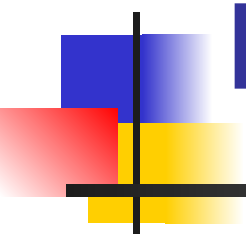
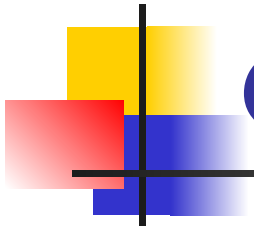


Python Introduction

Session 1

Python Workshop for Benjama EEP

http://goo.gl/Zmbzpj

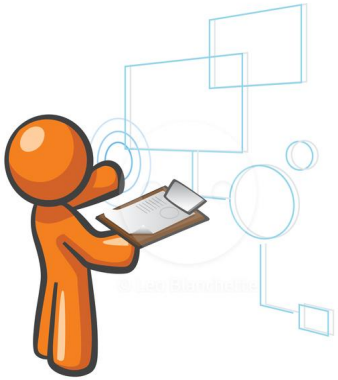


Objectives

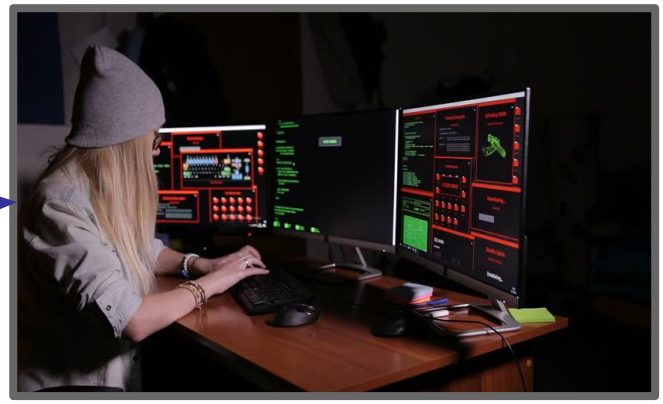
- what is programming?
- variables and data types
- input/output
- module and calling functions

What is programming?

Process



Design & Analyze



Coding / Implementing

Today, we'll be...

Coding / Implementing

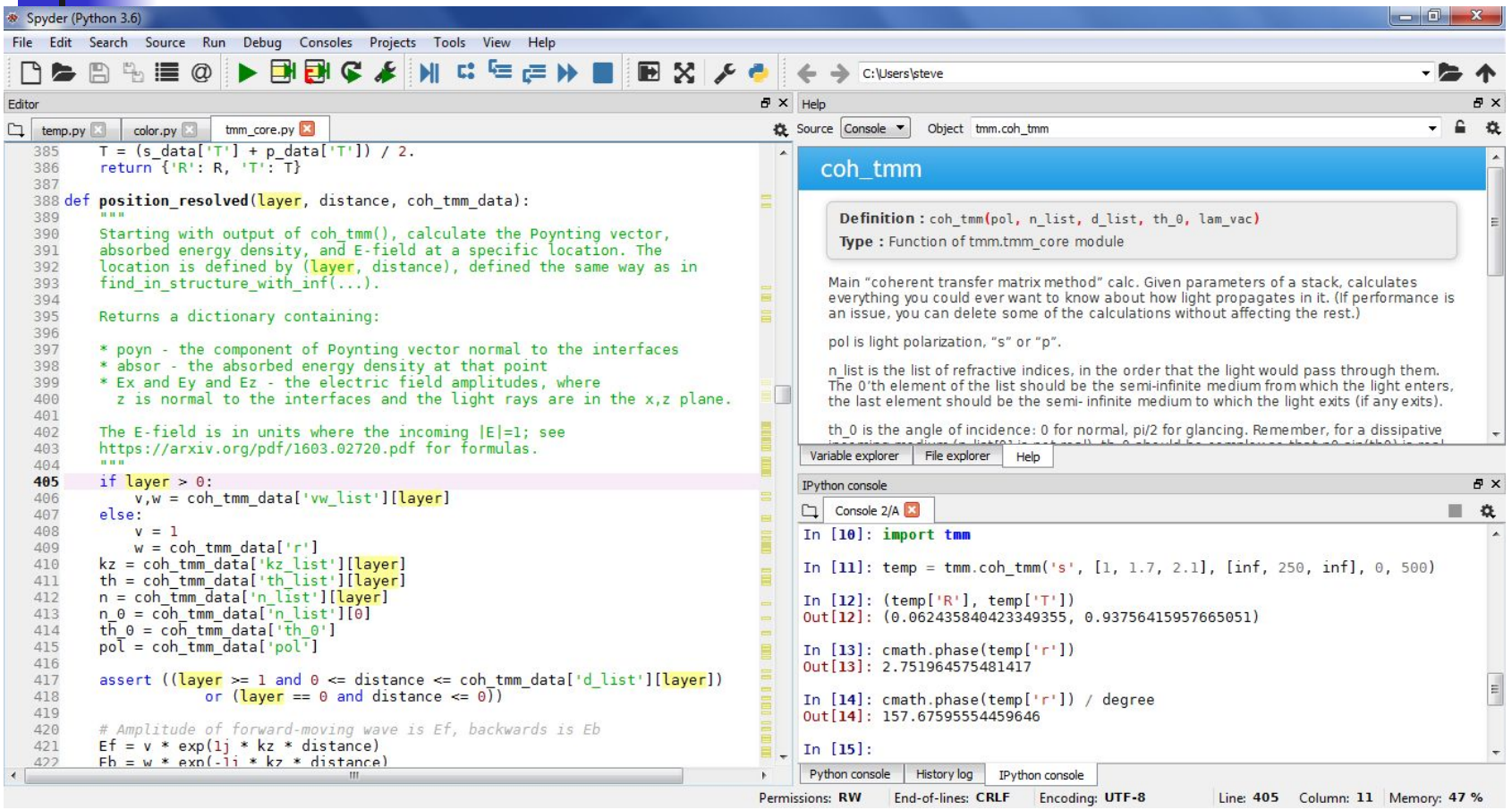
© www.spreadshirt.com

- General purpose
- Object-oriented
- Interpreted
- Focus on readability & productivity
- Strongly typed and dynamically typed

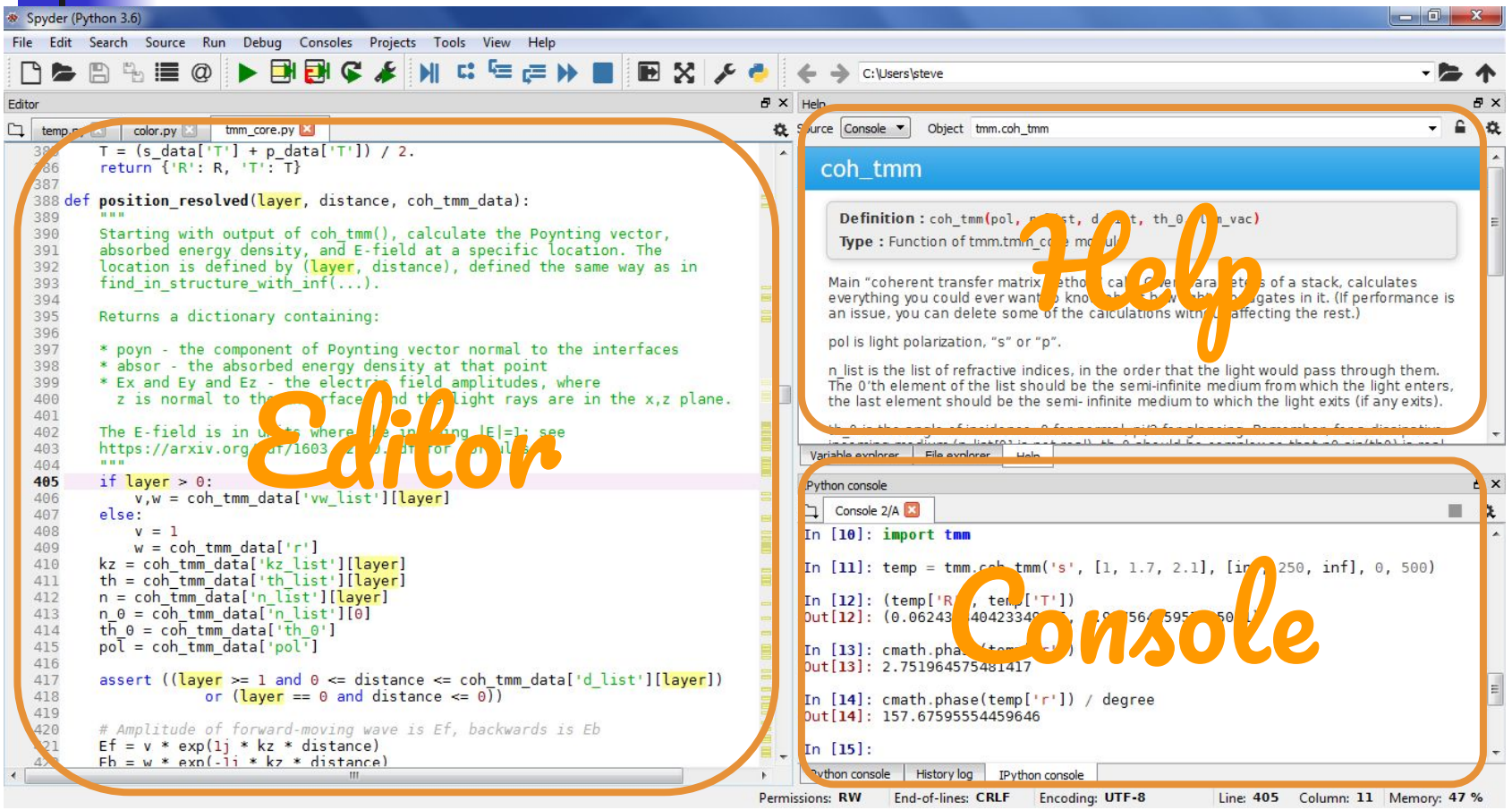
Workshop tools

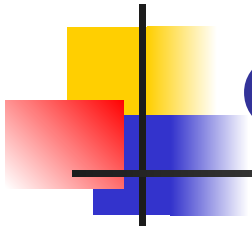
https://www.anaconda.com/download

let's start Spyder - IDE



Spyder - IDE


Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor
temp.py color.py tmm_core.py
def position_resolved(layer, distance, coh_tmm_data):
"""
Starting with output of coh_tmm(), calculate the Poynting vector,
absorbed energy density, and E-field at a specific location. The
location is defined by (layer, distance), defined the same way as in
find_in_structure_with_inf(...).
Returns a dictionary containing:
* poyn - the component of Poynting vector normal to the interfaces
* absor - the absorbed energy density at that point
* Ex and Ey and Ez - the electric field amplitudes, where
z is normal to the interfaces and the light rays are in the x,z plane.
The E-field is in units where the incoming |E|=1; see
https://arxiv.org/abs/1603.
for details.
"""
if layer > 0:
v,w = coh_tmm_data['vw_list'][layer]
else:
v = 1
w = coh_tmm_data['r']
kz = coh_tmm_data['kz_list'][layer]
th = coh_tmm_data['th_list'][layer]
n = coh_tmm_data['n_list'][layer]
n_0 = coh_tmm_data['n_list'][0]
th_0 = coh_tmm_data['th_0']
pol = coh_tmm_data['pol']
assert ((layer >= 1 and 0 <= distance <= coh_tmm_data['d_list'][layer])
or (layer == 0 and distance <= 0))
# Amplitude of forward-moving wave is Ef, backwards is Eb
Ef = v * exp(1j * kz * distance)
Editor
Help
Console
Object tmm.coh_tmm
coh_tmm
Definition : coh_tmm(pol, n_list, d_list, th_0, lam_vac)
Type : Function of tmm.tmm_core module
Main "coherent transfer matrix" method. Calculates parameters of a stack, calculates everything you could ever want to know about how it propagates in it. (If performance is an issue, you can delete some of the calculations without affecting the rest.)
pol is light polarization, "s" or "p".
n_list is the list of refractive indices, in the order that the light would pass through them. The 0'th element of the list should be the semi-infinite medium from which the light enters, the last element should be the semi-infinite medium to which the light exits (if any exits).
Variable explorer File explorer Help
Python console
Console 2/A
In [10]: import tmm
In [11]: temp = tmm.coh_tmm('s', [1, 1.7, 2.1], [inf, 250, inf], 0, 500)
In [12]: (temp['R'], temp['T'])
Out[12]: (0.06243...
In [13]: cmath.phase(temp['r'])
Out[13]: 2.751964575481417
In [14]: cmath.phase(temp['r']) / degree
Out[14]: 157.67595554459646
In [15]:
Python console History log IPython console
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 405 Column: 11 Memory: 47 %



Code - app101.py

- เขียนและทดสอบโปรแกรมแรกได้
  - Edit & Run

```
print('สวัสดี')
```

Code - app102.py

- เรียกใช้คำสั่งเพื่อให้ผู้ใช้กรอกข้อมูล
  - `input()`
  - `input('กรอกชื่อ ')`
  - `input('กรอกส่วนสูง ')`

```
print('สวัสดี')
input('กรอกชื่อ ')
```

Variables - app103.py

- กำหนดชื่อเพื่ออ้างถึงข้อมูลที่ผู้ใช้กรอก
- นำมาแสดงผล

```
name = input('กรอกชื่อ ')
print('สวัสดี ')
print(name)
```

Variables - app104.py

- การแสดงข้อมูลติดกัน

```
name = input('กรอกชื่อ ')
print('สวัสดี คุณ'+name)
```

Variables - ชนิดของข้อมูล

- **ข้อความ** - ตัวอักษรตัวเลขเรียงติดกันอยู่ระหว่าง ‘’
  - str
  - ‘กรอกชื่อ ’
  - “สวัสดีครับ Dr.Tom & Jerry”
- **ตัวเลข** - ตัวเลขที่สามารถนำมาคำนวณได้
  - 20 ⟶ int
  - 3.55 ⟶ float

Variables - app105.py

- รับชื่อและรับตัวเลขจำนวนเต็ม

```
name = input('กรอกชื่อ ')
print('สวัสดี คุณ'+name)
year = int(input('เกิดเมื่อ พ.ศ. '))
print('ปีเกิด '+str(year))
```

Variables - app106.py

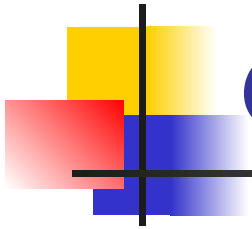
- รับชื่อและรับตัวเลขทศนิยม

```
name = input('กรอกชื่อ ')
print('สวัสดี คุณ'+name)
gpa = float(input('เกรดเฉลี่ย '))
print('เกรดเฉลี่ย '+str(gpa))
```

Variables - app107.py

- รับชื่อและปีพ.ศ.เกิด คำนวณอายุ

```
name = input('กรอกชื่อ ')
print('สวัสดี คุณ'+name)
year = int(input('ปีเกิด พ.ศ. '))
age = 2561-year
print('คุณอายุ '+str(age))
```



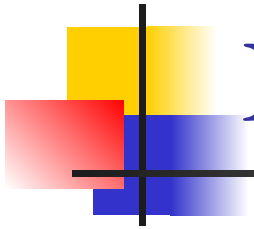
Operators

- คูณ - ยกกำลัง
  - 4*5 2**8
- หาร - หารตัดเศษ - หารเอาเศษ
  - 9/4 9//4 9%4
- บวก - ลบ
  - 9+4 9-4

Variables - app108.py

- คำนวณหาความชัน

```
x1 = float(input('x1 '))
y1 = float(input('y1 '))
x2 = float(input('x2 '))
y2 = float(input('y2 '))
m = (y2-y1)/(x2-x1)
print('ความชัน '+str(m))
```

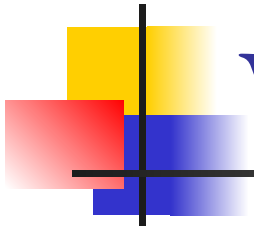


Exercise

1. โปรแกรมแปลงขนาดที่ดินจาก***ตารางวา***เป็น***ตารางเมตร***
2. โปรแกรมหาพื้นฐานของวงกลมโดยให้ผู้ใช้กรอกรัศมี

   (ใช้ $\pi = 22/7$)
3. โปรแกรมแปลงอุณหภูมิจาก *Fahrenheit* เป็น *Celsius*

   (สมการ F = C * 9/5 + 32)



Variables - list

- ลำดับของ **str**

  speakers = [‘K’, ‘Kai’, ‘Or’, ‘Paul’, ‘Tom’, ‘Woot’]

- ลำดับของ **int**

  heights = [180, 164, 165, 165, 180, 170]

- ลำดับของ **float**

  popularity = [3.25, 3.9, 4, 3, 3.5, 3.44]

Speakers in one line - app109.py

- รับชื่อวิทยากรที่อยู่ใน 1 บรรทัด

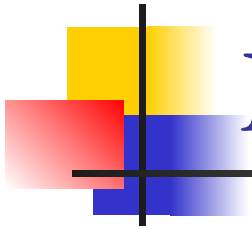
```
line = input('กรอกรายชื่อวิทยากร ')
print('line '+line)
speakers = line.split(' '))
print('speakers '+str(speakers))
หรือ
print(str(speakers))
```

Speakers Data - app110.py

- อ่านข้อมูลวิทยากรที่เป็นลำดับของตัวเลข

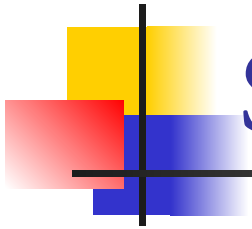
```
line = input('ส่วนสูง ')
height = map(int, line.split(' '))
print(height)

line = input('pop ')
pop = map(float, line.split(' '))
print(pop)
```



Module and Calling Functions

- จะคำนวณหา ค่าเฉลี่ย ค่ากลางอย่างไร?
  - เรียกใช้ module เสริมชื่อ statistics
  - `import statistics`
- หลังจากที่ได้ข้อมูลจากผู้ใช้แล้วเรียกใช้
  - `statistics.median(height)`
  - `statistics.mean(pop)`
  - `statistics.mode()/median()/stdev()`



Statistics - app111.py

```
import statistics

line = input('speakers ')
speakers = map(str, line.split(' '))
print(speakers)

line = input('height ')
height = map(int, line.split(' '))
print(statistics.mode(height))

line = input('pop ')
pop = map(float, line.split(' '))
print(statistics.mean(pop))
```