



# Python on TI-Nspire CX II

**T3 Europe  
April 16 2020**

# Presenters



**Marshal S Chhaya**  
Product Development  
Texas Instruments  
Dallas, TX

@hschhaya



**Michel G. Stella**  
Product Development  
Texas Instruments  
Dallas, TX

mstella@ti.com



**Steve DeBauge**  
Product Strategy  
Texas Instruments  
Dallas TX

# Why Python?



- Python is a programming language that can support students throughout their educational career (middle school through college) and into their professional career
- Python is one of the fastest growing programming languages in academia and industry
- Adding Python to the TI-Nspire CX II platform is a next step in our ongoing support for coding and STEM

# What makes it special on TI-Nspire CX II?



- **Designed for education and ease-of-use for new learners**

- Discoverable, menu-driven syntax
- Inline prompts to guide beginners
- Syntax highlighting and help through context-sensitive tool tips
- Automated indentation with visible indentation marks (◆◆◆◆)



- **Physical computing**, robotics, graphics, image processing and more
- The TI-Nspire **document model** to enrich the instructional use experience
- **Interoperability**: Data can be exchanged between Python and TI-Nspire applications to create a math, science and coding ecosystem
- Hub activities offer a gentle introduction to **object-oriented programming (OOP)**
- No additional configuration, computer, building infrastructure or IT personnel required



# Availability – Fall 2020

Supported



TI-Nspire CX II-T handhelds

Desktop software



TI-Nspire CX II-T and  
TI-Nspire CX II-T CAS



TI-Nspire CX Student SW  
TI-Nspire CX CAS Student SW  
TI-Nspire CX Premium Teacher SW  
Windows and Mac OS

Not Supported

TI-Nspire CX handhelds



TI-Nspire iPad Apps



# Where are TI Python Solutions Relevant?

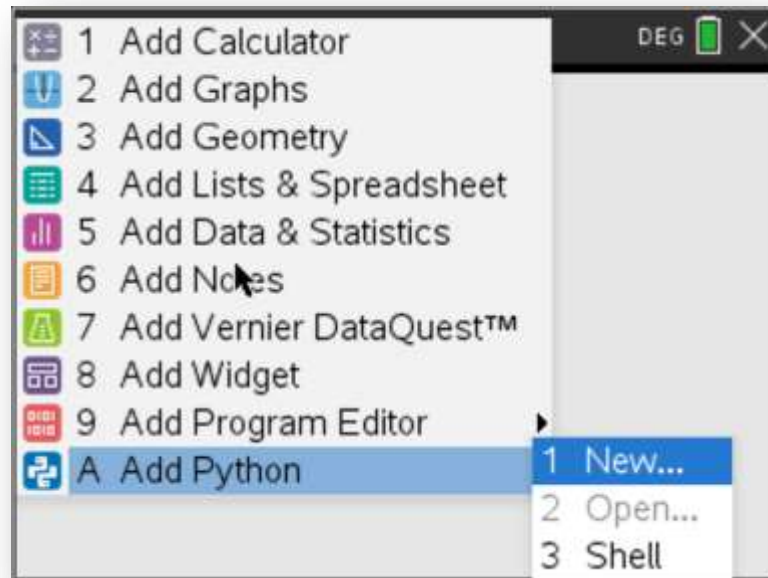


- **STEM Projects** – after school clubs and camps
  - Augmenting existing TI STEM Projects
    - <http://education.ti.com/stemprojects>
  - Opens opportunity for new project activities and innovations
- **Computer Science**
  - Ideal portable solution for “**physical computing**”
    - High student engagement through hands-on projects with Python coding
  - “**Object oriented**” language ... a modern paradigm
    - A meaningful difference for computer science
- **Math and Science**
  - Where coding supports mathematics and science curriculums
  - With teachers who are motivated to introduce coding in math and science



# Python | in the TI-Nspire World

- Python is at the same level as the other TI-Nspire CX II applications
- Python programs are part of the TNS file
  - Allows additional information (instructions, graphs etc.) to be included with Python programs
- A TNS file can contain multiple Python programs
- New “PyLib” folder allows use of Python programs as “libraries” across all TNS documents



# Python Modules

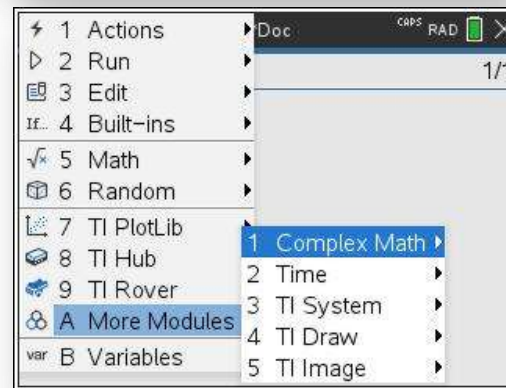
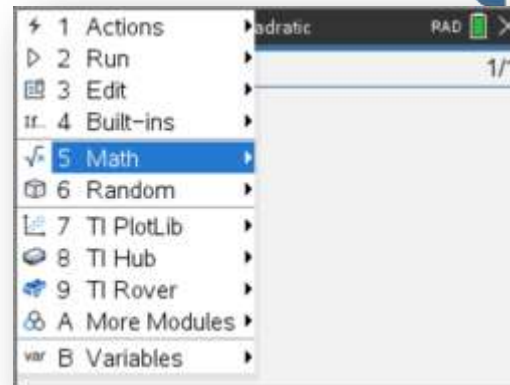


- Built-in

- **math**: special calculator functions (sin, sqrt, abs, etc.)
- **random**: random number generation
- **cmath**: math with complex numbers
- **time**: time measurement functions

- TI created

- **ti\_system**: interface to TI-Nspire CX II variables
- **ti\_hub**: interface to TI-Innovator Hub
- **ti\_rover**: interface to TI-Rover
- **ti\_plotlib**: simple plotting functions
- **ti\_draw**: geometry graphics
- **ti\_image**: image processing





# Python Environments



## Editor

```
1.1 1.2 1.3 pythagoras RAD [X]
pythagoras.py 1/10
def f(a,b,c):
    return(a**2+b**2-c**2)

def trirec(a,b,c):
    print("a:", a, "b:", b, "c:", c)
    if f(a,b,c)==0 or f(a,c,b)==0 or f(b,c,a)==0:
        print("right triangle")
    else:
        print("non-right triangle")
```

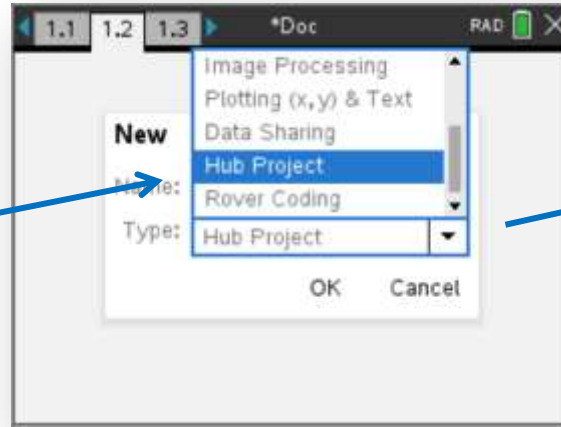
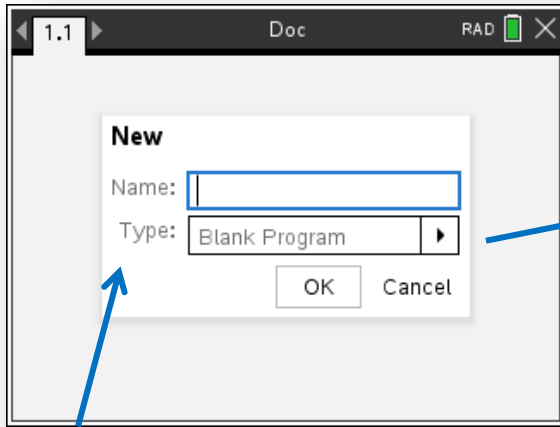
**Write, save, run**

## Shell

```
1.1 1.2 1.3 pythagoras RAD [X]
Python Shell 9/9
>>>#Running pythagoras.py
>>>from pythagoras import *
>>>trirec(3,4,5)
a: 3 b: 4 c: 5
right triangle
>>>trirec(5,6,7)
a: 5 b: 6 c: 7
non-right triangle
>>>|
```

**Interact**

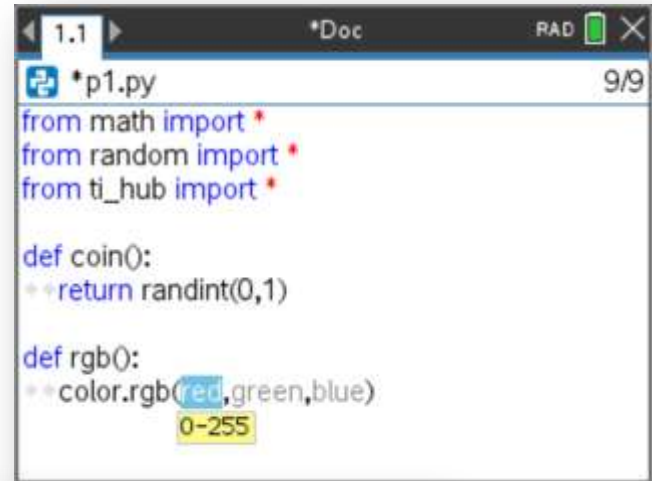
# Program templates – for new programs



Optional method to pick a program type – pastes the appropriate “import” statements  
Reduces need to import individual modules  
Default is a ‘Blank Program’.

# Python Editor

- Write, save and run Python programs
- Syntax highlighting + auto-indentation
- In-line prompts to guide with functions
- Tool-tips to show range of valid values
- “var” key to list global variables in use
  - Functions have a special icon
- Keypad shortcuts



```
1.1 *Doc RAD 9/9
p1.py
from math import *
from random import *
from ti_hub import *

def coin():
    return randint(0,1)

def rgb():
    color.rgb(red,green,blue)
           0-255
```



```
1.1 1.2 1.3 *Doc cap? RAD
eft3.py saved successfully

x=2
y=4.5
z=pi

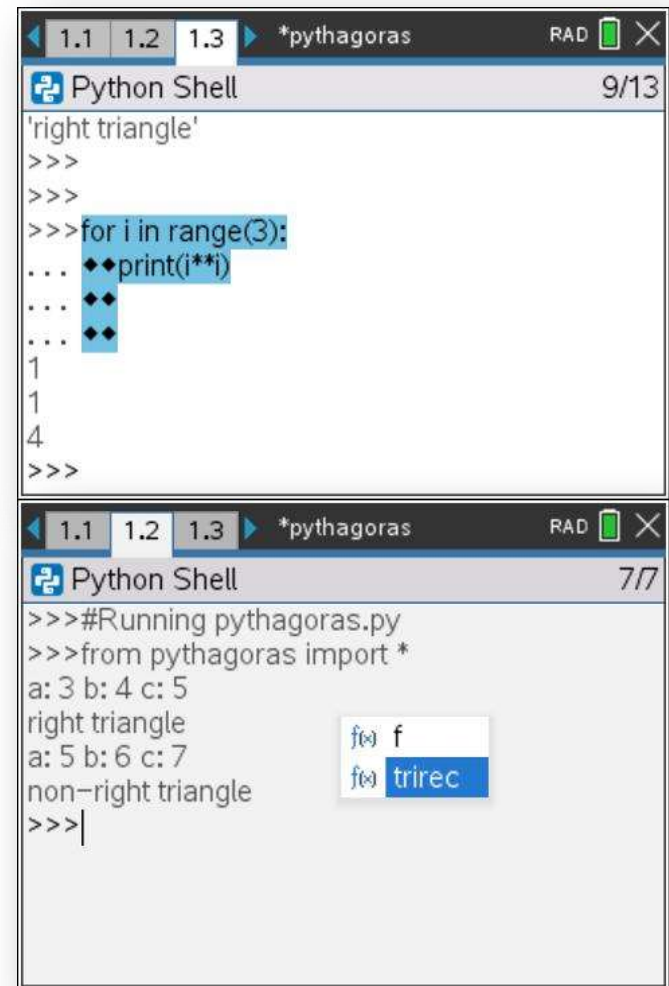
def f1(n):
    return n**2

def f2(n):
    return n**3
```

f() f1  
f() f2  
var x  
var y  
var z

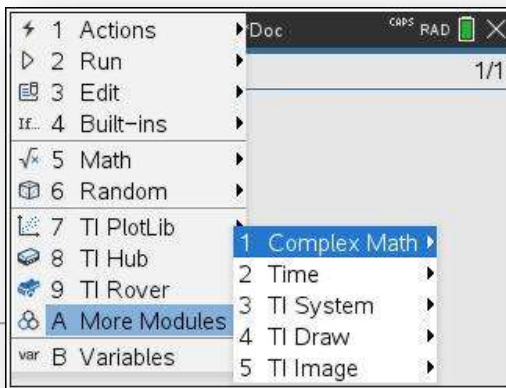
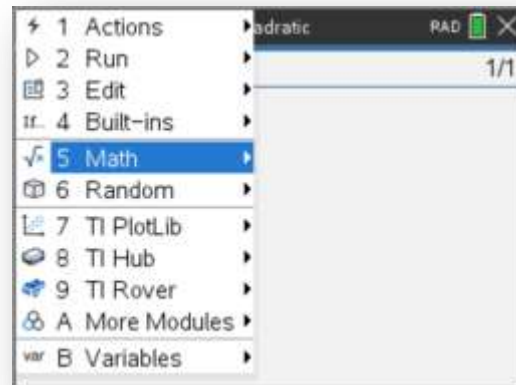
# Python Shell

- Interactive Python environment
- Convenient to test small code fragments
- Interaction with shell output is similar to Calculator
  - Select previous inputs and outputs for re-use
- “var” key lists functions from Python program



# TI Modules

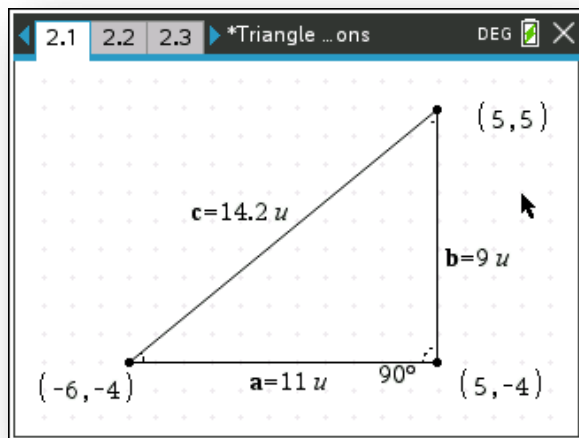
- TI modules enable Python programs to interact with TI-Innovator Hub, TI-Rover and other applications
  - **ti\_system**: interface to TI-Nspire CX II variables
  - **ti\_hub**: interface to TI-Innovator Hub
  - **ti\_rover**: interface to TI-Rover
  - **ti\_plotlib**: simple plotting functions
  - **ti\_draw**: geometry graphics
  - **ti\_image**: image processing



# ti\_system example

- Interact with TI-Nspire variables
  - **Store** values and lists created in Python to use in TI-Nspire native apps.
  - **Recall** TI-Nspire native variables and lists to use in Python
  - **Evaluate** TI-Nspire native functions in Python

G&G: triangle with sides of length a, b, and c



Python Editor: recall values a, b, and c

```
right_triangle_abc.py 1/18
from ti_system import*

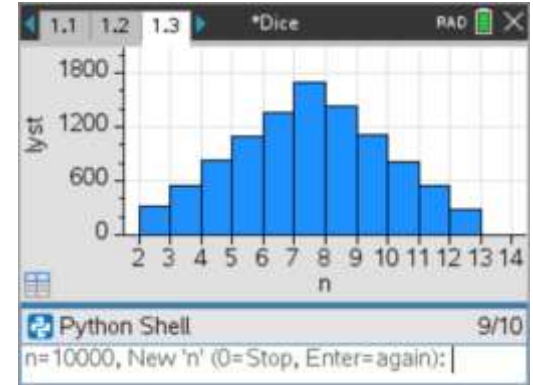
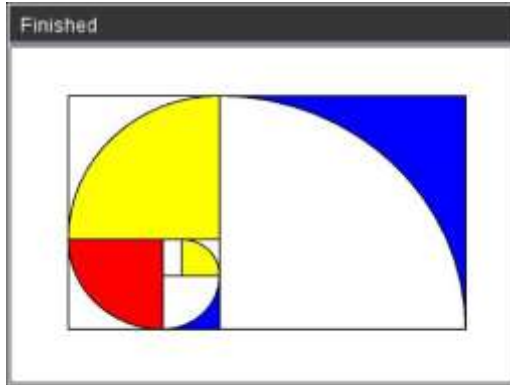
def rect_abc():
    a=recall_value("a")
    b=recall_value("b")
    c=recall_value("c")
    L=[a,b,c]
    L.sort()
    a = L[0]
    b = L[1]
    c = L[2]
```

Python Shell: determine if right triangle

```
Python Shell 5/5
>>>#Running right_triangle_abc.py
>>>from right_triangle_abc import *
Triangle Δ(9.52,11.00,11.43)
is NOT a right triangle
>>>|
```

# PYTHON DEMO

# Examples of Python programs on TI-Nspire CX II



```
1.1 1.2 1.3 Bert_Gold_ral RAD 13/39
golden_spiral.py
def square(x,y,d,h,c1,c2):
    x,y=x,13-y
    x1=x-hx[h]*d
    y1=y-hy[h]*d
    set_color(c1[0],c1[1],c1[2])
    fill_rect(x,y,d,d)
    set_color(c2[0],c2[1],c2[2])
    fill_arc(x1,y1,2*d,2*d,h*90,90)
    set_color(0,0,0)
    draw_rect(x,y,d,d)
    draw_arc(x1,y1,2*d,2*d,h*90,90)
```

```
1.1 1.2 1.3 Bert_Con_ons RAD 46/108
test.py
c=[big_dipper,little_dipper,cepheus, cassiopeia]

def draw_stars(a):
    x=a[0]
    y=a[1]
    for i in range(len(x)):
        set_color(black)
        fill_circle(x[i],y[i],3)
        set_color(white)
        fill_circle(x[i],y[i],2)
```

```
1.1 1.2 1.3 *Dice 1 RAD 18/22
dice.py
n=1
while n!=0:
    try:
        n=int(input("n="+str(n-1)+" , Enter = next to
    except:
        n=n+1
    pass
    if n==0:
        break
    s=randint(1,6)+randint(1,6)
    lyst1[s]+=1
```



# Questions

- Do you use Python today?
- How will you use Python on TI-Nspire CX II with your students?
- If you have any questions later, you can email: [nspirepython@list.ti.com](mailto:nspirepython@list.ti.com)