            }

            // open and parse the sheet
            pWS = xls_getWorkSheet(pWB, i);
            xls_parseWorkSheet(pWS);

            // process all rows of the sheet
            for (cellRow = 0; cellRow <= pWS->rows.lastrow; cellRow++) {
                    int isFirstCol = 1;
                    row = xls_row(pWS, cellRow);

                    // process cells
                    if (!isFirstLine) {
                            printf("%s", lineSeparator);
                    } else {
                            isFirstLine = 0;
                    }

                    for (cellCol = 0; cellCol <= pWS->rows.lastcol; cellCol++) {
            //printf("Processing row=%d col=%d\n", cellRow+1, cellCol+1);

                            xlsCell *cell = xls_cell(pWS, cellRow, cellCol);
```



```
                                        if ((!cell) || (cell->isHidden)) {
                                                continue;
                                        }

                                        if (!isFirstCol) {
                                                printf("%s", fieldSeparator);
                                        } else {
                                                isFirstCol = 0;
                                        }

                                        // display the colspan as only one cell, but reject
                                        if (cell->rowspan > 1) {
                                                fprintf(stderr, "Warning: %d rows spanned at
                                        }

                                        // display the value of the cell (either numeric or
                                        if (cell->id == 0x27e || cell->id == 0x0BD || cell->
                                                OutputNumber(cell->d);
                                        } else if (cell->id == 0x06) {
                          // formula
                                                if (cell->l == 0) // its a number
                                                {
                                                        OutputNumber(cell->d);
                                                } else {
                                                        if (!strcmp((char *)cell->str, "bool'
                                                        {
                                                                OutputString((int) cell->d ?
                                                        } else if (!strcmp((char *)cell->str
                                                        {
                                                                OutputString("*error*");
                                                        } else // ... cell->str is valid as
                                                        {
                                                                OutputString((char *)cell->s
                                                        }
                                                }
                                        } else if (cell->str != NULL) {
                                                OutputString((char *)cell->str);
                                        } else {
                                                OutputString("");
                                        }
                                }
                        }
                        xls_close_WS(pWS);
                }

        xls_close(pWB);
        return EXIT_SUCCESS;
}

// Output a CSV String (between double quotes)
// Escapes (doubles)" and \ characters
static void OutputString(const char *string) {
        const char *str;

        printf("%c", stringSeparator);
        for (str = string; *str; str++) {
                if (*str == stringSeparator) {
                        printf("%c%c", stringSeparator, stringSeparator);
                } else if (*str == '\\') {
                        printf("\\\\");
                } else {
                        printf("%c", *str);
                }
        }
        printf("%c", stringSeparator);
}

// Output a CSV Number
static void OutputNumber(const double number) {
        printf("%.15g", number);
}
```

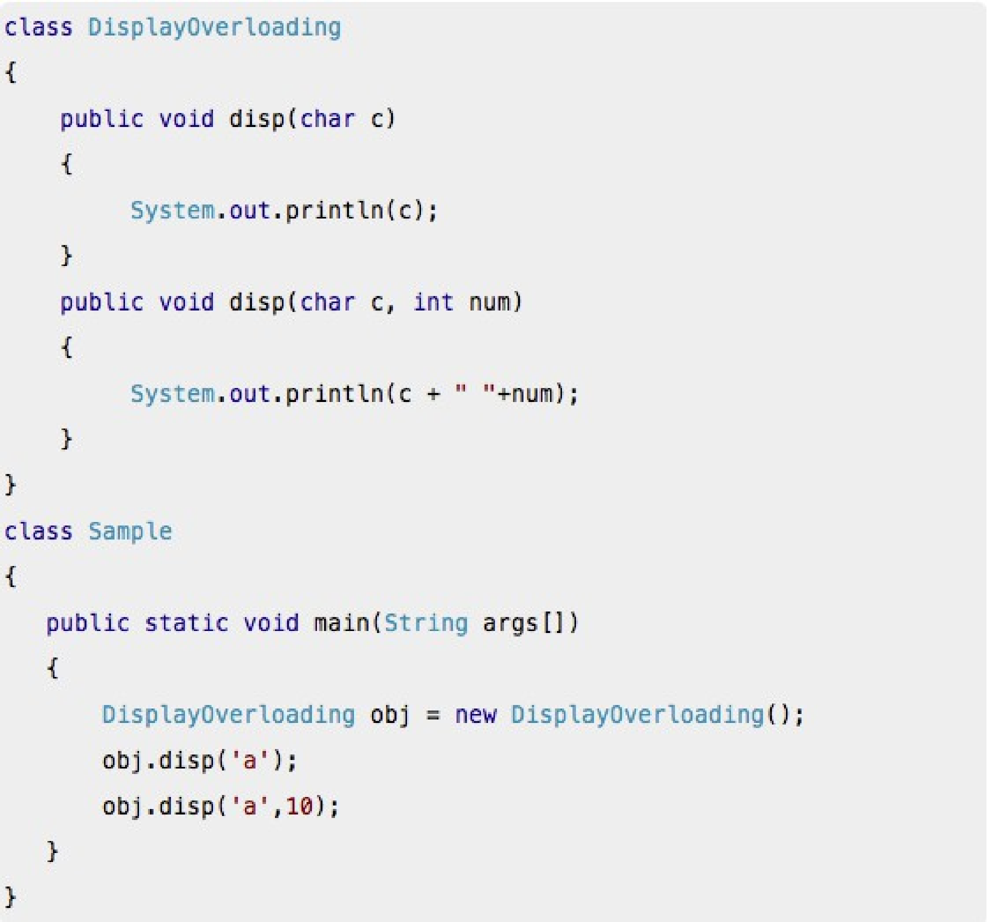
Ex.2: Use XLS file in R



```
> mydata = read.xls("mydata.xls")  # read from first sheet
> write.csv(MyData, file = "MyData.csv")
```

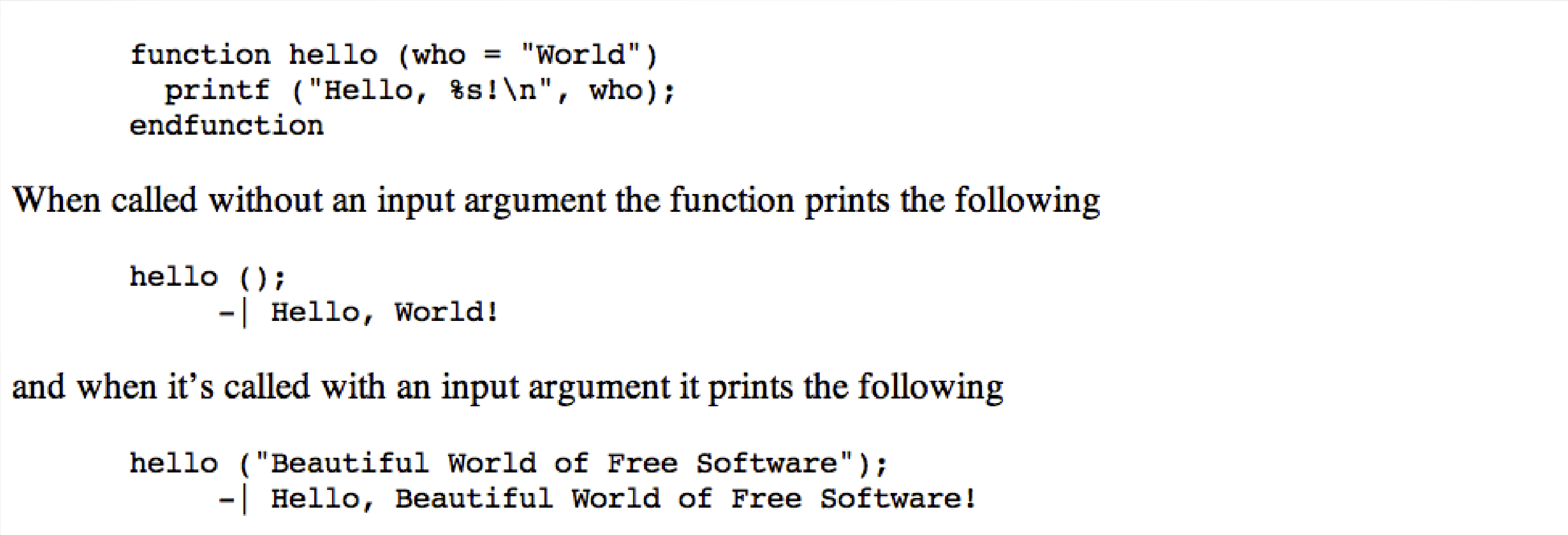
https://cran.r-project.org/web/packages/gdata/

Ex.3: default args in Java



```
class DisplayOverloading
{
    public void disp(char c)
    {
         System.out.println(c);
    }
    public void disp(char c, int num)
    {
         System.out.println(c + " "+num);
    }
}
class Sample
{
   public static void main(String args[])
   {
       DisplayOverloading obj = new DisplayOverloading();
       obj.disp('a');
       obj.disp('a',10);
   }
}
```

Ex.3: default args in Octave



```
function hello (who = "World")
  printf ("Hello, %s!\n", who);
endfunction
```

When called without an input argument the function prints the following

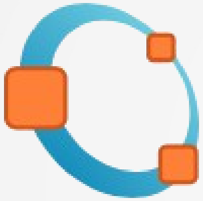
```
hello ();
     -| Hello, World!
```

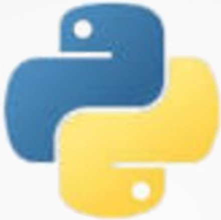
and when it’s called with an input argument it prints the following

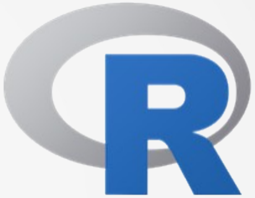
```
hello ("Beautiful World of Free Software");
     -| Hello, Beautiful World of Free Software!
```

https://www.gnu.org/software/octave/doc/interpreter/Default-Arguments.html

1.







Why those three?

Why those three?

- All very much used in scientific applications

  R (S/SPlus): strong for statistics

  Octave (Matlab): strong for engineering

  Python Scipy/Numpy (Canopy,Anaconda): strong for data science
- All free and free.
- Fun fact: All started as wrappers for Fortran code!

Why those three?

S was designed by John Chambers (Bell Lags) as an interactive interface to a Fortran-callable library, ca 1976.

MATLAB was built by Cleve Moler (University of New Mexico) to give students access to LINPACK and EISPACK without them having to learn Fortran

Python Numpy (Travis Oliphant, Brigham Young University) originates from f2py, a tool to easily extend Python with Fortran code.

Why those three?

Octave: Fortran optimized routines made easy to use. Easily handle (multi-dimensional) matrices, Nans, Infs, no need to worry about memory allocation, etc.

R: Easily handle matrices, strings, dates, and categories and missing values

Python: Full programming language, can handle custom objects

Why those three?

By contrast,

- Ruby, Perl: smaller bioinformatics-only community
- Javascript, PHP, Bash, TCL, Lua: totally different goal
- Matlab, IDL, Mathematica: not free
- Julia: very young – good luck to get help when needed

Why those three?

By contrast,

- Ruby, Perl: smaller bioinformatics-only community
- Javascript, PHP, Bash, TCL, Lua: totally different goal
- Matlab, IDL, Mathematica: not free
- Julia: ~~very young – good luck to get help when needed~~

Not true anymore.
Worth considering !

(but not yet in this session...)

2.

TripleQuickstart

Operators and assignment

```
a=1; b=2;
a + b
a - b
a * b
a / b
a .^ b


rem(a,b)



a(:,1) = 99
a(:,1) = [99 98 97]'
a(a>90) = 90;
```

```
a=1; b=1
a + b or add(a,b)
a - b or subtract(a,b)
a * b or multiply(a,b)
a / b or divide(a,b)
a ** b
power(a,b)
pow(a,b)
a % b
remainder(a,b)
fmod(a,b)


a[:,0] = 99
a[:,0] = array([99,98,97])
(a>90).choose(a,90)
a.clip(min=None, max=90)

a.clip(min=2, max=5)
```

```
a<-1; b<-2
a + b
a - b
a * b
a / b
a ^ b


a %% b



a[,1] <- 99
a[,1] <- c(99,98,97)
a[a>90] <- 90
```

http://sebastianraschka.com/Articles/2014_matrix_cheatsheet_table.html

Building arrays/matrices

```
a=[2 3 4 5];
adash=[2 3 4 5]';
```

```
1:10

0:9
1:3:10
10:-1:1
10:-3:1
linspace(1,10,7)
reverse(a)
a(:) = 3
```

```
a=array([2,3,4,5])
array([2,3,4,5])[:,NewAxis]
array([2,3,4,5]).reshape(-1,1)
r_[1:10,'c']
```

```
arange(1,11, dtype=Float)
range(1,11)
arange(10.)
arange(1,11,3)
arange(10,0,-1)
arange(10,0,-3)
linspace(1,10,7)
a[::-1] or
a.fill(3), a[:] = 3
```

```
a <- c(2,3,4,5)
adash <- t(c(2,3,4,5))
```

```
seq(10) or 1:10

seq(0,length=10)
seq(1,10,by=3)
seq(10,1) or 10:1
seq(from=10,to=1,by=-3)
seq(1,10,length=7)
rev(a)
```

http://sebastianraschka.com/Articles/2014_matrix_cheatsheet_table.html

Indexing/slicing

		R
`a(2,3)`	`a[1,2]`	`a[2,3]`
`a(1,:)`	`a[0,]`	`a[1,]`
`a(:,1)`	`a[:,0]`	`a[,1]`
`a([1 3],[1 4]);`	`a.take([0,2]).take([0,3], axis=1)`	
`a(2:end,:)`	`a[1:,]`	`a[-1,]`
`a(end-1:end,:)`	`a[-2:,]`	
`a(1:2:end,:)`	`a[::2,:]`	
	`a[...,2]`	
		`a[-2,-3]`
`a(:,[1 3 4])`	`a.take([0,2,3],axis=1)`	`a[,-2]`
	`a.diagonal(offset=0)`	

http://sebastianraschka.com/Articles/2014_matrix_cheatsheet_table.html

Searching arrays/matrices

		R
`find(a)`	`a.ravel().nonzero()`	`which(a != 0)`

`[i j] = find(a)`	`(i,j) = a.nonzero()`
	`(i,j) = where(a!=0)`
		`which(a != 0, arr.ind=T)`

`[i j v] = find(a)`	`v = a.compress((a!=0).flat)`
	`v = extract(a!=0,a)`
		`ij <- which(a != 0, arr.ind=T); v <- a[ij]`

`find(a>5.5)`	`(a>5.5).nonzero()`	`which(a>5.5)`
	`a.compress((a>5.5).flat)`	`ij <- which(a>5.5, arr.ind=T); v <- a[ij]`

`a .* (a>5.5)`	`where(a>5.5,0,a) or a * (a>5.5)`
	`a.put(2,indices)`	

http://mathesaurus.sourceforge.net/matlab-python-xref.pdf

Control structures

```
for i=1:5; disp(i); end
for i=1:5
   disp(i)
   disp(i*2)
end
```

MATLAB/Octave

```
if 1>0 a=100; end
if 1>0 a=100; else a=0; end
```

```
for i in range(1,6): print(i)
for i in range(1,6):
    print(i)
    print(i*2)
```

Python

```
if 1>0: a=100
```

```
for(i in 1:5) print(i)
for(i in 1:5) {
    print(i)
    print(i*2)
}
```

R

```
if (1>0) a <- 100

ifelse(a>0,a,0)
```

http://mathesaurus.sourceforge.net/matlab-python-xref.pdf

Linear regression

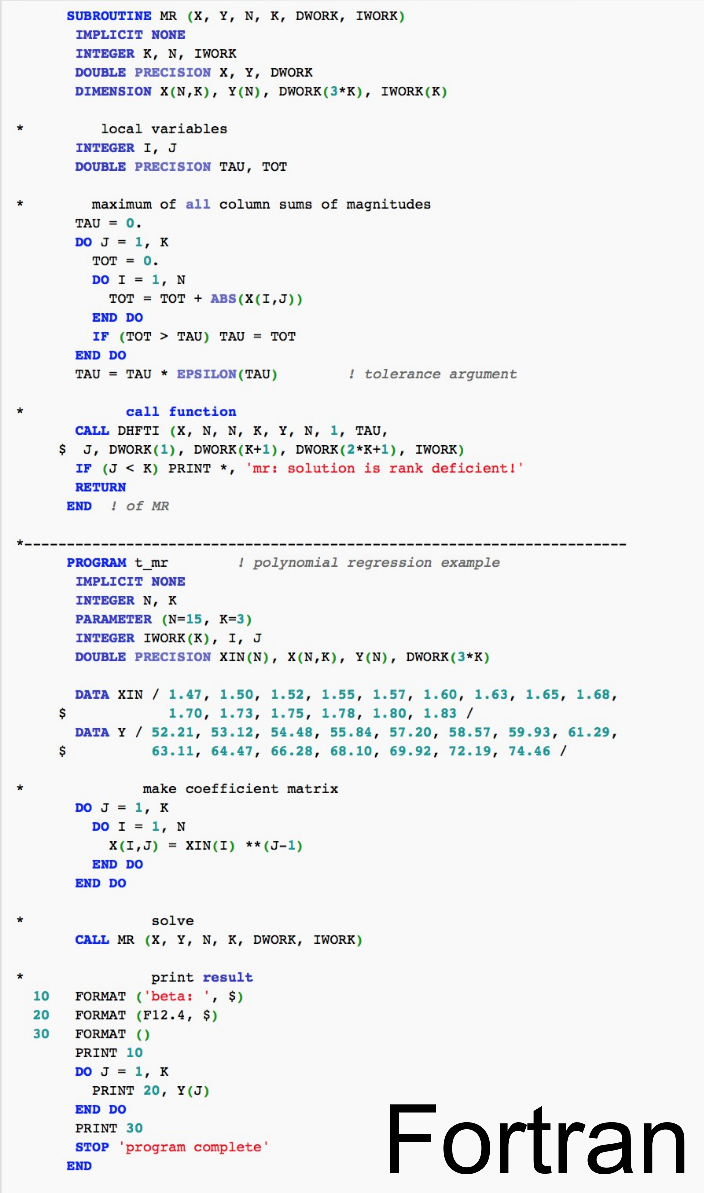
```
z = polyval(polyfit(x,y,1),x)
plot(x,y,’o’, x,z ,’-’)

a = x\y
```

```
(a,b) = polyfit(x,y,1)
plot(x,y,’o’, x,a*x+b,’-’)

linalg.lstsq(x,y)
```

```
z <- lm(y˜x)
plot(x,y)
abline(z)
solve(a,b)
```

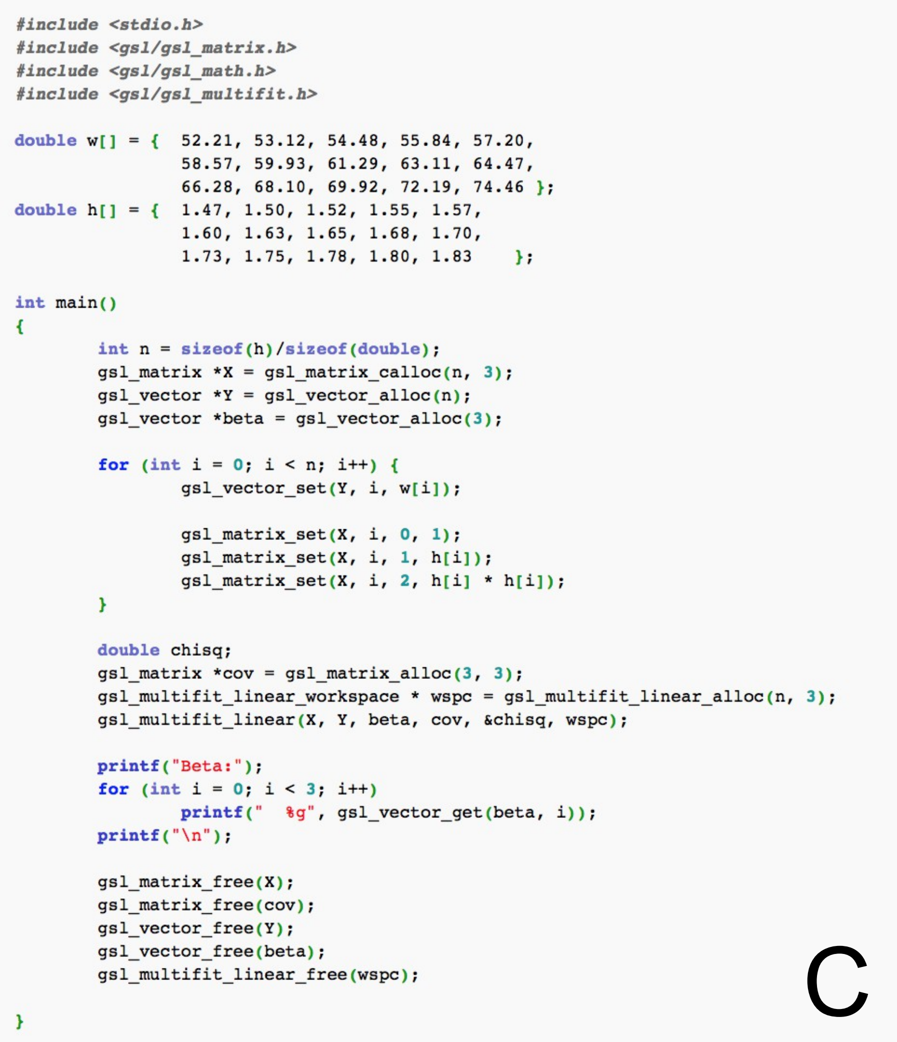
http://mathesaurus.sourceforge.net/matlab-python-xref.pdf

Linear regression



```fortran
      SUBROUTINE MR (X, Y, N, K, DWORK, IWORK)
       IMPLICIT NONE
       INTEGER K, N, IWORK
       DOUBLE PRECISION X, Y, DWORK
       DIMENSION X(N,K), Y(N), DWORK(3*K), IWORK(K)

*         local variables
       INTEGER I, J
       DOUBLE PRECISION TAU, TOT

*        maximum of all column sums of magnitudes
       TAU = 0.
       DO J = 1, K
         TOT = 0.
         DO I = 1, N
           TOT = TOT + ABS(X(I,J))
         END DO
         IF (TOT > TAU) TAU = TOT
       END DO
       TAU = TAU * EPSILON(TAU)        ! tolerance argument

*            call function
       CALL DHFTI (X, N, N, K, Y, N, 1, TAU,
     $  J, DWORK(1), DWORK(K+1), DWORK(2*K+1), IWORK)
       IF (J < K) PRINT *, 'mr: solution is rank deficient!'
       RETURN
      END  ! of MR

*-----------------------------------------------------------------------
      PROGRAM t_mr        ! polynomial regression example
       IMPLICIT NONE
       INTEGER N, K
       PARAMETER (N=15, K=3)
       INTEGER IWORK(K), I, J
       DOUBLE PRECISION XIN(N), X(N,K), Y(N), DWORK(3*K)

       DATA XIN / 1.47, 1.50, 1.52, 1.55, 1.57, 1.60, 1.63, 1.65, 1.68,
     $            1.70, 1.73, 1.75, 1.78, 1.80, 1.83 /
       DATA Y / 52.21, 53.12, 54.48, 55.84, 57.20, 58.57, 59.93, 61.29,
     $          63.11, 64.47, 66.28, 68.10, 69.92, 72.19, 74.46 /

*              make coefficient matrix
       DO J = 1, K
         DO I = 1, N
           X(I,J) = XIN(I) **(J-1)
         END DO
       END DO

*               solve
       CALL MR (X, Y, N, K, DWORK, IWORK)

*               print result
  10   FORMAT ('beta: ', $)
  20   FORMAT (F12.4, $)
  30   FORMAT ()
       PRINT 10
       DO J = 1, K
         PRINT 20, Y(J)
       END DO
       PRINT 30
       STOP 'program complete'
      END
```

Fortran



```c
#include <stdio.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_multifit.h>

double w[] = {  52.21, 53.12, 54.48, 55.84, 57.20,
                58.57, 59.93, 61.29, 63.11, 64.47,
                66.28, 68.10, 69.92, 72.19, 74.46 };
double h[] = {  1.47, 1.50, 1.52, 1.55, 1.57,
                1.60, 1.63, 1.65, 1.68, 1.70,
                1.73, 1.75, 1.78, 1.80, 1.83    };

int main()
{
        int n = sizeof(h)/sizeof(double);
        gsl_matrix *X = gsl_matrix_calloc(n, 3);
        gsl_vector *Y = gsl_vector_alloc(n);
        gsl_vector *beta = gsl_vector_alloc(3);

        for (int i = 0; i < n; i++) {
                gsl_vector_set(Y, i, w[i]);

                gsl_matrix_set(X, i, 0, 1);
                gsl_matrix_set(X, i, 1, h[i]);
                gsl_matrix_set(X, i, 2, h[i] * h[i]);
        }

        double chisq;
        gsl_matrix *cov = gsl_matrix_alloc(3, 3);
        gsl_multifit_linear_workspace * wspc = gsl_multifit_linear_alloc(n, 3);
        gsl_multifit_linear(X, Y, beta, cov, &chisq, wspc);

        printf("Beta:");
        for (int i = 0; i < 3; i++)
                printf("  %g", gsl_vector_get(beta, i));
        printf("\n");

        gsl_matrix_free(X);
        gsl_matrix_free(cov);
        gsl_vector_free(Y);
        gsl_vector_free(beta);
        gsl_multifit_linear_free(wspc);

}
```

C

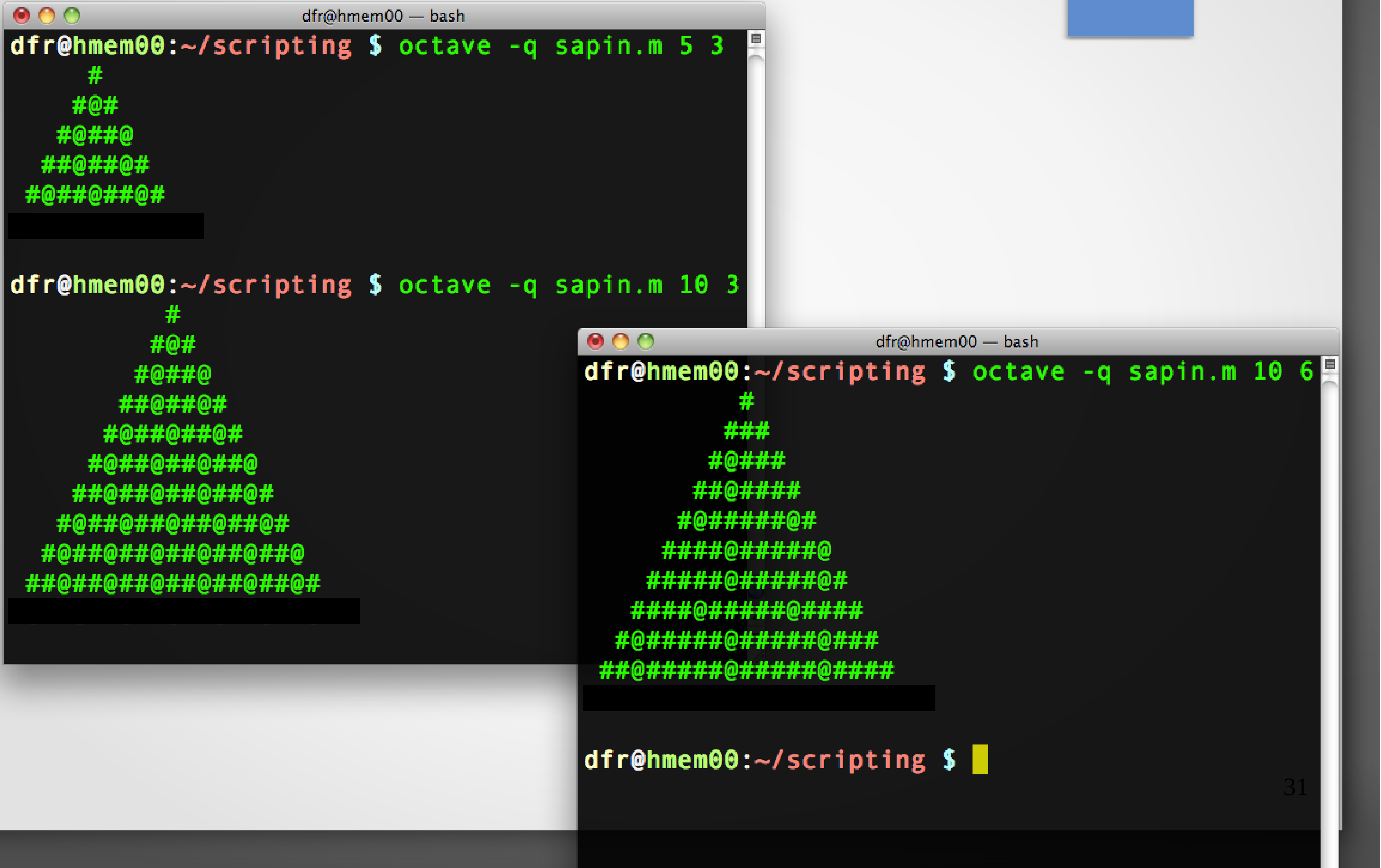
http://mathesaurus.sourceforge.net/matlab-python-xref.pdf

So..

Fast to learn
Fast to code

http://mathesaurus.sourceforge.net/matlab-python-xref.pdf

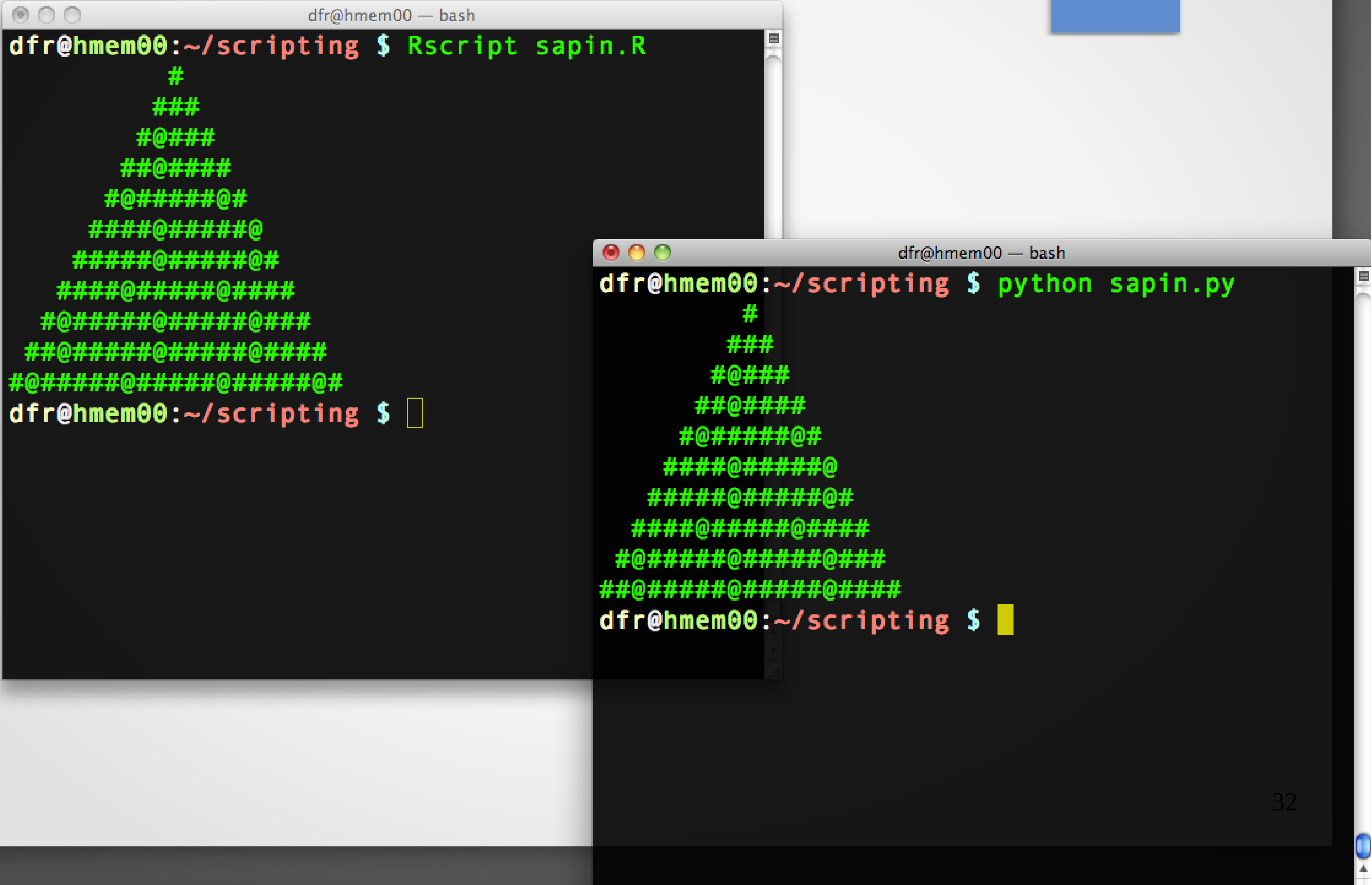
Challenge.. Write 'sapin.[m|R|py]'



```
dfr@hmem00:~/scripting $ octave -q sapin.m 5 3
     #
    #@#
   #@##@
  ##@##@#
 #@##@##@#
[redacted]

dfr@hmem00:~/scripting $ octave -q sapin.m 10 3
          #
         #@#
        #@##@
       ##@##@#
      #@##@##@#
     #@##@##@##@
    ##@##@##@##@#
   #@##@##@##@##@#
  #@##@##@##@##@##@
 ##@##@##@##@##@##@#
[redacted]
```

```
dfr@hmem00:~/scripting $ octave -q sapin.m 10 6
          #
         ###
        #@###
       ##@####
      #@#####@#
     ####@#####@
    #####@#####@#
   ####@#####@####
  #@#####@#####@###
 ##@#####@#####@####
[redacted]

dfr@hmem00:~/scripting $ 
```

Challenge.. Write 'sapin.[m|R|py]'



```
dfr@hmem00:~/scripting $ Rscript sapin.R
          #
         ###
        #@###
       ##@####
      #@#####@#
     ####@#####@
    #####@#####@#
   ####@#####@####
  #@#####@#####@###
 ##@#####@#####@####
#@#####@#####@#####@#
dfr@hmem00:~/scripting $ 
```

```
dfr@hmem00:~/scripting $ python sapin.py
         #
        ###
       #@###
      ##@####
     #@#####@#
    ####@#####@
   #####@#####@#
  ####@#####@####
 #@#####@#####@###
##@#####@#####@####
dfr@hmem00:~/scripting $ 
```

Help

You will need for-loops, if-conditionals, variable assignment, and printing which you can find in the slides

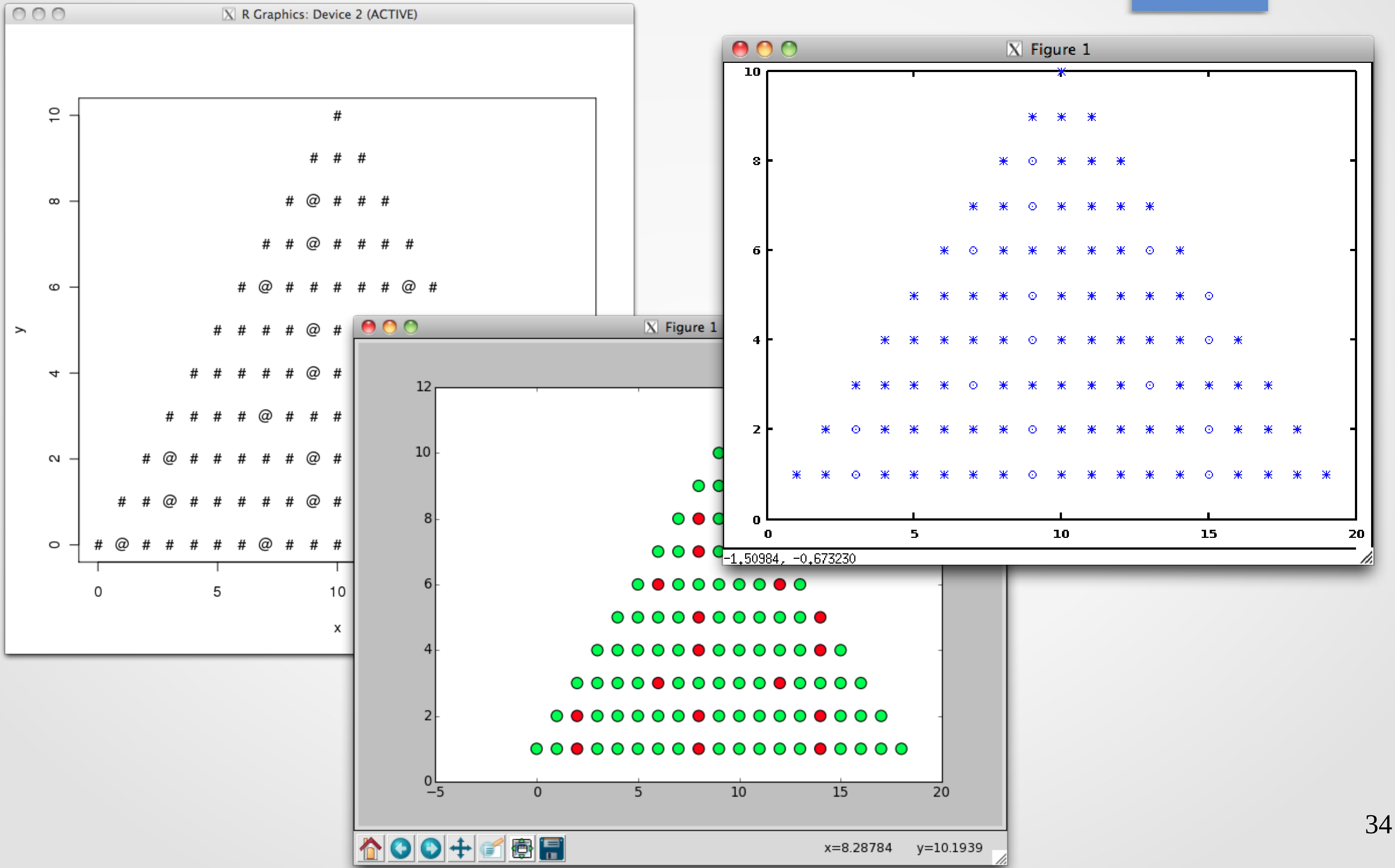
Other resources:
https://en.wikibooks.org/wiki/Octave_Programming_Tutorial/Getting_started
https://cran.r-project.org/doc/manuals/R-intro.html
http://wiki.scipy.org/Tentative_NumPy_Tutorial

http://stackoverflow.com/questions/14395569/how-to-output-text-in-the-r-console-without-creating-new-lines

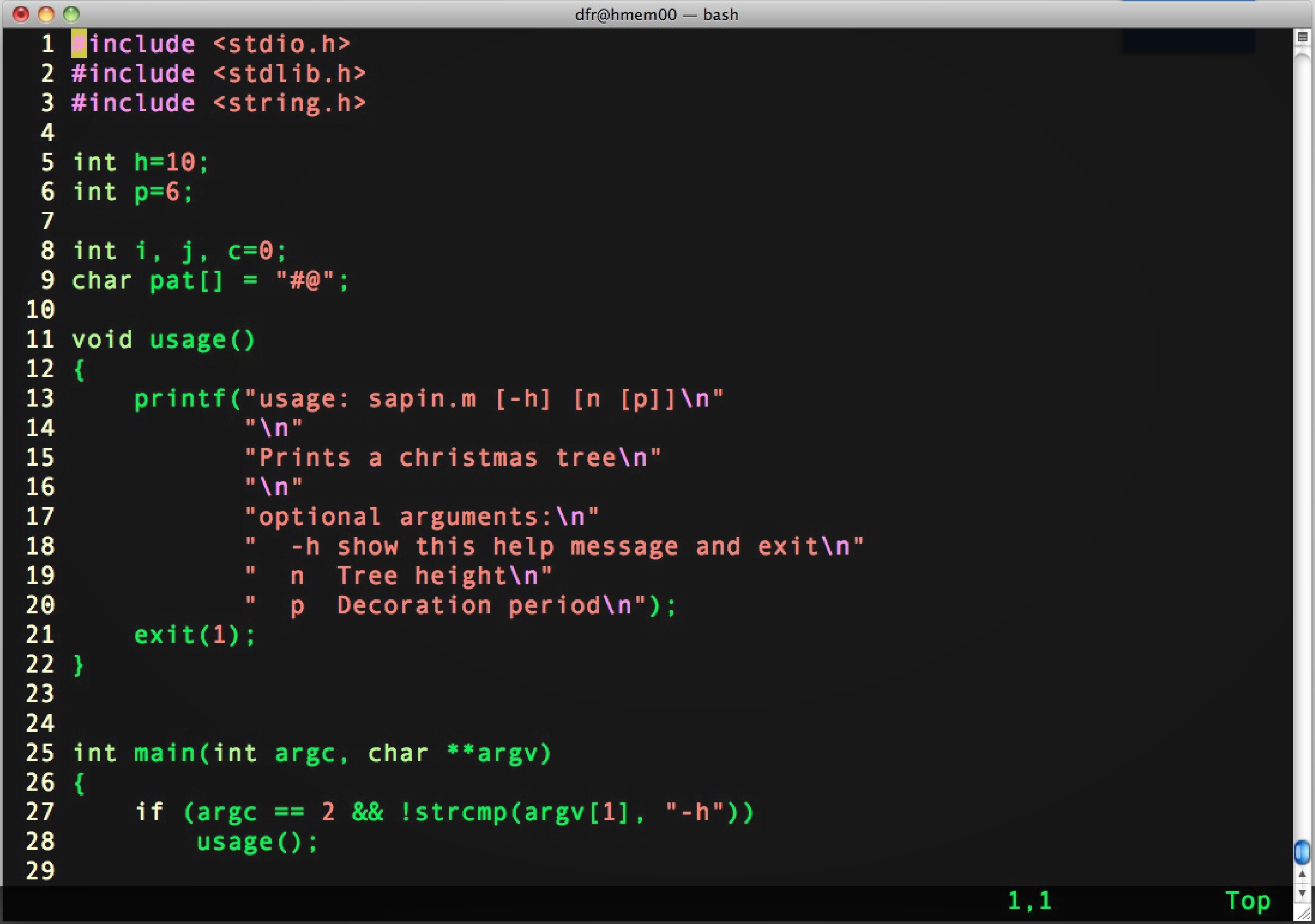
http://stackoverflow.com/questions/493386/how-to-print-in-python-without-newline-or-space

http://stackoverflow.com/questions/1012597/displaying-information-from-matlab-without-a-line-feed

If you are that quick... Try this:

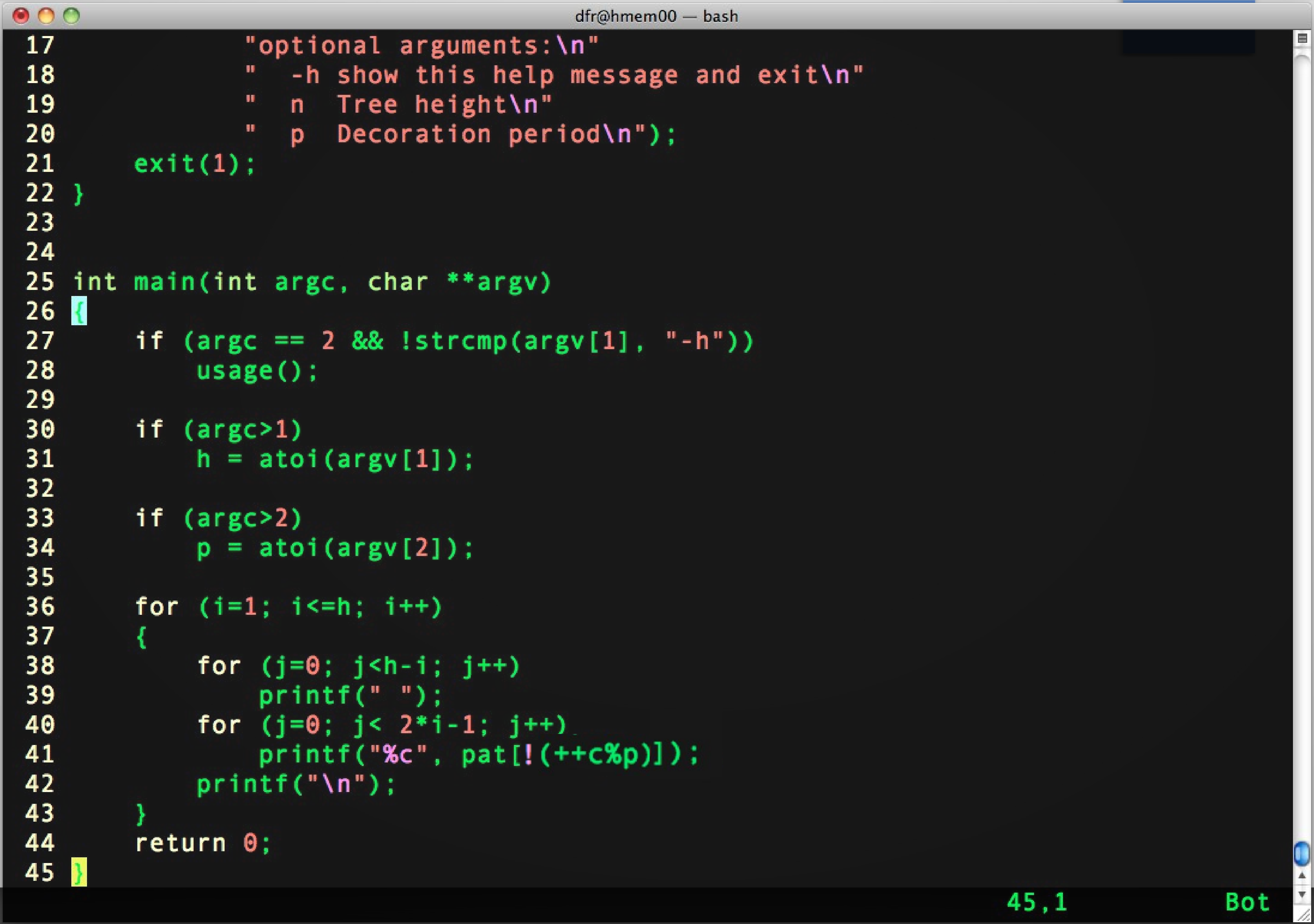


Possible solution (C)



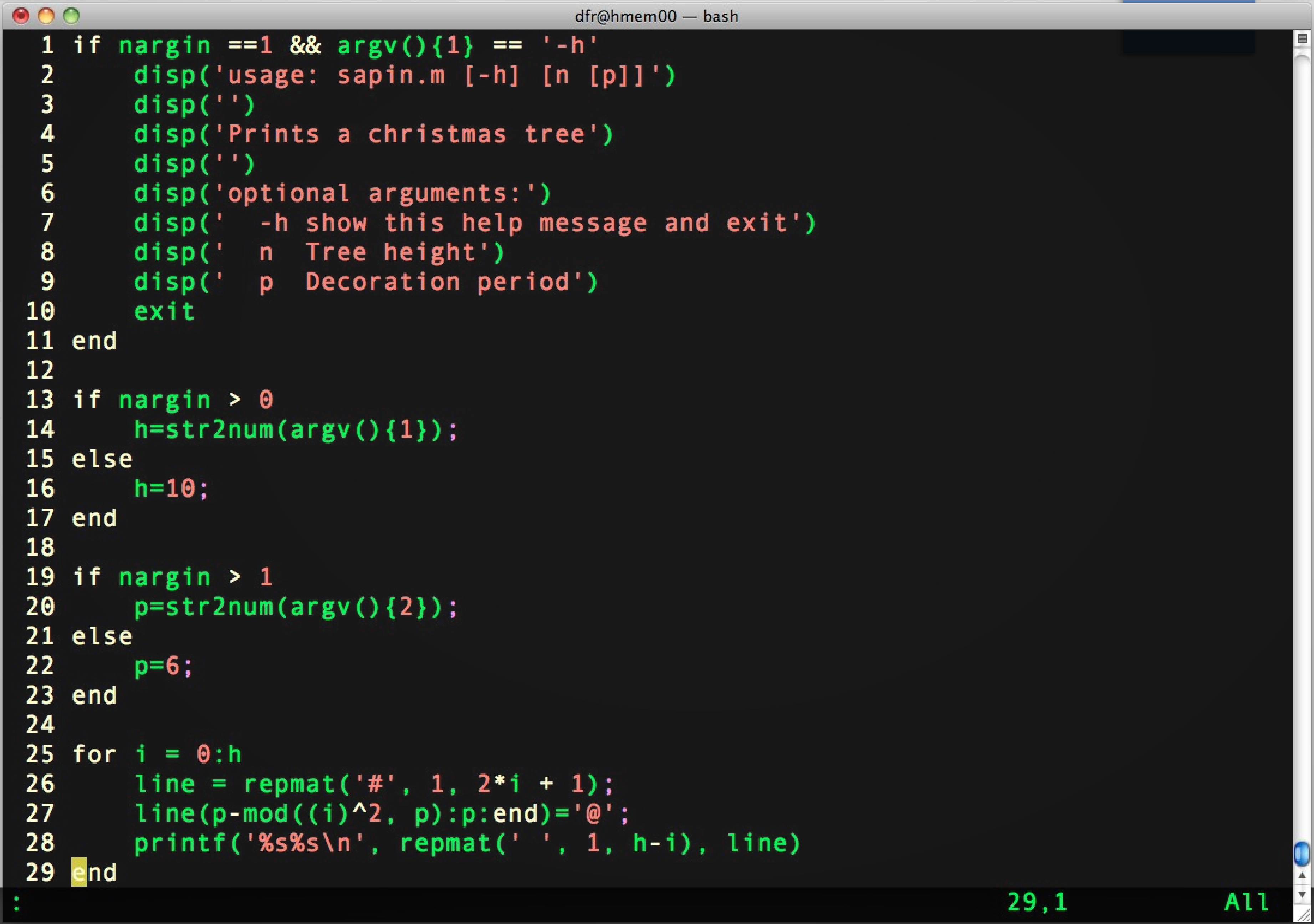
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int h=10;
int p=6;

int i, j, c=0;
char pat[] = "#@";

void usage()
{
    printf("usage: sapin.m [-h] [n [p]]\n"
           "\n"
           "Prints a christmas tree\n"
           "\n"
           "optional arguments:\n"
           "  -h show this help message and exit\n"
           "  n  Tree height\n"
           "  p  Decoration period\n");
    exit(1);
}


int main(int argc, char **argv)
{
    if (argc == 2 && !strcmp(argv[1], "-h"))
        usage();

```

Possible solution (C, cont'd)



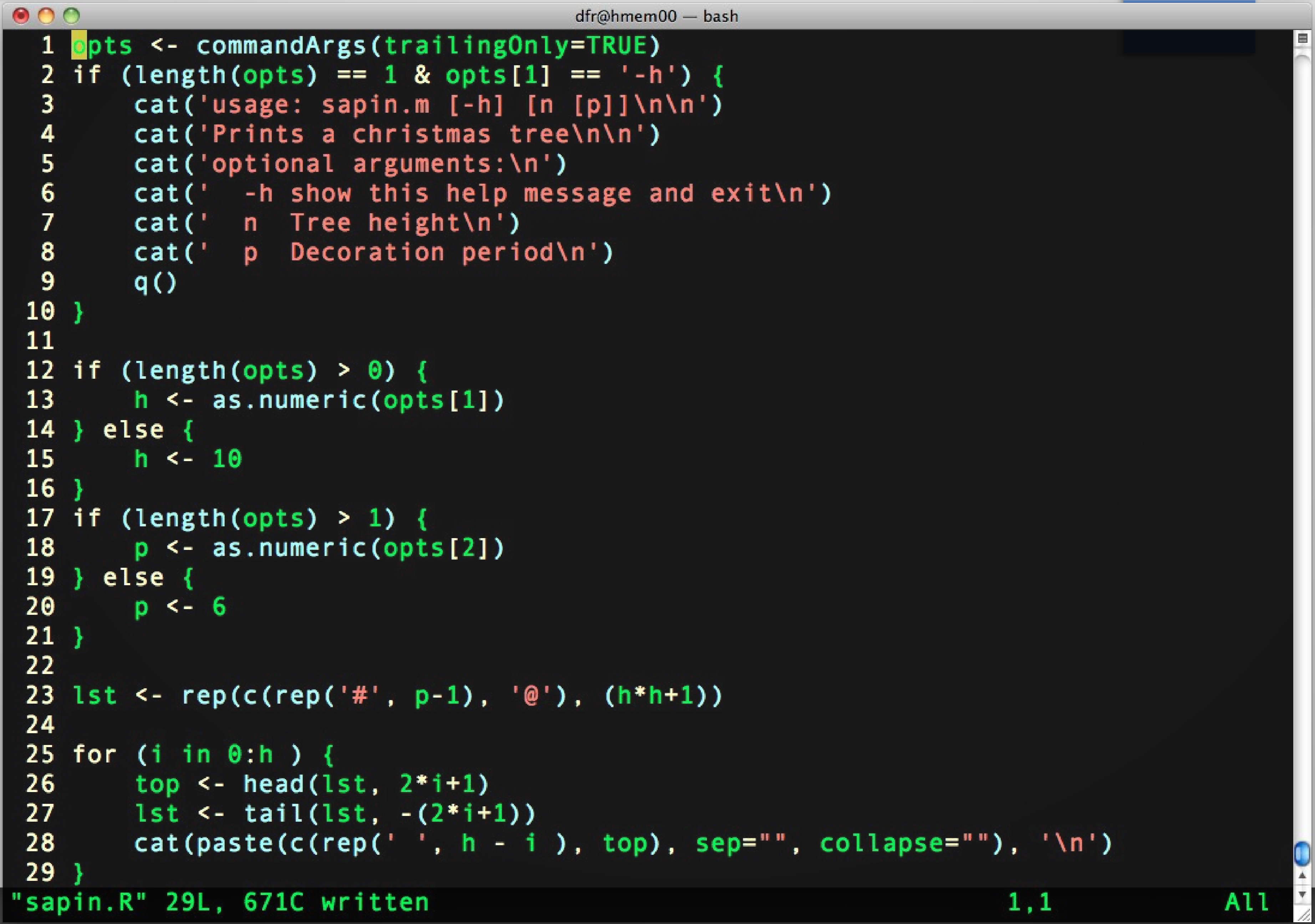
```
           "optional arguments:\n"
           "  -h show this help message and exit\n"
           "  n  Tree height\n"
           "  p  Decoration period\n");
    exit(1);
}


int main(int argc, char **argv)
{
    if (argc == 2 && !strcmp(argv[1], "-h"))
        usage();

    if (argc>1)
        h = atoi(argv[1]);

    if (argc>2)
        p = atoi(argv[2]);

    for (i=1; i<=h; i++)
    {
        for (j=0; j<h-i; j++)
            printf(" ");
        for (j=0; j< 2*i-1; j++)
            printf("%c", pat[!(++c%p)]);
        printf("\n");
    }
    return 0;
}
```

Possible solution (Octave)



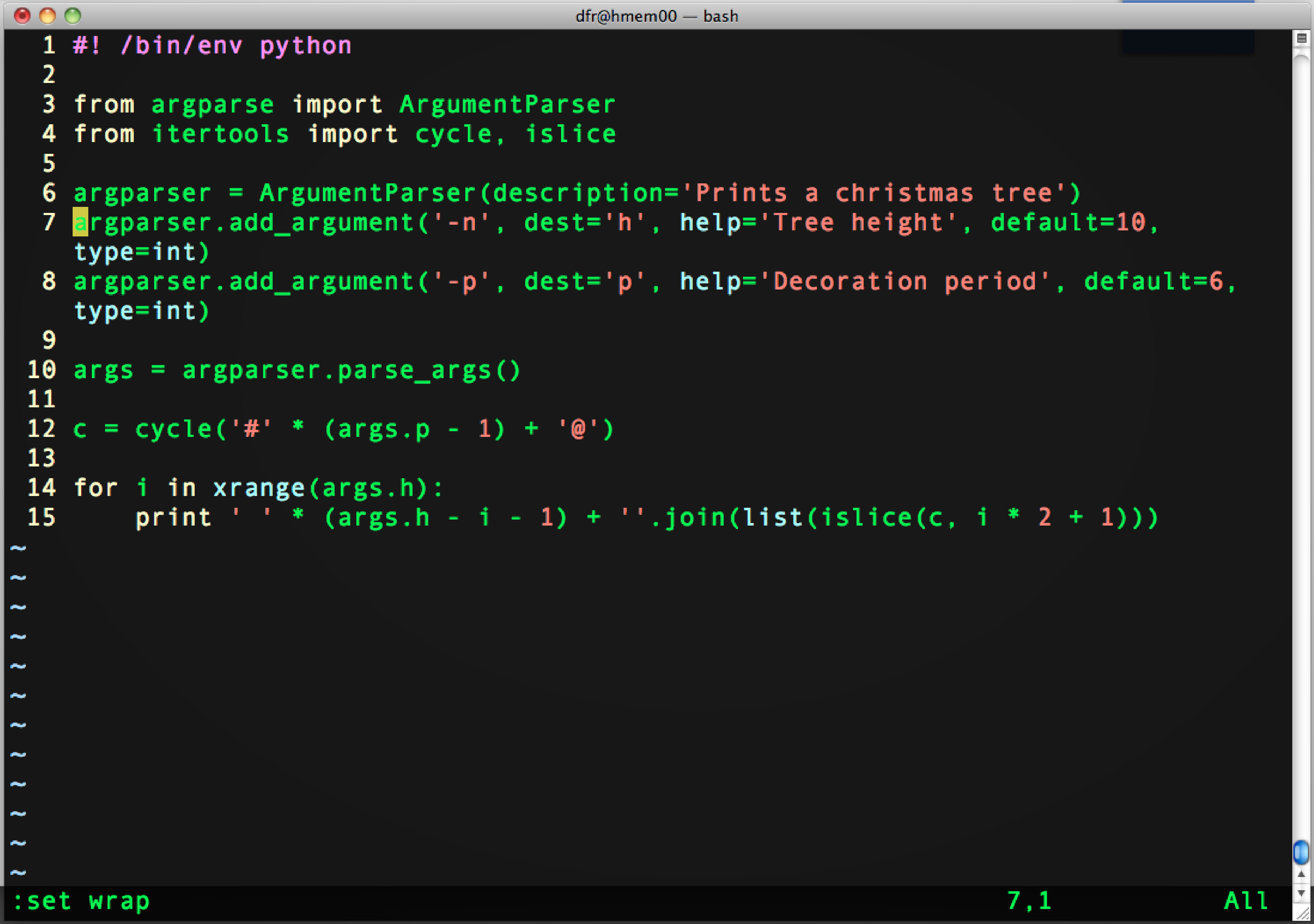
```
if nargin ==1 && argv(){1} == '-h'
    disp('usage: sapin.m [-h] [n [p]]')
    disp('')
    disp('Prints a christmas tree')
    disp('')
    disp('optional arguments:')
    disp('  -h show this help message and exit')
    disp('  n  Tree height')
    disp('  p  Decoration period')
    exit
end

if nargin > 0
    h=str2num(argv(){1});
else
    h=10;
end

if nargin > 1
    p=str2num(argv(){2});
else
    p=6;
end

for i = 0:h
    line = repmat('#', 1, 2*i + 1);
    line(p-mod((i)^2, p):p:end)='@';
    printf('%s%s\n', repmat(' ', 1, h-i), line)
end
:                                                              29,1          All
```

Possible solution (R)



```
opts <- commandArgs(trailingOnly=TRUE)
if (length(opts) == 1 & opts[1] == '-h') {
    cat('usage: sapin.m [-h] [n [p]]\n\n')
    cat('Prints a christmas tree\n\n')
    cat('optional arguments:\n')
    cat('  -h show this help message and exit\n')
    cat('  n  Tree height\n')
    cat('  p  Decoration period\n')
    q()
}

if (length(opts) > 0) {
    h <- as.numeric(opts[1])
} else {
    h <- 10
}
if (length(opts) > 1) {
    p <- as.numeric(opts[2])
} else {
    p <- 6
}

lst <- rep(c(rep('#', p-1), '@'), (h*h+1))

for (i in 0:h ) {
    top <- head(lst, 2*i+1)
    lst <- tail(lst, -(2*i+1))
    cat(paste(c(rep(' ', h - i ), top), sep="", collapse=""), '\n')
}
"sapin.R" 29L, 671C written                              1,1           All
```

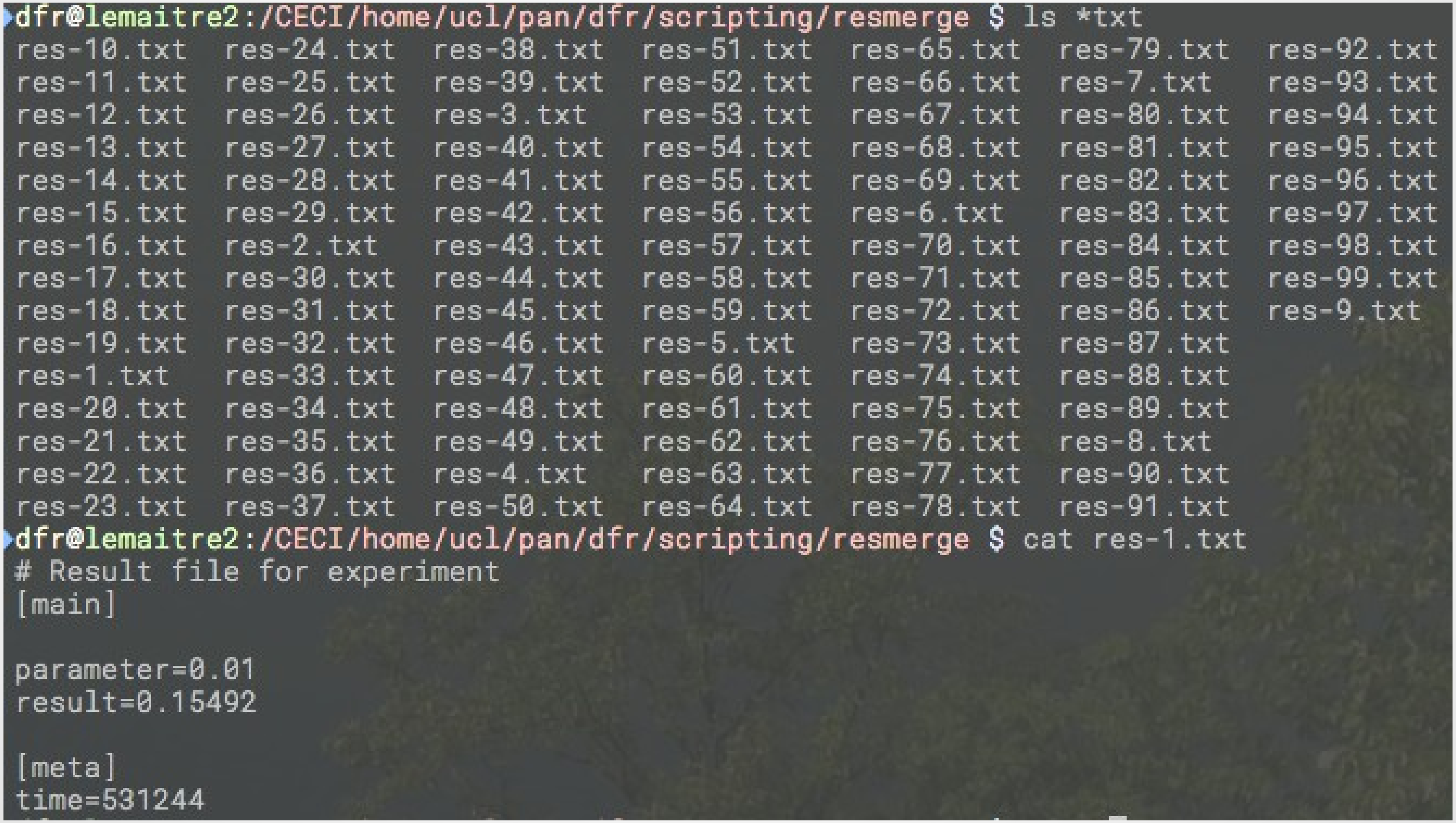
Possible solution (Python)



```
#! /bin/env python

from argparse import ArgumentParser
from itertools import cycle, islice

argparser = ArgumentParser(description='Prints a christmas tree')
argparser.add_argument('-n', dest='h', help='Tree height', default=10, type=int)
argparser.add_argument('-p', dest='p', help='Decoration period', default=6, type=int)

args = argparser.parse_args()

c = cycle('#' * (args.p - 1) + '@')

for i in xrange(args.h):
    print ' ' * (args.h - i - 1) + ''.join(list(islice(c, i * 2 + 1)))
```

```
:set wrap                                                      7,1           All
```

Second challenge

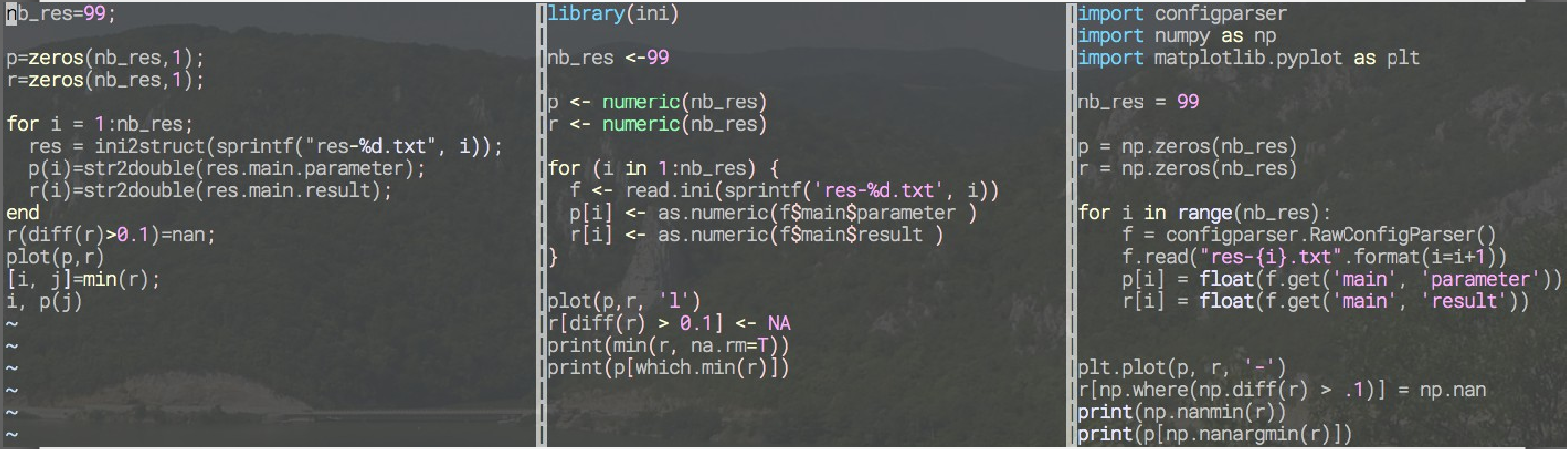


```
dfr@lemaitre2:/CECI/home/ucl/pan/dfr/scripting/resmerge $ ls *txt
res-10.txt  res-24.txt  res-38.txt  res-51.txt  res-65.txt  res-79.txt  res-92.txt
res-11.txt  res-25.txt  res-39.txt  res-52.txt  res-66.txt  res-7.txt   res-93.txt
res-12.txt  res-26.txt  res-3.txt   res-53.txt  res-67.txt  res-80.txt  res-94.txt
res-13.txt  res-27.txt  res-40.txt  res-54.txt  res-68.txt  res-81.txt  res-95.txt
res-14.txt  res-28.txt  res-41.txt  res-55.txt  res-69.txt  res-82.txt  res-96.txt
res-15.txt  res-29.txt  res-42.txt  res-56.txt  res-6.txt   res-83.txt  res-97.txt
res-16.txt  res-2.txt   res-43.txt  res-57.txt  res-70.txt  res-84.txt  res-98.txt
res-17.txt  res-30.txt  res-44.txt  res-58.txt  res-71.txt  res-85.txt  res-99.txt
res-18.txt  res-31.txt  res-45.txt  res-59.txt  res-72.txt  res-86.txt  res-9.txt
res-19.txt  res-32.txt  res-46.txt  res-5.txt   res-73.txt  res-87.txt
res-1.txt   res-33.txt  res-47.txt  res-60.txt  res-74.txt  res-88.txt
res-20.txt  res-34.txt  res-48.txt  res-61.txt  res-75.txt  res-89.txt
res-21.txt  res-35.txt  res-49.txt  res-62.txt  res-76.txt  res-8.txt
res-22.txt  res-36.txt  res-4.txt   res-63.txt  res-77.txt  res-90.txt
res-23.txt  res-37.txt  res-50.txt  res-64.txt  res-78.txt  res-91.txt
dfr@lemaitre2:/CECI/home/ucl/pan/dfr/scripting/resmerge $ cat res-1.txt
# Result file for experiment
[main]

parameter=0.01
result=0.15492

[meta]
time=531244
```

Second challenge

- Find for which value of 'parameter' is 'result' the lowest.
- Course of action:
  - Read all files and parse them (you might need to install additional packages/libraries/modules)
  - Build two arrays one of parameter values and the other one for result values
  - Remove problematic values (plotting might help here)
  - Find minimum

Possible solution



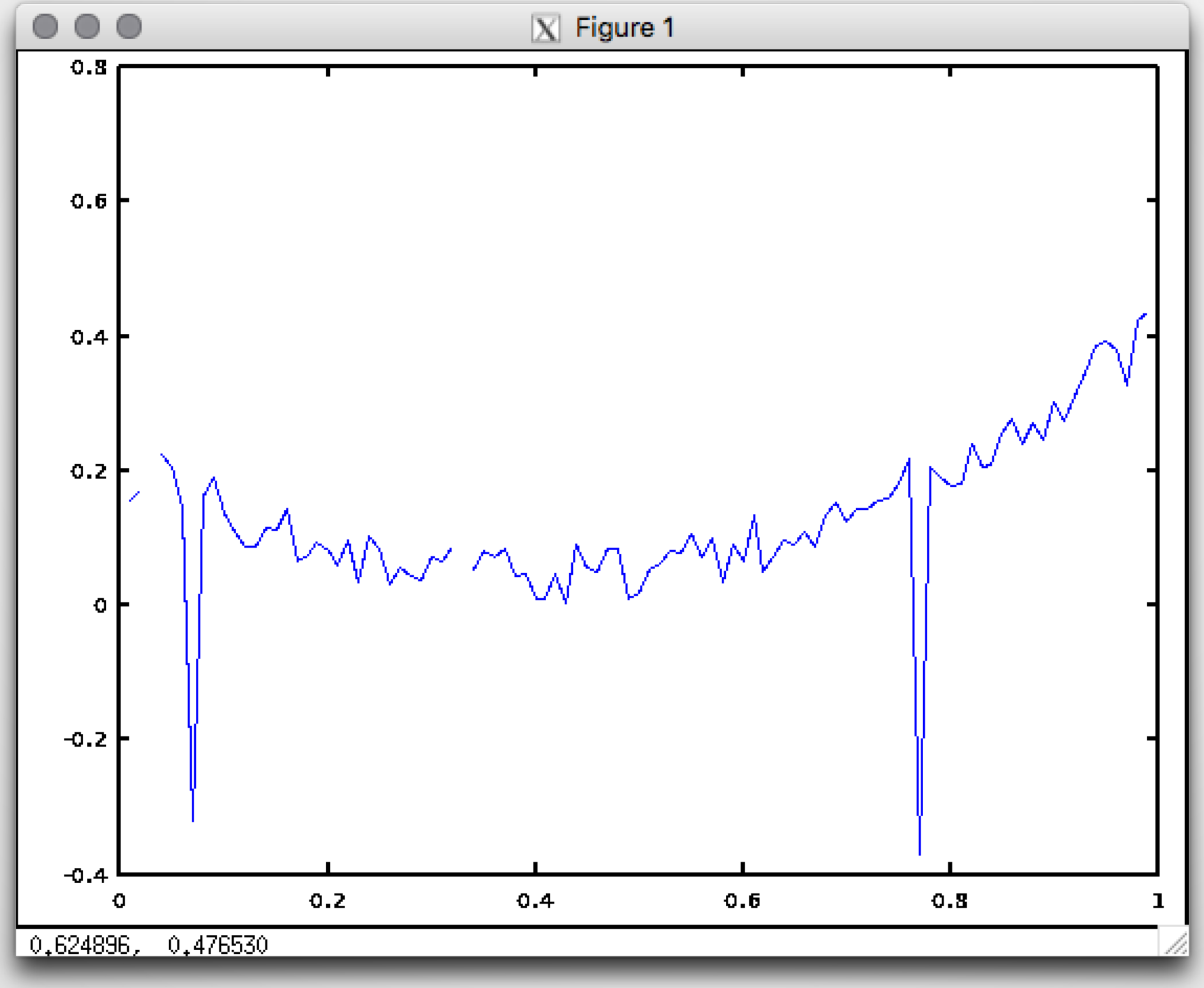
```
nb_res=99;

p=zeros(nb_res,1);
r=zeros(nb_res,1);

for i = 1:nb_res;
  res = ini2struct(sprintf("res-%d.txt", i));
  p(i)=str2double(res.main.parameter);
  r(i)=str2double(res.main.result);
end
r(diff(r)>0.1)=nan;
plot(p,r)
[i, j]=min(r);
i, p(j)
~
~
~
~
~
~
```

```
library(ini)

nb_res <-99

p <- numeric(nb_res)
r <- numeric(nb_res)

for (i in 1:nb_res) {
  f <- read.ini(sprintf('res-%d.txt', i))
  p[i] <- as.numeric(f$main$parameter )
  r[i] <- as.numeric(f$main$result )
}

plot(p,r, 'l')
r[diff(r) > 0.1] <- NA
print(min(r, na.rm=T))
print(p[which.min(r)])
```

```
import configparser
import numpy as np
import matplotlib.pyplot as plt

nb_res = 99

p = np.zeros(nb_res)
r = np.zeros(nb_res)

for i in range(nb_res):
    f = configparser.RawConfigParser()
    f.read("res-{i}.txt".format(i=i+1))
    p[i] = float(f.get('main', 'parameter'))
    r[i] = float(f.get('main', 'result'))


plt.plot(p, r, '-')
r[np.where(np.diff(r) > .1)] = np.nan
print(np.nanmin(r))
print(p[np.nanargmin(r)])
```

- https://nl.mathworks.com/matlabcentral/fileexchange/17177-ini2struct
- https://cran.r-project.org/web/packages/ini/index.html
- https://docs.python.org/3/library/configparser.html

Second challenge



0.624896, 0.476530

3.

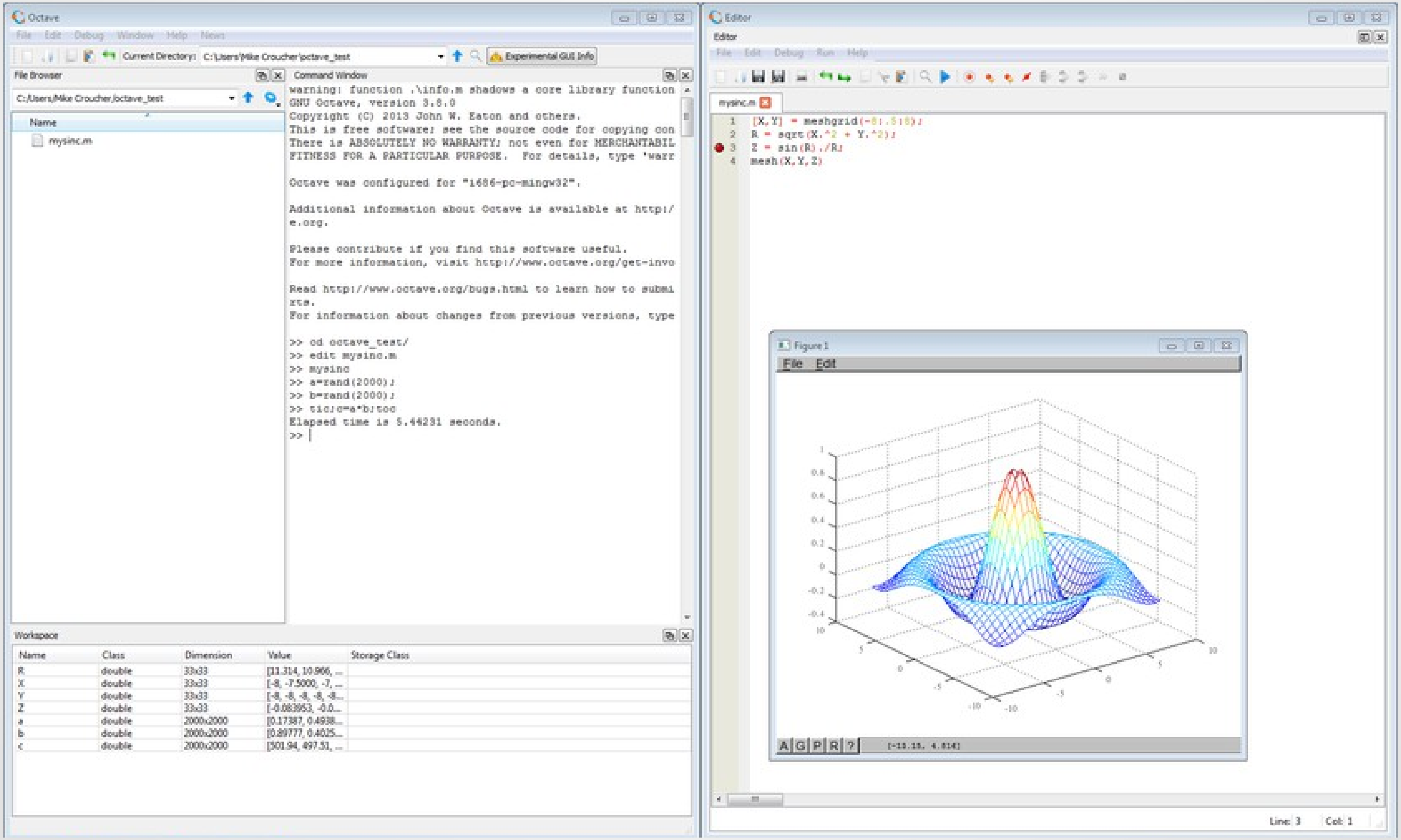
Graphical User Interfaces

Editing, debugging, accessing the doc, made easy

Literate programming

Authoring dynamic documents with code in them

Octave



Current Directory: C:\Users\Mike Croucher\octave_test

C:/Users/Mike Croucher/octave_test

```
warning: function .\info.m shadows a core library function
GNU Octave, version 3.8.0
Copyright (C) 2013 John W. Eaton and others.
This is free software; see the source code for copying con
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABIL
FITNESS FOR A PARTICULAR PURPOSE.  For details, type 'warr

Octave was configured for "i686-pc-mingw32".

Additional information about Octave is available at http:/
e.org.

Please contribute if you find this software useful.
For more information, visit http://www.octave.org/get-invo

Read http://www.octave.org/bugs.html to learn how to submi
rts.
For information about changes from previous versions, type

>> cd octave_test/
>> edit mysinc.m
>> mysinc
>> a=rand(2000);
>> b=rand(2000);
>> tic;c=a*b;toc
Elapsed time is 5.44231 seconds.
>> 
```

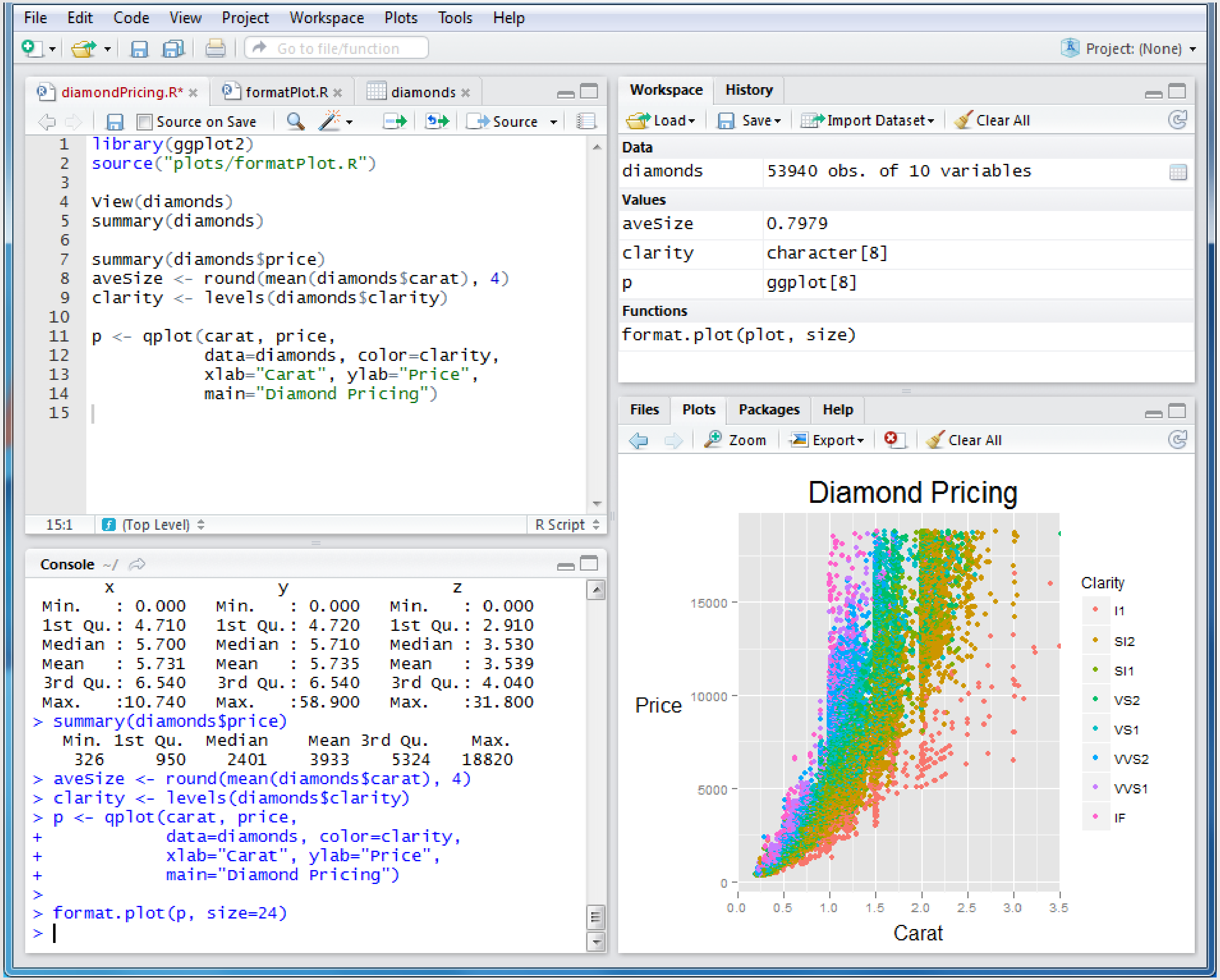
Name	Class	Dimension	Value	Storage Class
R	double	33x33	[11.314, 10.966, ...	
X	double	33x33	[-8, -7.5000, -7, ...	
Y	double	33x33	[-8, -8, -8, -8, -8...	
Z	double	33x33	[-0.083953, -0.0...	
a	double	2000x2000	[0.17387, 0.4938...	
b	double	2000x2000	[0.89777, 0.4025...	
c	double	2000x2000	[501.94, 497.51, ...	

```
[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2);
Z = sin(R)./R;
mesh(X,Y,Z)
```

[-13.15, 4.814]

Line: 3 Col: 1

Rstudio



```r
library(ggplot2)
source("plots/formatPlot.R")

View(diamonds)
summary(diamonds)

summary(diamonds$price)
aveSize <- round(mean(diamonds$carat), 4)
clarity <- levels(diamonds$clarity)

p <- qplot(carat, price,
           data=diamonds, color=clarity,
           xlab="Carat", ylab="Price",
           main="Diamond Pricing")

```

Data

diamonds	53940 obs. of 10 variables

Values

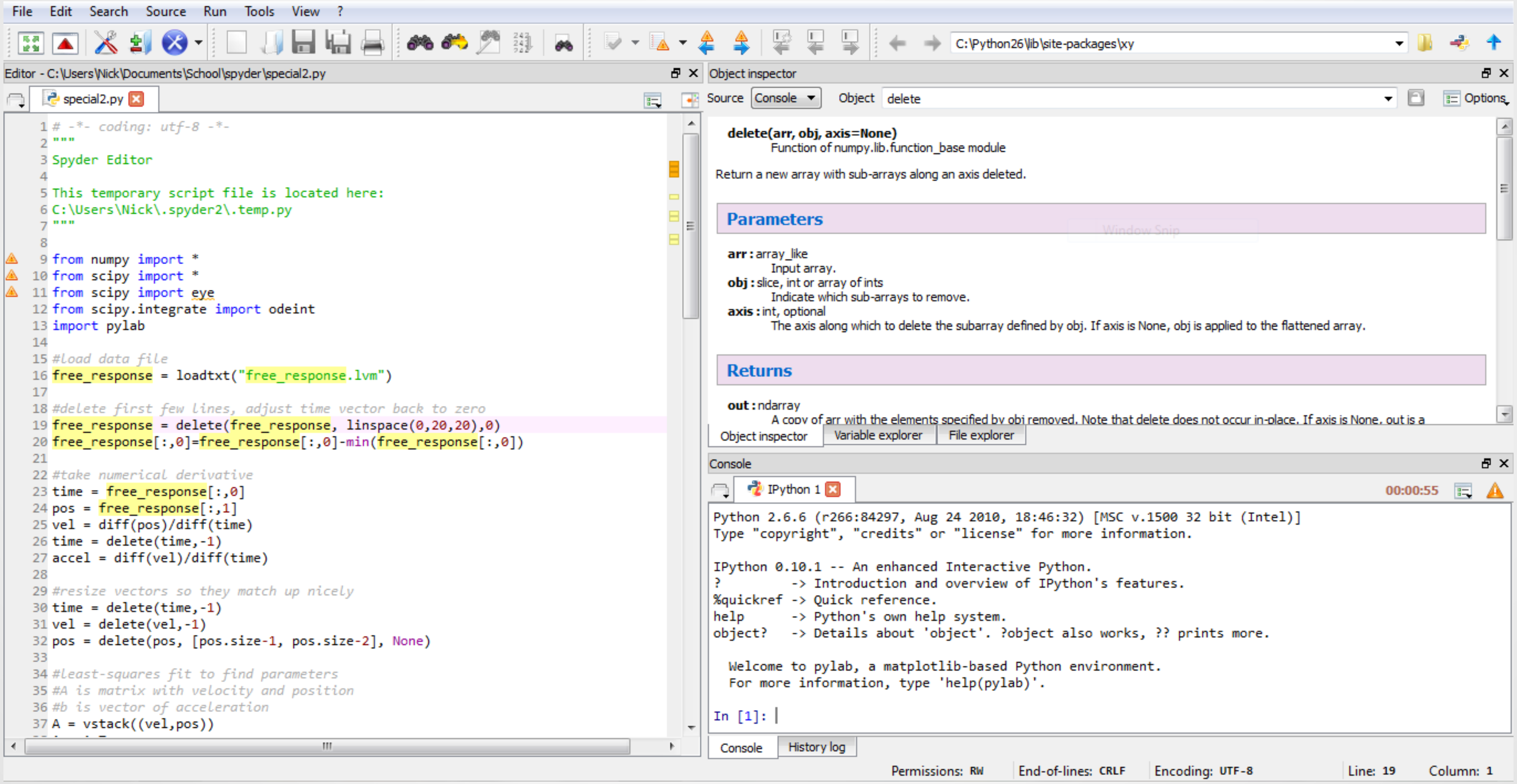
aveSize	0.7979
clarity	character[8]
p	ggplot[8]

Functions

format.plot(plot, size)

```
       x                y                z
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
 1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
 Median : 5.700   Median : 5.710   Median : 3.530
 Mean   : 5.731   Mean   : 5.735   Mean   : 3.539
 3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040
 Max.   :10.740   Max.   :58.900   Max.   :31.800
> summary(diamonds$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    326     950    2401    3933    5324   18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+            data=diamonds, color=clarity,
+            xlab="Carat", ylab="Price",
+            main="Diamond Pricing")
>
> format.plot(p, size=24)
> 
```

Spyder



```python
# -*- coding: utf-8 -*-
"""
Spyder Editor

This temporary script file is located here:
C:\Users\Nick\.spyder2\.temp.py
"""

from numpy import *
from scipy import *
from scipy import eye
from scipy.integrate import odeint
import pylab

#load data file
free_response = loadtxt("free_response.lvm")

#delete first few lines, adjust time vector back to zero
free_response = delete(free_response, linspace(0,20,20),0)
free_response[:,0]=free_response[:,0]-min(free_response[:,0])

#take numerical derivative
time = free_response[:,0]
pos = free_response[:,1]
vel = diff(pos)/diff(time)
time = delete(time,-1)
accel = diff(vel)/diff(time)

#resize vectors so they match up nicely
time = delete(time,-1)
vel = delete(vel,-1)
pos = delete(pos, [pos.size-1, pos.size-2], None)

#least-squares fit to find parameters
#A is matrix with velocity and position
#b is vector of acceleration
A = vstack((vel,pos))
```

delete(arr, obj, axis=None)
Function of numpy.lib.function_base module

Return a new array with sub-arrays along an axis deleted.

Parameters

arr : array_like
Input array.
obj : slice, int or array of ints
Indicate which sub-arrays to remove.
axis : int, optional
The axis along which to delete the subarray defined by obj. If axis is None, obj is applied to the flattened array.

Returns

out : ndarray
A copy of arr with the elements specified by obj removed. Note that delete does not occur in-place. If axis is None, out is a

```
Python 2.6.6 (r266:84297, Aug 24 2010, 18:46:32) [MSC v.1500 32 bit (Intel)]
Type "copyright", "credits" or "license" for more information.

IPython 0.10.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object'. ?object also works, ?? prints more.

  Welcome to pylab, a matplotlib-based Python environment.
  For more information, type 'help(pylab)'.

In [1]: 
```

Permissions: RW   End-of-lines: CRLF   Encoding: UTF-8   Line: 19   Column: 1

3.

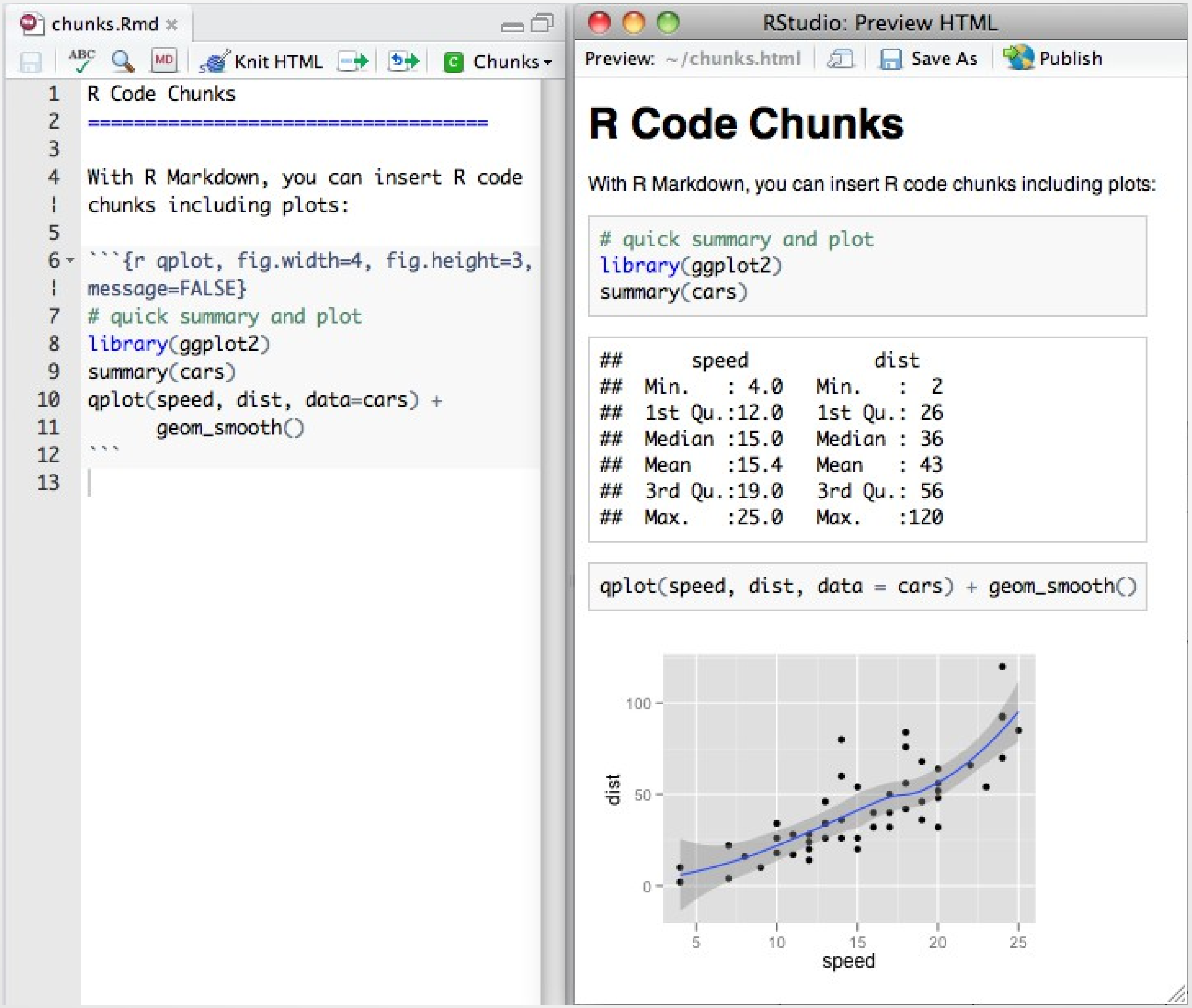
Graphical User Interfaces

Editing, debugging, accessing the doc, made easy

Literate programming

Authoring HTML or LaTeX documents
with code and results in them

RMarkdown and KnitR



~~~
R Code Chunks
===================================

With R Markdown, you can insert R code chunks including plots:

```{r qplot, fig.width=4, fig.height=3, message=FALSE}
# quick summary and plot
library(ggplot2)
summary(cars)
qplot(speed, dist, data=cars) +
      geom_smooth()
```

~~~

R Code Chunks

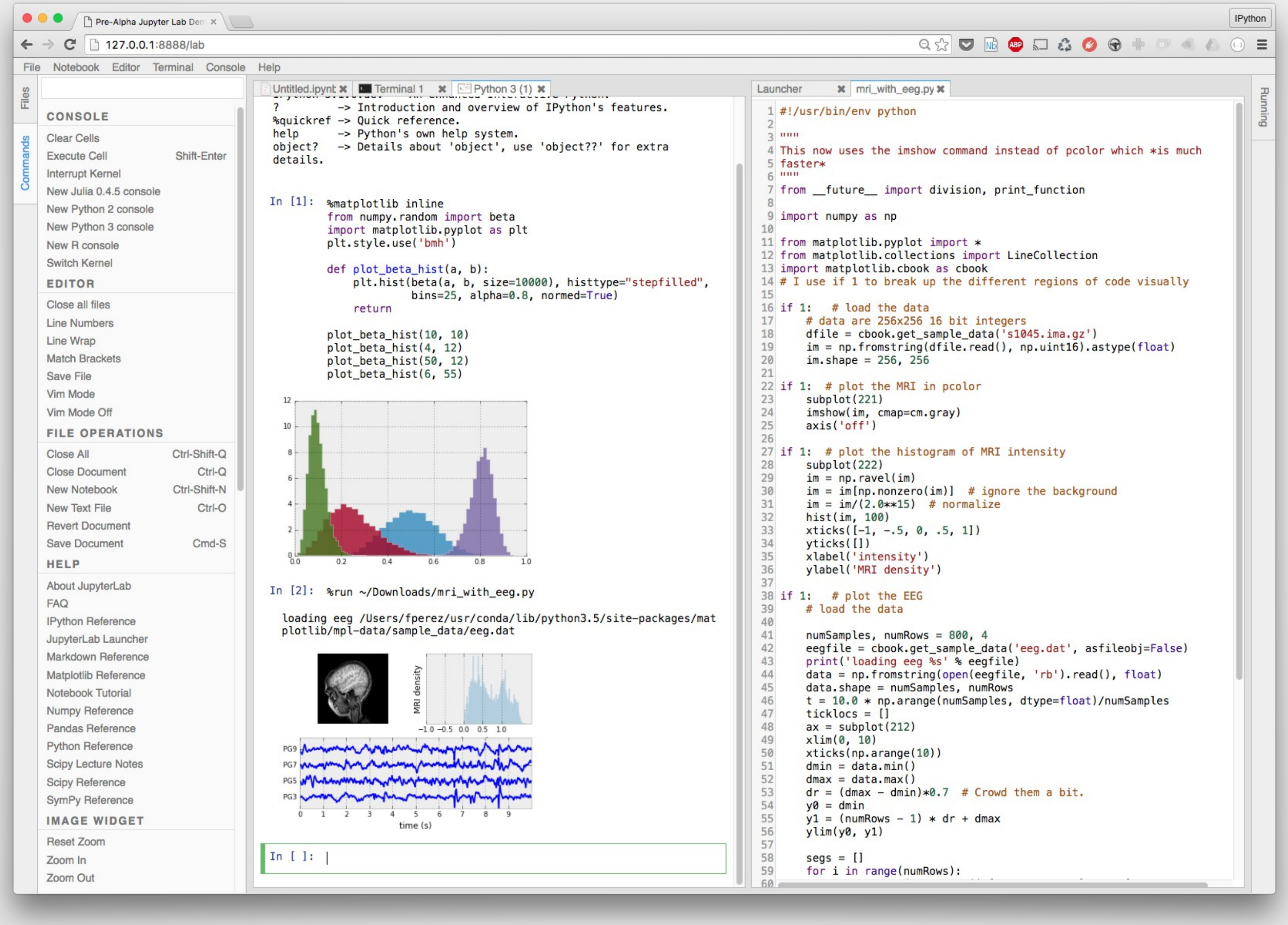
With R Markdown, you can insert R code chunks including plots:

```
# quick summary and plot
library(ggplot2)
summary(cars)
```

```
##      speed           dist    
##  Min.   : 4.0   Min.   :  2  
##  1st Qu.:12.0   1st Qu.: 26  
##  Median :15.0   Median : 36  
##  Mean   :15.4   Mean   : 43  
##  3rd Qu.:19.0   3rd Qu.: 56  
##  Max.   :25.0   Max.   :120  
```

```
qplot(speed, dist, data = cars) + geom_smooth()
```

Jupyter notebooks



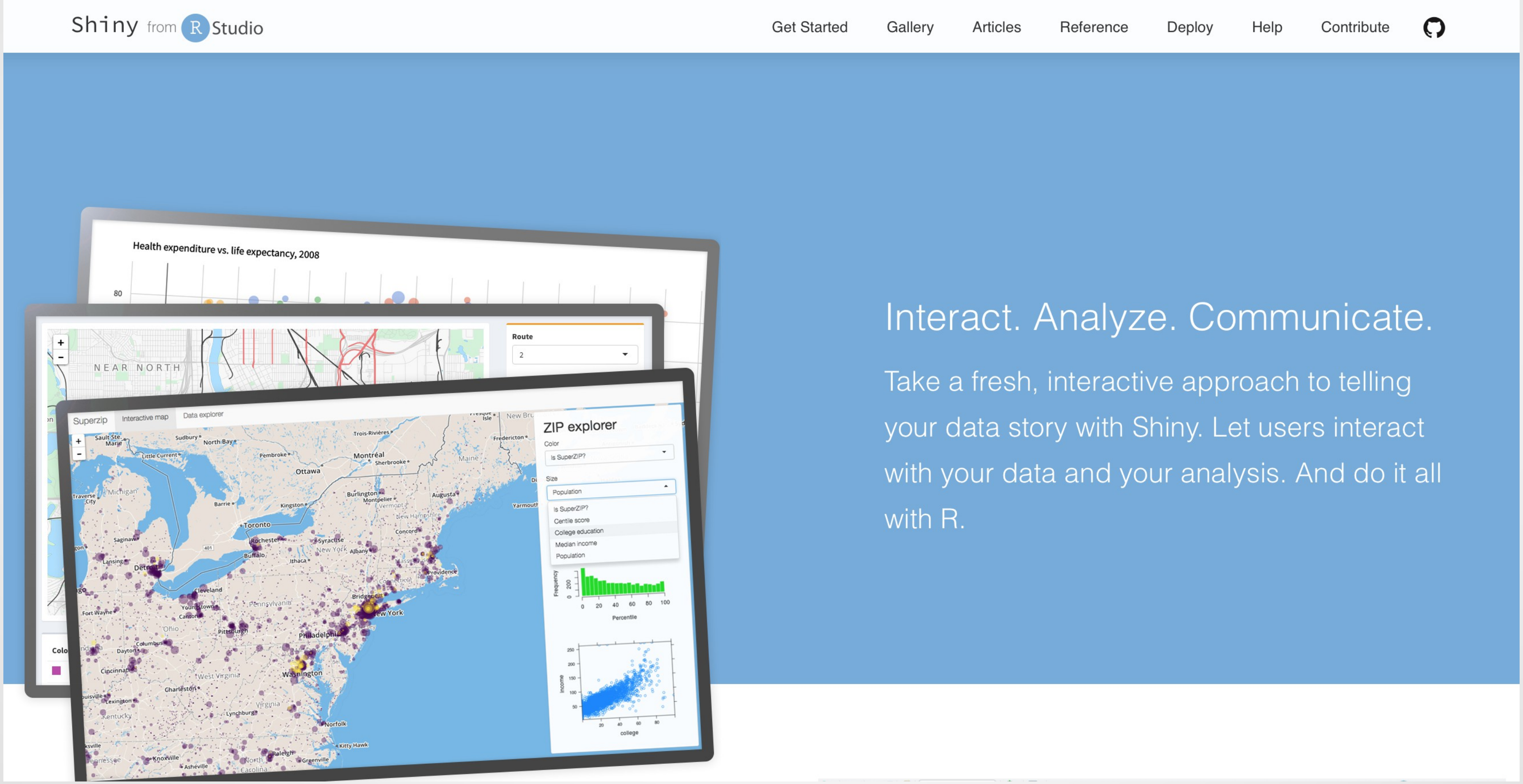
```
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra
details.


In [1]:  %matplotlib inline
         from numpy.random import beta
         import matplotlib.pyplot as plt
         plt.style.use('bmh')

         def plot_beta_hist(a, b):
             plt.hist(beta(a, b, size=10000), histtype="stepfilled",
                      bins=25, alpha=0.8, normed=True)
             return

         plot_beta_hist(10, 10)
         plot_beta_hist(4, 12)
         plot_beta_hist(50, 12)
         plot_beta_hist(6, 55)
```

```
In [2]:  %run ~/Downloads/mri_with_eeg.py

 loading eeg /Users/fperez/usr/conda/lib/python3.5/site-packages/mat
 plotlib/mpl-data/sample_data/eeg.dat
```

```
In [ ]:
```

```python
#!/usr/bin/env python

"""
This now uses the imshow command instead of pcolor which *is much
faster*
"""
from __future__ import division, print_function

import numpy as np

from matplotlib.pyplot import *
from matplotlib.collections import LineCollection
import matplotlib.cbook as cbook
# I use if 1 to break up the different regions of code visually

if 1:   # load the data
    # data are 256x256 16 bit integers
    dfile = cbook.get_sample_data('s1045.ima.gz')
    im = np.fromstring(dfile.read(), np.uint16).astype(float)
    im.shape = 256, 256

if 1:  # plot the MRI in pcolor
    subplot(221)
    imshow(im, cmap=cm.gray)
    axis('off')

if 1:  # plot the histogram of MRI intensity
    subplot(222)
    im = np.ravel(im)
    im = im[np.nonzero(im)]  # ignore the background
    im = im/(2.0**15)  # normalize
    hist(im, 100)
    xticks([-1, -.5, 0, .5, 1])
    yticks([])
    xlabel('intensity')
    ylabel('MRI density')

if 1:   # plot the EEG
    # load the data

    numSamples, numRows = 800, 4
    eegfile = cbook.get_sample_data('eeg.dat', asfileobj=False)
    print('loading eeg %s' % eegfile)
    data = np.fromstring(open(eegfile, 'rb').read(), float)
    data.shape = numSamples, numRows
    t = 10.0 * np.arange(numSamples, dtype=float)/numSamples
    ticklocs = []
    ax = subplot(212)
    xlim(0, 10)
    xticks(np.arange(10))
    dmin = data.min()
    dmax = data.max()
    dr = (dmax - dmin)*0.7  # Crowd them a bit.
    y0 = dmin
    y1 = (numRows - 1) * dr + dmax
    ylim(y0, y1)

    segs = []
    for i in range(numRows):
```

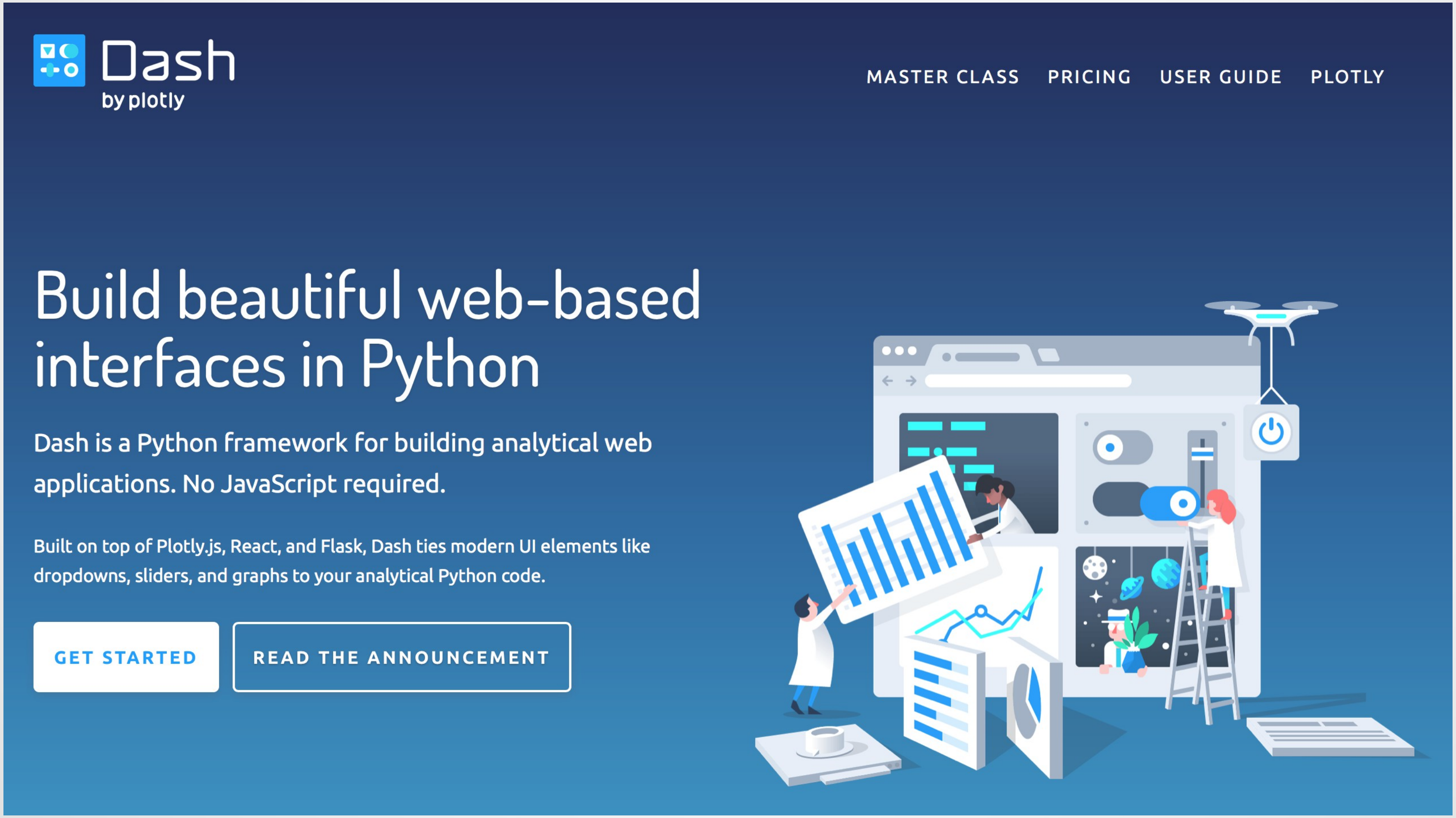
Shiny



Interact. Analyze. Communicate.

Take a fresh, interactive approach to telling your data story with Shiny. Let users interact with your data and your analysis. And do it all with R.

Dash



Build beautiful web-based interfaces in Python

Dash is a Python framework for building analytical web applications. No JavaScript required.

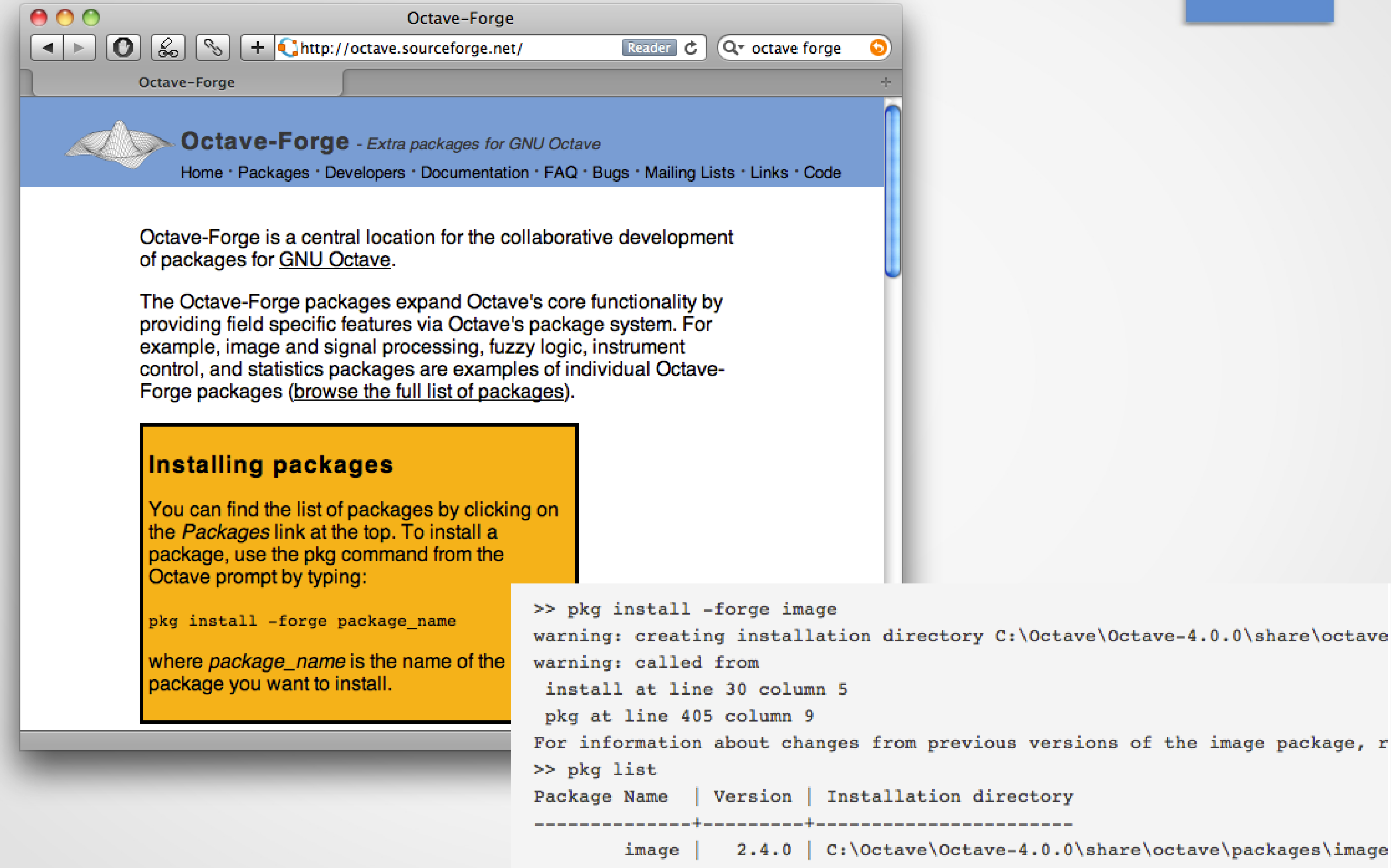
Built on top of Plotly.js, React, and Flask, Dash ties modern UI elements like dropdowns, sliders, and graphs to your analytical Python code.

4.

Extensions

Packages – Libraries – Modules

Octave Forge



Octave-Forge
http://octave.sourceforge.net/
Reader
octave forge
Octave-Forge
Octave-Forge - Extra packages for GNU Octave
Home · Packages · Developers · Documentation · FAQ · Bugs · Mailing Lists · Links · Code

Octave-Forge is a central location for the collaborative development of packages for GNU Octave.

The Octave-Forge packages expand Octave's core functionality by providing field specific features via Octave's package system. For example, image and signal processing, fuzzy logic, instrument control, and statistics packages are examples of individual Octave-Forge packages (browse the full list of packages).

Installing packages

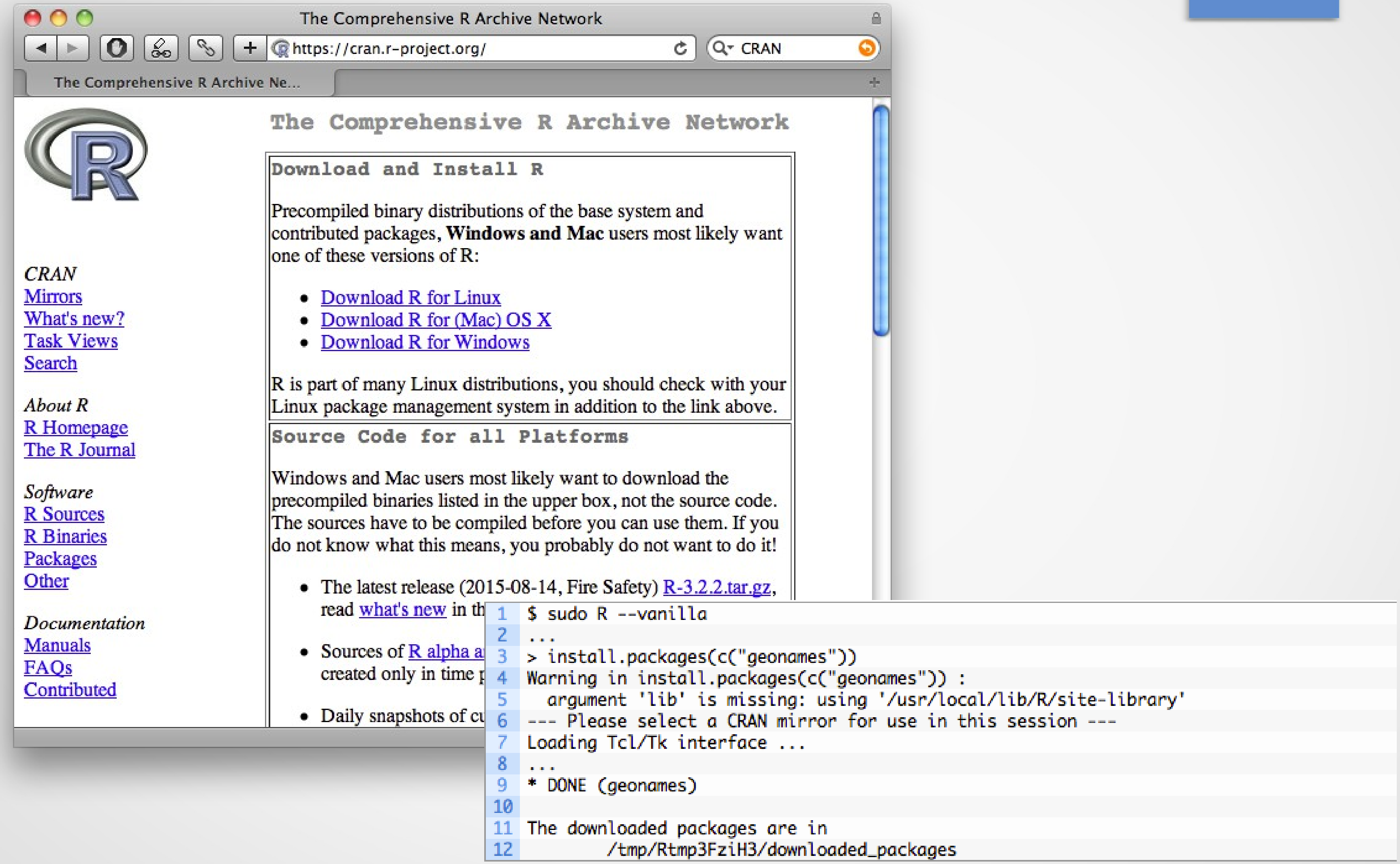
You can find the list of packages by clicking on the *Packages* link at the top. To install a package, use the pkg command from the Octave prompt by typing:

```
pkg install -forge package_name
```

where *package_name* is the name of the package you want to install.

```
>> pkg install -forge image
warning: creating installation directory C:\Octave\Octave-4.0.0\share\octave
warning: called from
 install at line 30 column 5
 pkg at line 405 column 9
For information about changes from previous versions of the image package, r
>> pkg list
Package Name  | Version | Installation directory
--------------+---------+-----------------------
        image |   2.4.0 | C:\Octave\Octave-4.0.0\share\octave\packages\image
```

CRAN



The Comprehensive R Archive Network
https://cran.r-project.org/
CRAN
The Comprehensive R Archive Ne...

The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

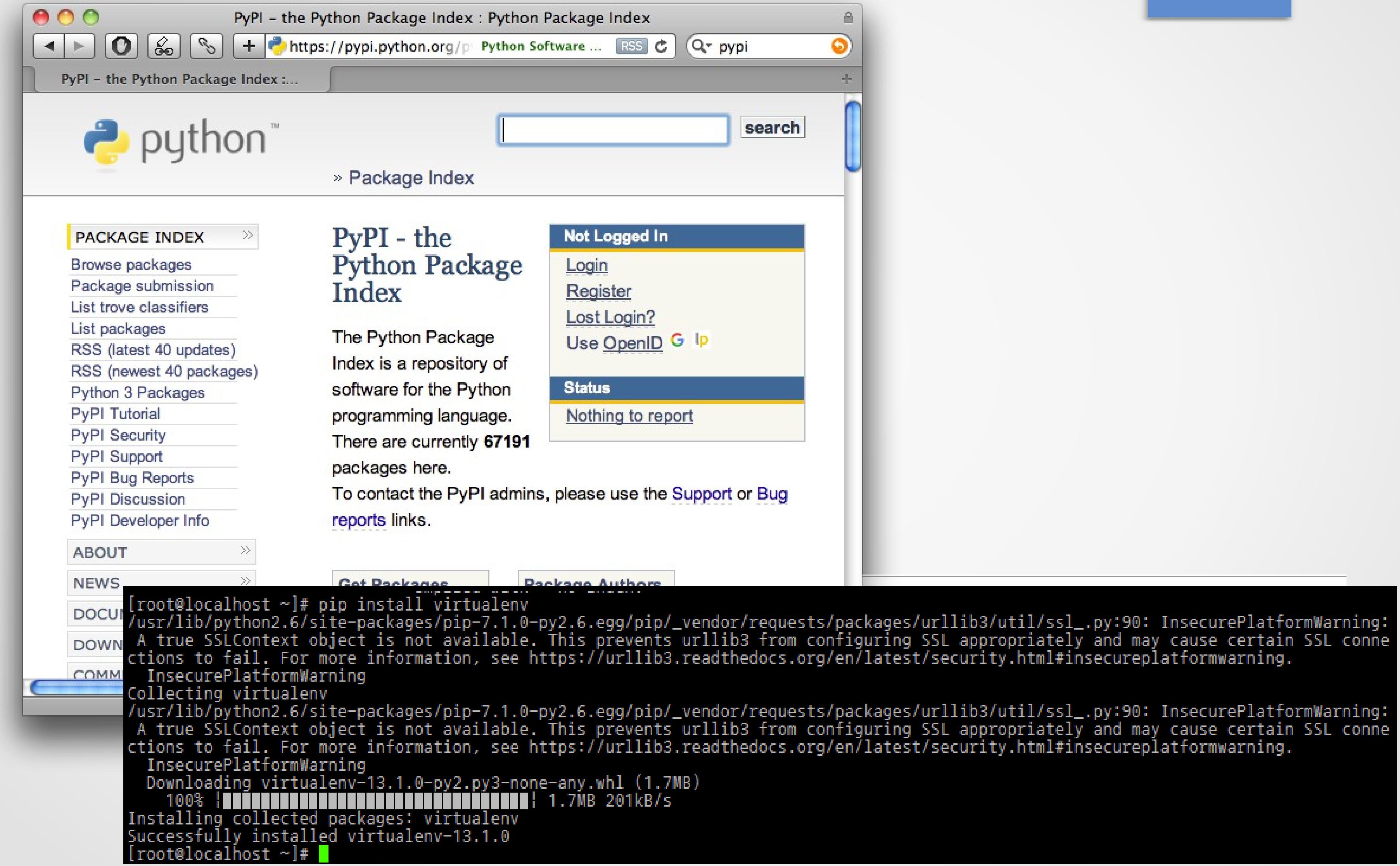
Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2015-08-14, Fire Safety) R-3.2.2.tar.gz, read what's new in th
- Sources of R alpha a created only in time p
- Daily snapshots of cu

```
$ sudo R --vanilla
...
> install.packages(c("geonames"))
Warning in install.packages(c("geonames")) :
  argument 'lib' is missing: using '/usr/local/lib/R/site-library'
--- Please select a CRAN mirror for use in this session ---
Loading Tcl/Tk interface ...
...
* DONE (geonames)

The downloaded packages are in
        /tmp/Rtmp3FziH3/downloaded_packages
```

PyPI



PyPI – the Python Package Index : Python Package Index

https://pypi.python.org/p Python Software ...

PyPI – the Python Package Index :...

python™

search

» Package Index

PACKAGE INDEX

- Browse packages
- Package submission
- List trove classifiers
- List packages
- RSS (latest 40 updates)
- RSS (newest 40 packages)
- Python 3 Packages
- PyPI Tutorial
- PyPI Security
- PyPI Support
- PyPI Bug Reports
- PyPI Discussion
- PyPI Developer Info

ABOUT

NEWS

DOCU

DOWN

COMM

PyPI - the Python Package Index

The Python Package Index is a repository of software for the Python programming language. There are currently **67191** packages here.

To contact the PyPI admins, please use the Support or Bug reports links.

Get Packages

Package Authors

Not Logged In

- Login
- Register
- Lost Login?
- Use OpenID

Status

Nothing to report

```
[root@localhost ~]# pip install virtualenv
/usr/lib/python2.6/site-packages/pip-7.1.0-py2.6.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:90: InsecurePlatformWarning:
 A true SSLContext object is not available. This prevents urllib3 from configuring SSL appropriately and may cause certain SSL conne
ctions to fail. For more information, see https://urllib3.readthedocs.org/en/latest/security.html#insecureplatformwarning.
  InsecurePlatformWarning
Collecting virtualenv
/usr/lib/python2.6/site-packages/pip-7.1.0-py2.6.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:90: InsecurePlatformWarning:
 A true SSLContext object is not available. This prevents urllib3 from configuring SSL appropriately and may cause certain SSL conne
ctions to fail. For more information, see https://urllib3.readthedocs.org/en/latest/security.html#insecureplatformwarning.
  InsecurePlatformWarning
  Downloading virtualenv-13.1.0-py2.py3-none-any.whl (1.7MB)
    100% |████████████████████████████████| 1.7MB 201kB/s
Installing collected packages: virtualenv
Successfully installed virtualenv-13.1.0
[root@localhost ~]#
```

5. General tips when it is slow

- Program thoughtfully:
  - Use vectorized functions
  - Avoid loops
  - Preallocate
  - Force type
  - Avoid copy-on-write
- Link to fast libraries (C/C++, Fortran, Java)
- Write low-level parts in C or Fortran
- Compile – jit
- Go parallel

6. Bridges

Python	→	R	http://rpython.r-forge.r-project.org/
Octave	→	Python	https://pypi.python.org/pypi/oct2py
R	→	Python	http://rpy.sourceforge.net/
Octave	→	R	https://cran.r-project.org/web/packages/RcppOctave
Python	→	Octave	https://github.com/daniel-e/pyoctave
R	→	Octave	http://www.omegahat.org/ROctave/

Summary

Octave, R, Python (and Julia)

Much more programmer-friendly than C/C++/Fortran

Still able to use fast compiled code

Focus on the unsolved problems

Try all and choose one