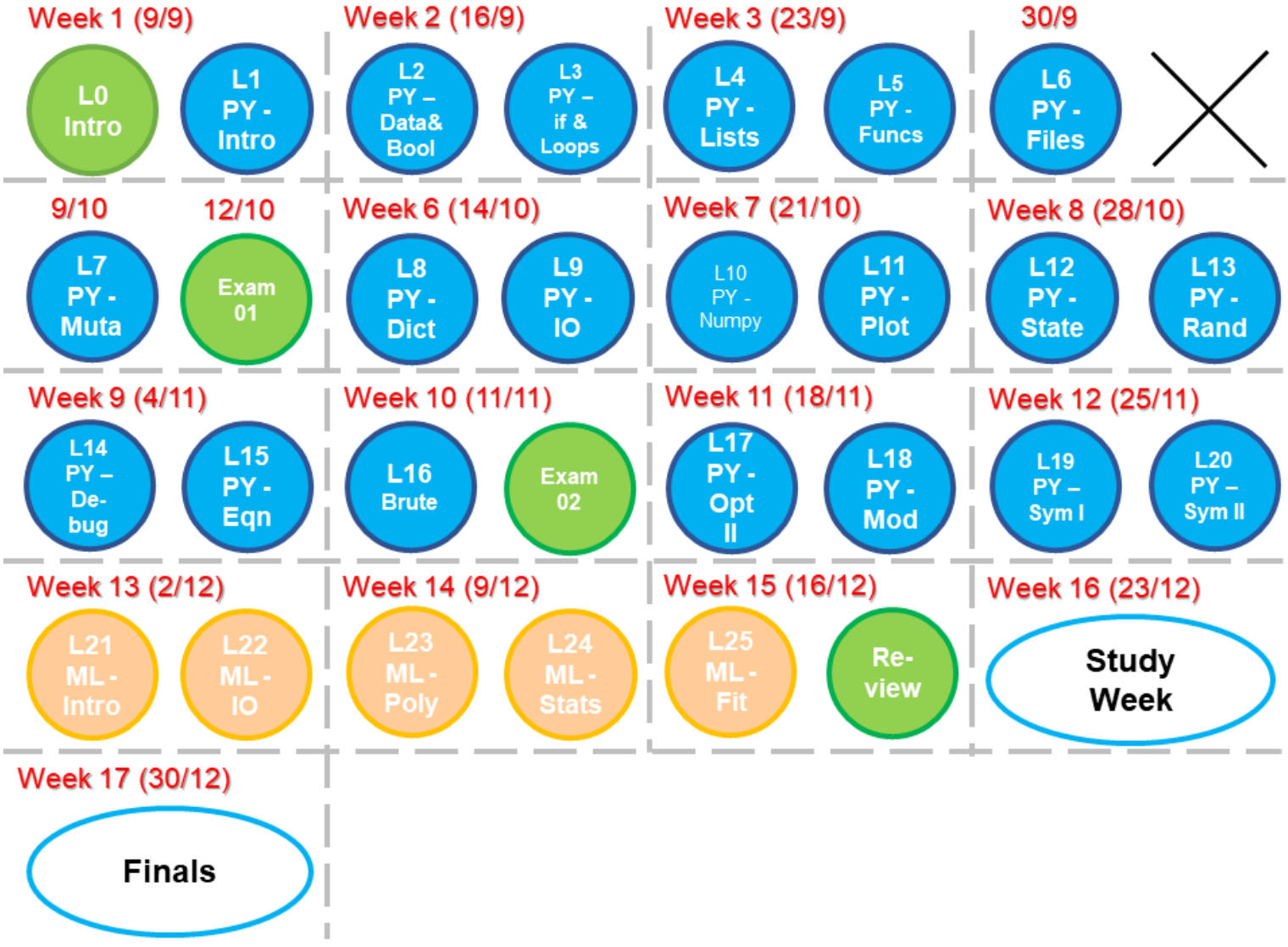
One last time

Python n MATLAB

Roadmap


Week 1 (9/9)
L0 Intro
L1 PY - Intro
Week 2 (16/9)
L2 PY – Data& Bool
L3 PY – if & Loops
Week 3 (23/9)
L4 PY - Lists
L5 PY - Funcs
30/9
L6 PY - Files
9/10
L7 PY - Muta
12/10
Exam 01
Week 6 (14/10)
L8 PY - Dict
L9 PY - IO
Week 7 (21/10)
L10 PY - Numpy
L11 PY - Plot
Week 8 (28/10)
L12 PY - State
L13 PY - Rand
Week 9 (4/11)
L14 PY – De-bug
L15 PY - Eqn
Week 10 (11/11)
L16 Brute
Exam 02
Week 11 (18/11)
L17 PY - Opt II
L18 PY - Mod
Week 12 (25/11)
L19 PY – Sym I
L20 PY – Sym II
Week 13 (2/12)
L21 ML - Intro
L22 ML - IO
Week 14 (9/12)
L23 ML - Poly
L24 ML - Stats
Week 15 (16/12)
L25 ML - Fit
Re-view
Week 16 (23/12)
Study Week
Week 17 (30/12)
Finals

Announcements

Check all your scores on RELATE between 23 - 27 Dec!

exam: `final exam` on 27 Dec 2019 9am

Paper Exam - MCQs + Short Questions + Coding Question/Section will indicate which is MATLAB, which is PYTHON

Question

Are you present today?

A Yes

B Yes

C Yes

D Yes

Recap Python

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

C. `import` this and that libraries; `library.functions`

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

C. `import` this and that libraries; `library.functions`

D. `str`, Attribute operator `.` , ASCII table, ' ', " ", """ """, ''' ''', + *

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

C. `import` this and that libraries; `library.functions`

D. `str`, Attribute operator `.` , ASCII table, ' ', " ", """ """, ''' ''', + *

E. "%d","%f","%c","%s"?

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

C. `import` this and that libraries; `library.functions`

D. `str`, Attribute operator `.` , ASCII table, ' ', " ", """ """, ''' ''', + *

E. "%d","%f","%c","%s"?

F. string index: `[ ]` starts from what?; `[-ve index]`, slice `[ a:b ]` => `[1:4]` includes what?, excludes what?

Python L01 - L05

A. `Operators: * / + - // % ** += -= *= /=`

B. Data Type: `int`, `float`, `complex` etc; Promotion: `int` -> `Float` -> `Complex`

C. `import` this and that libraries; `library.functions`

D. `str`, Attribute operator `.` , ASCII table, ' ', " ", """ """, ''' ''', + *

E. "%d","%f","%c","%s"?

F. string index: `[ ]` starts from what?; `[-ve index]`, slice `[ a:b ]` => `[1:4]` includes what?, excludes what?

G. string.methods like `ss.strip()`, `ss.replace(str1,str2)`, `ss.count(str)` `ss.isalnum()` and `ss.isXXXXX()`, etc

H. Do these functions return a value or modify `ss` immediately?

Python L03 - L05

A. `input( )` returns what data type?

Python L03 - L05

A. `input( )` returns what data type?

B. Boolean: `True, False, <, <=, >, >=, ==, !=, not, and, or`. Which has higher priority?

Python L03 - L05

A. `input( )` returns what data type?

B. Boolean: `True, False, <, <=, >, >=, ==, !=, not, and, or`. Which has higher priority?

C. Casting: `int( ), float( ), complex( ), str( )`

Python L03 - L05

A. `input( )` returns what data type?

B. Boolean: `True, False, <, <=, >, >=, ==, !=, not, and, or`. Which has higher priority?

C. Casting: `int( ), float( ), complex( ), str( )`

D. Conditions: `if... :, elif... :, else:,` indent needed (Compare to MATLAB)

Python L03 - L05

A. `input( )` returns what data type?

B. Boolean: `True, False, <, <=, >, >=, ==, !=, not, and, or`. Which has higher priority?

C. Casting: `int( ), float( ), complex( ), str( )`

D. Conditions: `if... :, elif... :, else:`, indent needed (Compare to MATLAB)

E. Loop: `while ... :, for... in... :, range( a, b, steps)`; includes `a` but excludes `b`

Python L03 - L05

A. `input( )` returns what data type?

B. Boolean: `True, False, <, <=, >, >=, ==, !=, not, and, or`. Which has higher priority?

C. Casting: `int( ), float( ), complex( ), str( )`

D. Conditions: `if... :, elif... :, else:,` indent needed (Compare to MATLAB)

E. Loop: `while ... :, for... in... :, range( a, b, steps)`; includes `a` but excludes `b`

F. `break`, `continue` what can `break` or `continue`? `break` vs `continue`

Example

```
x = 3
s = ("%i" % (x+1)) * x**(5%x)
print(s)
```

What does this program print?

A `333333333333`

B `444444444`

C `9999`

D `%i%i%i%i%i`

Example

```
x = 3
s = (”%i” % (x+1)) * x**(5%x)
print(s)
```

What does this program print?

A `333333333333`

B `444444444` ⋆(Trace the steps!)

C `9999`

D `%i%i%i%i%i`

Python L1 - L5

A. `list` for [many things, anything], indexing with `[ a ]` and slicing with `[ a:b ]`, negative indexing, `for` loop in `list`, `+ *` like `strings`

Python L1 - L5

A. `list` for [many things, anything], indexing with `[ a ]` and slicing with `[ a:b ]`, negative indexing, `for` loop in `list`, `+ *` like `strings`

B. list.methods like attributes that are functions: `.sort()`, `.append()`, `.extend()`, `etc`. Functions like `max(x)`, `min(x)`, `len(x)` Which methods return `none`?

Python L1 - L5

A. `list` for [many things, anything], indexing with `[ a ]` and slicing with `[ a:b ]`, negative indexing, `for` loop in `list`, `+ *` like `strings`

B. list.methods like attributes that are functions: `.sort(), .append(), .extend(), etc`. Functions like `max(x), min(x), len(x)` Which methods return `none`?

C. function: `def fName(yy, zz):`, `import`, variable scope, `return`, default value like `yy = ??`

Python L1 - L5

A. `list` for [many things, anything], indexing with `[ a ]` and slicing with `[ a:b ]`, negative indexing, `for` loop in `list`, `+ *` like `strings`

B. list.methods like attributes that are functions: `.sort(), .append(), .extend(), etc`. Functions like `max(x), min(x), len(x)` Which methods return `none`?

C. function: `def fName(yy, zz):`, `import`, variable scope, `return`, default value like `yy = ??`

D. Multiple Loops in loop, recursive function! Do you know how to write these?

Python L6 - L8

A. FILE: `open('????','r' or 'w')`, `.write()`, `.close()` text file, `.read()`, `.readlines()`, `.split("?")`, `"?".join()`,

B. `break, continue, zip, enumerate`

Python L6 - L8

A. FILE: `open('????','r' or 'w')`, `.write()`, `.close()` text file, `.read()`, `.readlines()`, `.split("?")`, `"?".join()`,

B. `break, continue, zip, enumerate`

C. Mutable vs Immutable, `list? dict? tuple? str? int? float? bool?` and others? mutable can change without return from function

Python L6 - L8

A. FILE: `open('????','r' or 'w')`, `.write()`, `.close()` text file, `.read()`, `.readlines()`, `.split("?")`, `"?".join()`,

B. `break, continue, zip, enumerate`

C. Mutable vs Immutable, `list? dict? tuple? str? int? float? bool?` and others? mutable can change without return from function

D. `tuple` similar to `list` but immutable;

Python L6 - L8

A. FILE: `open('????','r' or 'w')`, `.write()`, `.close()` text file, `.read()`, `.readlines()`, `.split("?")`, `"?".join()`,

B. `break, continue, zip, enumerate`

C. Mutable vs Immutable, `list? dict? tuple? str? int? float? bool?` and others? mutable can change without return from function

D. `tuple` similar to `list` but immutable;

E. 2D list or List in list with `[a][b]` indexing,

F. create Dictionary using dictionary[key] = value or `{}`, keys can be any immutable type, indexing using `[]`, `d.items(), d.keys(), d.values()`, Applications: encoding, decoding, counter/accumulator

Question 1

```
d = { 'a':2, 'c':3, 'b':1 }
x = d[ 'a' ] + d[ 'c' ]
```

What is the final value of `x`?

A `4`

B `'ac'`

C `'5'`

D `5`

Question 1

```
d = { 'a':2, 'c':3, 'b':1 }
x = d[ 'a' ] + d[ 'c' ]
```

What is the final value of `x`?

A `4`

B `'ac'`

C `'5'`

D `5` ⋆

Example return List?

```
def appender( q ):
    z = [ ]
    z.append( 3 )
    q.append( 3 )

a = [ ]
for i in range( 3 ):
    appender( a )
print( a )
print( z )
```

ans:

Example return List?

```
def appender( q ):
    z = [ ]
    z.append( 3 )
    q.append( 3 )

a = [ ]
for i in range( 3 ):
    appender( a )
print( a )
print( z )
```

ans:
a = [3, 3, 3]
z Error!

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

E `x.shape`, `x.dtype`, `x*x` elementwise (compare with MATLAB)

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

E `x.shape`, `x.dtype`, `x*x` elementwise (compare with MATLAB)

F `numpy.sin(x)`, `numpy.exp(x)` and others; array`.all( )` and array`.any( )`

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

E `x.shape`, `x.dtype`, `x*x` elementwise (compare with MATLAB)

F `numpy.sin(x)`, `numpy.exp(x)` and others; array`.all( )` and array`.any( )`

G np.array is mutable

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

E `x.shape`, `x.dtype`, `x*x` elementwise (compare with MATLAB)

F `numpy.sin(x)`, `numpy.exp(x)` and others; array`.all( )` and array`.any( )`

G np.array is mutable

H `x.sort(i) 0=column, 1,nothing=row`, `x.tolist()`, `x.argsort()`

Lec 9-10

A `import numpy as np`

B `numpy.array` vs `list`, `numpy` vs `math`

C `x = np.array( [ n1, n2 ] )` and `x = np.array( [ [ n11, n12 ], [ n21, n22 ] ] )`

D `numpy.zeros(x,y)`, `numpy.ones(x,y)`, `numpy.eye(x)`

E `x.shape`, `x.dtype`, `x*x` elementwise (compare with MATLAB)

F `numpy.sin(x)`, `numpy.exp(x)` and others; array`.all( )` and array`.any( )`

G np.array is mutable

H `x.sort(i) 0=column, 1,nothing=row`, `x.tolist()`, `x.argsort()`

I `np.linspace(start, finish, n)`

Lec 11-12

A `matplotlib` and `import matplotlib.pyplot as plt`

B `plt.plot(x,y, ....)`,
`'r','g','b','k','y','-','- -','o','x'`,
`plt.show()`

C `xlim(...), ylim(...), xticks(...), yticks(...), xlabel(...), ylabel(...), legend(...), title(...), savefig(...)`

Lec 11-12

A `matplotlib` and `import matplotlib.pyplot as plt`

B `plt.plot(x,y, ....)`, `'r','g','b','k','y','-','- -','o','x'`, `plt.show()`

C `xlim(...)`, `ylim(...)`, `xticks(...)`, `yticks(...)`, `xlabel(...)`, `ylabel(...)`, `legend(...)`, `title(...)`, `savefig(...)`

D Modeling - Analytical, Numerical

E Forward and backward difference

Lec 13-14

A
```
import numpy.random as npr
```

B
```
npr.uniform(x,y,size=[?]),
npr.randint(x,y,size=[?]),
npr.normal(size=?)*y + x
```

Lec 13-14

A `import numpy.random as npr`

B `npr.uniform(x,y,size=[?]),`
`npr.randint(x,y,size=[?]),`
`npr.normal(size=?)*y + x`

C `npr.choice( ), npr.shuffle( )`

Lec 13-14

A `import numpy.random as npr`

B `npr.uniform(x,y,size=[?])`,
`npr.randint(x,y,size=[?])`,
`npr.normal(size=?)*y + x`

C `npr.choice( )`, `npr.shuffle( )`

D Errors, `SyntaxError`, `NameError`, `TypeError`, `ZeroDivisionError`, `FileNotFoundError`, `IndexError`, `KeyError`, `IndentationError`...

Lec 13-14

A `import numpy.random as npr`

B `npr.uniform(x,y,size=[?])`,
`npr.randint(x,y,size=[?])`,
`npr.normal(size=?)*y + x`

C `npr.choice( )`, `npr.shuffle( )`

D Errors, `SyntaxError`, `NameError`, `TypeError`, `ZeroDivisionError`, `FileNotFoundError`, `IndexError`, `KeyError`, `IndentationError`...

E `try:...except xxxx:... except yyyy:...`
`else:... finally:...`

Lec 13-14

A `import numpy.random as npr`

B `npr.uniform(x,y,size=[?])`,
`npr.randint(x,y,size=[?])`,
`npr.normal(size=?)*y + x`

C `npr.choice( )`, `npr.shuffle( )`

D Errors, `SyntaxError`, `NameError`, `TypeError`, `ZeroDivisionError`, `FileNotFoundError`, `IndexError`, `KeyError`, `IndentationError`...

E `try:...except xxxx:... except yyyy:... else:... finally:...`

F Compare float: `math.isclose(a, b)`, `np.isclose(a, b)`, `np.allclose(a, b)`

Lec 15 - 16

A Equation as numeric; (Fast) Analytical, Series, Monte Carlo (Slow)

B `import timeit`, `timeit.timeit(code, number = ?)`, code = ”’ def xxx(yyy): ..... ”’

Lec 15 - 16

A Equation as numeric; (Fast) Analytical, Series, Monte Carlo (Slow)

B `import timeit`, `timeit.timeit(code, number = ?)`, code = ”’ def xxx(yyy): ..... ”’

C `import scipy.optimize as sco`, `sco.newton( f, x0 )`, `sco.fmin( f, x0 )`, `sco.minimize( f, x0 )` and `def f( yyy ): .....`

Lec 15 - 16

A Equation as numeric; (Fast) Analytical, Series, Monte Carlo (Slow)

B `import timeit`, `timeit.timeit(code, number = ?)`, code = ”’ def xxx(yyy): ..... ”’

C `import scipy.optimize as sco`, `sco.newton( f, x0 )`, `sco.fmin( f, x0 )`, `sco.minimize( f, x0 )` and `def f( yyy ): .....`

D Brute-Force method for optimization; `import itertools`, `itertools.combinations( a, n )`, `itertools.combinations_with_replacement( a, n )`, `itertools.product( a, b )` or `itertools.product( a, repeat = n )`

Lec 17 - 18

A figure-of-merit; Idea, advantages„ disadvantages and steps to implement hill-climbing, random walk

B `pip list`, `pip install`, `-upgrade`, `pip uninstall`, how you make your own package

Lec 19 - 20

A import sympy

B Define variable: `x, a, b = sympy.S('x,a,b')`

Lec 19 - 20

A import sympy

B Define variable: `x, a, b = sympy.S('x,a,b')`

C Define equation: `eq1 = a*x + b*y`

Lec 19 - 20

A import sympy

B Define variable: `x, a, b = sympy.S('x,a,b')`

C Define equation: `eq1 = a*x + b*y`

D sympy library: `sympy.exp(..)`, `sympy.sqrt(..)`, etc.

E `sympy.I, sympy.E, sympy.pi`

Lec 19 - 20

A import sympy

B Define variable: `x, a, b = sympy.S('x,a,b')`

C Define equation: `eq1 = a*x + b*y`

D sympy library: `sympy.exp(..)`, `sympy.sqrt(..)`, etc.

E `sympy.I, sympy.E, sympy.pi`

F
```
ans = sympy.solve((eq1,eq2),(x,y)),
ans[0].subs(a,1).subs(b,2)
```

G `sympy.expand(..)`, `sympy.factor(..)`, `symsy.simplify(..)`

H `sympy.plotting.plot( eqn, (x, LL, UL ))`

Lec 19 - 20

A `sympy.diff( eq1, x, n )`, nth order of diffferentiation

B `sympy.integrate( eq1, x )` or `sympy.integrate( eq1, ( x, Lower, Upper ) )`

C double integration

Lec 19 - 20

A `sympy.diff( eq1, x, n )`, nth order of diffferentiation

B `sympy.integrate( eq1, x )` or `sympy.integrate( eq1, ( x, Lower, Upper ) )`

C double integration

D `sympy.series( eqn, variable, which-Point, num-terms ).removeO()`

Example

Multiple integrals can also be handled smoothly.

$$\int_0^1 dy \int_{-1}^{+1} dx \, 2\sin^2 x + 3y$$

Example

Multiple integrals can also be handled smoothly.

$$\int_0^1 dy \int_{-1}^{+1} dx\, 2\sin^2 x + 3y$$

```
sympy.integrate(sympy.integrate(2*sympy.sin(x)**2
                +3*y,( x,-1,+1 ) ), ( y,0,1 ) )
```

Lec 22

A MATLAB; `A = [ 1 2; 3 4 ]`
`* + - / ^ .* .^ ./`

B indexing `A( 1, 2 )` = `A( 2 ) = 3` and `A( 1,: );`
`ones( n, n );zeros( n, n );eyes( n, n )`

C element operation `A.*A`; the usual maths functions

D `function [ output1, output2 ] =`
`functionName ( input ) ... output1 = xxxx;`
`... end;` No need return but need `end`

E define function in a `.m` file NOT on terminal; fileName.m same as functionName

F Difference between `'This is a char array'` and `"This is a string"`.

Lec 22

A `for i = start:step:last ... end; continue; break`

B `linspace(start,end,number)`

C `while ... end`

D `if.... elseif.... else.... end`

`< > <= >= == ~= && ||`

E `True == 1, False == 0`

F `rand( n, m ), randn( n, m ), randi( [ a, b ], m , n )`

Lec 23

A Most variables are created as `double`

B save as a matlab file `save('fileName', 'variable')` 'variable' optional

C append into text file `save filename.txt variableName -ascii -append`

D `load('matlabFile'), p = imread( ? )` to read image into MATLAB' `image( p )` show image

E `importdata('fileName')`

F `figure` creates a new figure space for plotting

G `plot(...), hold on`; single line function `f = @(t) cos( 3*t )`

Lec 24-25

`polyN = [ 1 2 3 0 1 ]`

A What highest power? `polyval( polyN, x0 );`
`polyint( p );polyder( p );conv([...],[...])`

B `r = roots( p );poly( r );fzero( f, x0 );`
`fminbnd(f, x1, x2 )` When `f` or `@f`

C `trapz( x, y )`

D `interp1( x, y, x0, 'method' )` method -
`'nearest', 'pchip', 'linear'`

E `inv(A) * b` or `A  b`

F `polyfit( x, y, o );spline( x, y, x-est )`

Example

```
A1 = [1 2].*2
A2 = [1 2]*2

C1 = [1 2].+2
C2 = [1 2]+2

B1 = [1 2].^2
B2 = [1 2]^2
```

Example

```
A1 = [1 2].*2 ***
A2 = [1 2]*2 ***

C1 = [1 2].+2 - error
C2 = [1 2]+2 ***

B1 = [1 2].^2 ***
B2 = [1 2]^2 - error for elementwise
    - but ok for matrix operations
```