Research Methods

Robert E Simpson

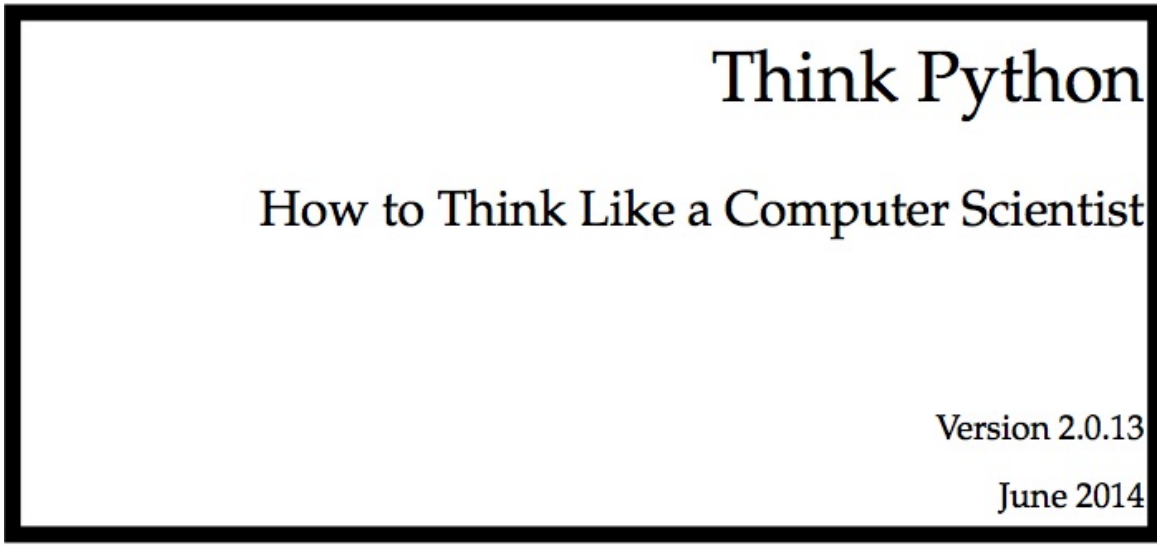
Singapore University of Technology & Design

robert_simpson@sutd.edu.sg

September 12, 2018

Python Book

"Think Python: how to think like a computer scientist" is an excellent book for those who are new to programming and python. The ebook is available to download from the resources section of edimension.



The beginning of this course will closely follow this book.

Class Objectives

The overall objective is to learn python sufficiently to write programs to efficiently perform statistical analyses on datasets. The objective of today's class is to:

1. Learn how to use python in interactive mode
2. Ensure everybody has a working copy of Python 2.7 and the Spyder IDE
3. Learn how to use, variables, strings, and lists
4. Learn how to use basic structures such as for loops and while loops
5. Learn how to write and call functions
6. Learn how to code functions to calculate basic statistical parameters

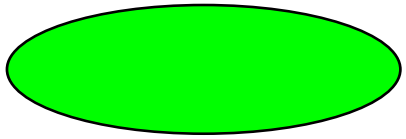
Programs

The basic elements of a program are:

Inputs	Data provided from the keyboard, a file, USB, or from some other device
Outputs	Display data or send data
Process	Perform operations, such as mathematics
Conditions	Check to see whether a condition is satisfied, and if it is satisfied execute some code
Loops	Execute some code repeatedly

A program is a list of these elements that are run systematically.

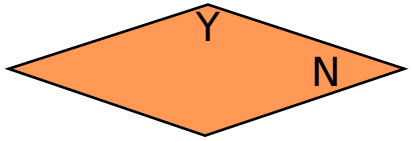
Program Flow Charts



Start/Stop

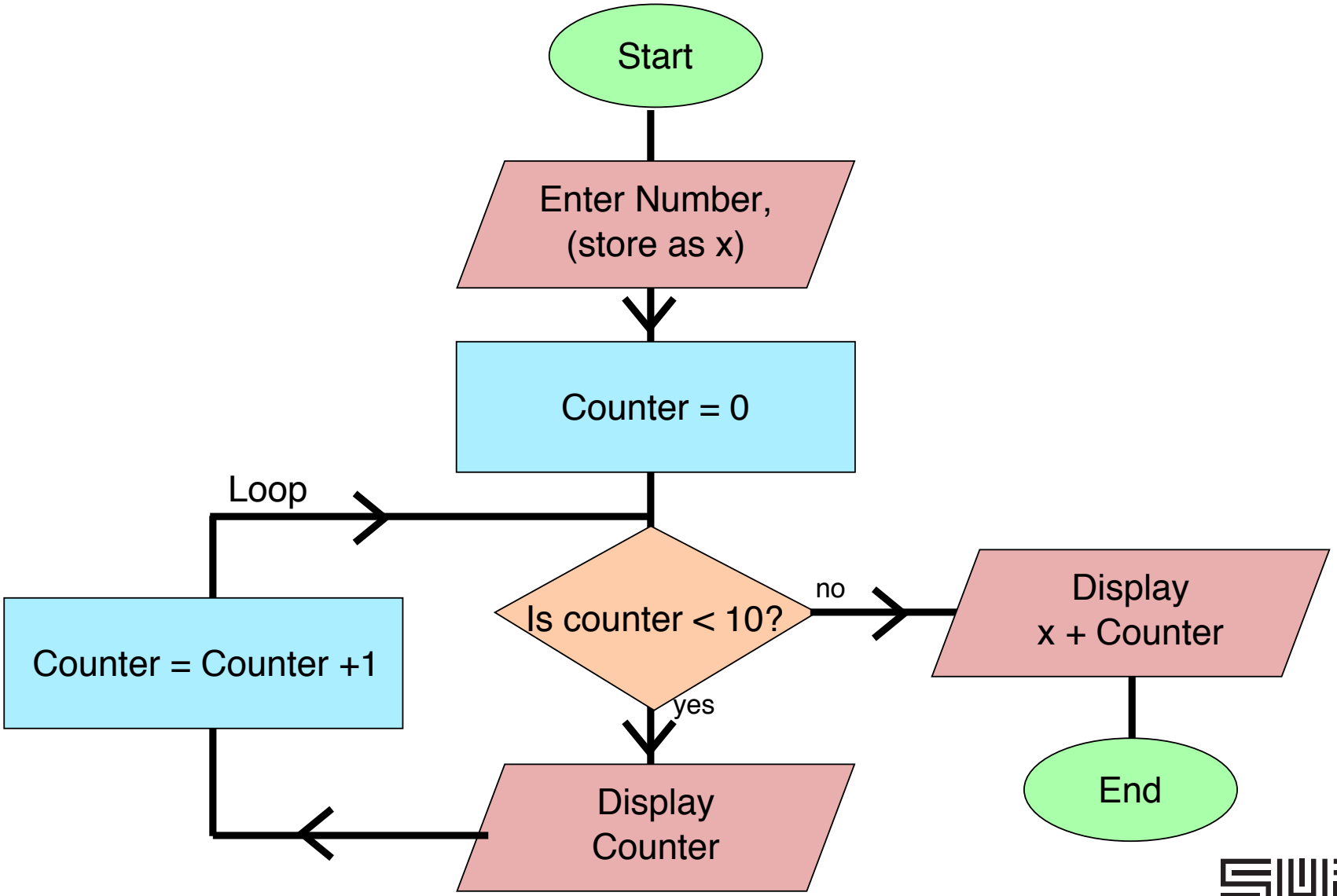
Set Variables
Do a Process

Input/Output



Query

Example of a program flow with loop



Interpreter vs Compilers

Interpreters translate the program code, line by line, into instructions that the computer can understand every time the program. **Python, Matlab, and Igor programs are executed by an interpreter**.

Compilers translate the program *source code* to create the object code or executable, which is a set of instructions that is already translated for the computer. Once the program is compiled into object code, it can be executed without further translation. C#, C++, Fortran codes are compiled.

Interactive vs Script

Interactive You type the programs into a terminal and the interpreter displays the result. These programs are not saved. The interactive mode is useful for testing lines of a longer script– this is an advantage of interpreted languages.

Script The program is stored in a file and the interpreter executes the instructions in the file.

Example 1: Interactive environment

- On a linux/mac machine open a terminal/shell
- Type: `python` –starts the interactive python interpreter. The chevrons, `>>>`, indicate that python is ready for your program.
- Type: `>>> print ('Hello World!')` [It's traditon to learn a new programming language with this]
- Type a sum, such as: `>>> 1 + 1`. The program will run and 2 will be output.
- Now try typing `>>> 1.0 + 1.0`. The program outputs 2.0.
- Try: `>>> 1/3`
- Try: `>>> 1.0/3`
- Try: `>>> 1/3.0`

Why does 1/3 and 1/3.0 give a different answer?

Shortcuts in terminal

control a	moves cursor to beginning of line
control e	moves cursor to end of line
up cursor	moves back through the history of lines entered
down cursor	moves forward through the history of lines entered

Using interactive python as a calculator

- `+` Addition, e.g `10+2`
- `-` Subtraction, e.g `10-2`
- `/` Division, e.g `10/2`
- `*` Multiplication, e.g `10*2`
- `**` Exponentiation, e.g `10**2`

Try it out using the python interpreter

Order of operations

Python follows the **PEMDAS** mathematical convention:

- **P**arentheses have the highest precedence and can be used to force an expression to evaluate in the order you want
- **E**xponentiation has the next highest precedence
- **M**ultiplication and Division have the same precedence. Operators with the same precedence are evaluated from left to right
- **A**ddition and **S**ubtraction have the same precedence and have a lower precedence than mutiplication and division.

Following these rules, what will be the solution to:
(5+5**3/25-25**0.5)**2?
Try out some examples of your own using the python interpreter.

It's chic to be geek

pythoni – run code,autocomplete,outline,color code

View More by This Developer

By XiaoWen Huang

This app is only available on the App Store for iOS devices.

This app is designed for both iPhone and iPad

Offers Apple Watch App for iPhone

Free

Category: Productivity
Updated: Sep 09, 2016
Version: 3.5
Size: 112 MB
Apple Watch: Yes
Languages: English, Simplified Chinese
Seller: XiaoWen Huang
© Huang XiaoWen
You must be at least 17 years old to download this app.
Unrestricted Web Access

Description

This is an ios python2.7 app,you can learn,run,share python2.7 script.
Features :
Autocomplate.

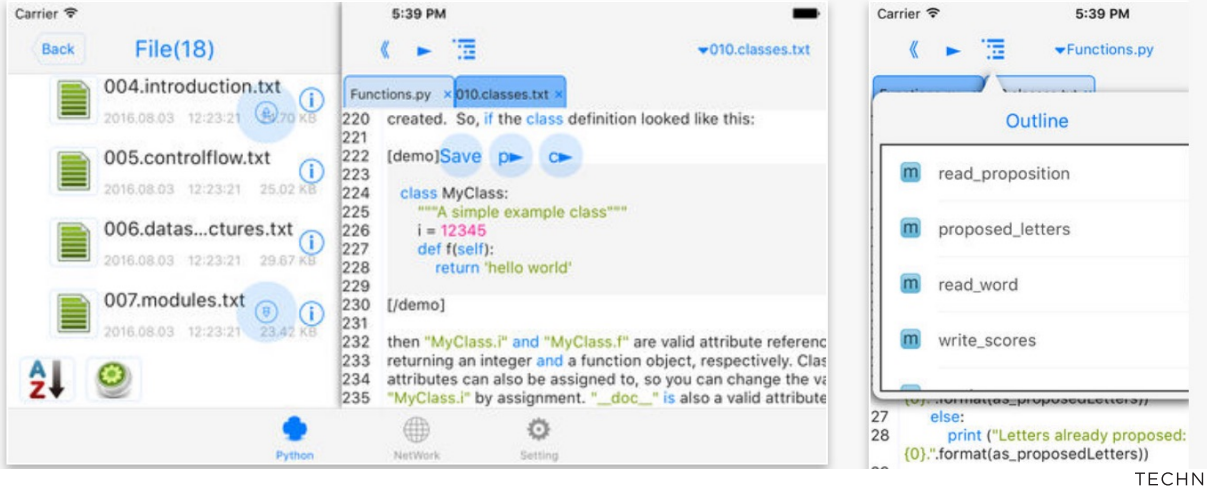
XiaoWen Huang Web Site ▸ pythoni – run code,autocomplete,outline,color code Support ▸ ...More

What's New in Version 3.5

Fix some bugs.

Screenshots

iPhone | iPad | Apple Watch



Types

String A sequence of characters that may be stored in a variable

E.g: `>>> x='Hello World!'`

Variable A variable is a name that refers to a value.

E.g: `>>> x=5`

Strings

Strings in python are set using the apostrophes (') or quotation marks (").
Strings can be stored in a variable, e.g.

```
>>> rob='itś chic to be geek'
>>> print (rob) it's chic to be geek.
```

The backslash (\) is an escape character. It tells python to ignore the spacial meaning of the apostrophe and treat it as a string character.

Strings

You cannot perform matermatical operations with strings, because they store characters and not a number. The string can be number character, but it does not hold the number value.

However, the addition sign (+) can be used to connect two strings, e.g:

```
>>> str1='cats '
>>> str2='dogs '
>>> str3='and '
>>> print (str1 + str3 + str2)
cats and dogs
```

The multiplication sign (*) can also be used to repeat a word: e.g:

```
>>>print str1*10
'cats cats cats cats cats cats cats cats cats cats '
```

Keywords

Python has 31 keywords. These cannot be used as variable names.

Table: Keywords

not	while	or	in
pass	yield	print	for
return	try	and	if
assert	break	class	as
finally	is	def	with
lambda	del	from	raise
elif	global	else	continue
except	import	exec	

Values and types

```
>>> type(‘Hello, World!’)
<type ‘str’>

>>> type(3)
<type ‘int’>

>>> type(3.0)
<type ‘float’>
```

Case problem 1– Practice using the Python interpreter as a calculator:

1. The volume of a sphere with radius r is $V = 4\pi r^3$. What is the volume of a sphere of radius 5m?
2. Suppose the cover price of a book is \$24.95, but bookstores get a 40% discount. Shipping costs \$3 for the first copy and \$0.75 for each additional copy. What is the total wholesale cost for 60 copies?
3. If I leave my house at 6:52 am and run 1 mile at an easy pace (8m15s per mile), then 3 miles at tempo (7m12s per mile) and 1 mile at easy pace again, what time do I get home for breakfast? What was my mean pace?

Python Script

- Python script is simply a text file containing the lines of python code, which the interpreter reads sequentially.
- Usually python script files have the extension `.py`
- On the mac they are run in terminal by typing `python filename.py`, where `filename` is the name of your script.
- The scripts can be written in any text editor, but we will generally use the Spyder IDE, which can recognise python syntax.

Case Problem 2

Write a script to calculate the hypotenuse of a right angled triangle.

Comments and annotations

It is good practice to annotate your code so that others can understand it. Indeed, I need to understand your code when I read your assessed problem sets.
In python you can add comments to your code using the # sign.

Libraries

Libraries of pre-written functions can be loaded by the script.
Libraries are loaded using the following format:

```
import [Library] as [your name for the library]
```

Examples:

```
import numpy as np          #Scientific computing
import matplotlib.pyplot as plt       #Plotting
import scipy as sp          #Science, Maths, Stats The
```

library functions are then used in the following way:

```
>>> np.cos(0)
1
>>> np.cos(np.pi)
-1
```

While Loops

While loops continue running the same segment of code until a condition is met.

For Loops

For loops run a segment of code a set number of times or for a set number of time.

Examples

```
for counter in range(10):
      print (counter)
```

The command `range` means for an iterator less than 10, count from 0 in steps of 1. The iterator is the variable `counter`. Each time the for loop iterates the value of the `counter` is printed to the display.

```
for counter in range(1, -10, -2):
      print (counter)
      print ('I love Research Methods')
```

The range command is now stated with a start number (1), the end number which breaks the loop (-10), and the step size (-2). Each time the loop will print the counter value and the tell you how much you love research methods!

Creating a list with loops

To write a value to a list during each iteration of the list we use `list_name.append`. First we need to tell Python that we want to create a list. This is done using `list_name[]`. For example:

```
my_list=[]
>>>for counter in range(1, -6, -2):
       my_list.append(counter)
>>>print (my_list)
[ 1, -1, -3, -5]

>>> print (my_list[2])
-3
```

The number in the square bracket is the line number of the list. The list line numbers start at 0.

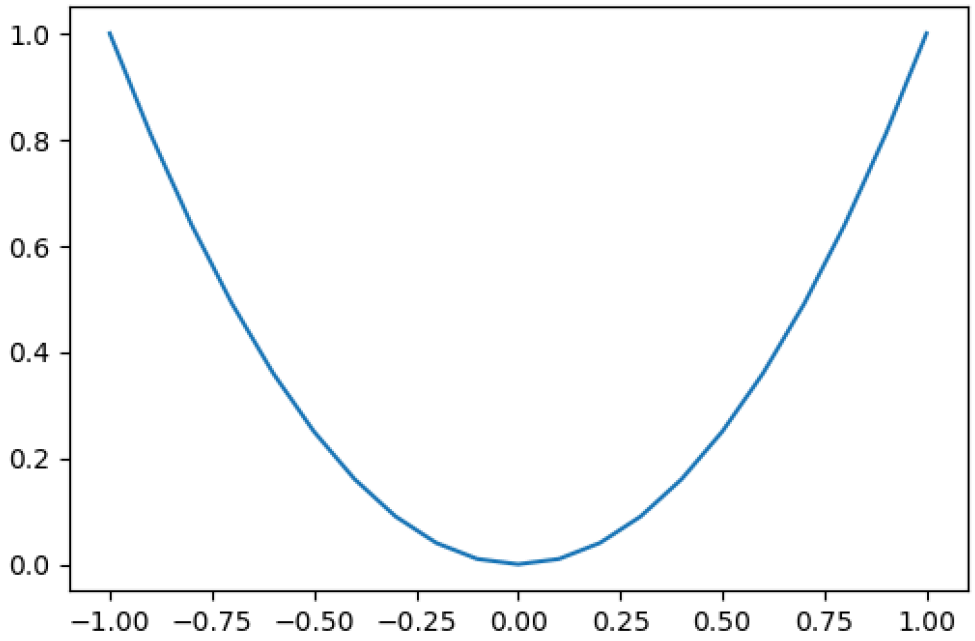
Creating a list of numbers

List of number with a regular step size can be created without using the numpy `linespace` command. It has the format `linespace(start value, final value, number of divisions)`. For example:

```
 >>> x=np.linspace(-1, 1, 21)
>>>print (x)
[-1.  -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 0.  0.1 0.2
0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.  ]
```

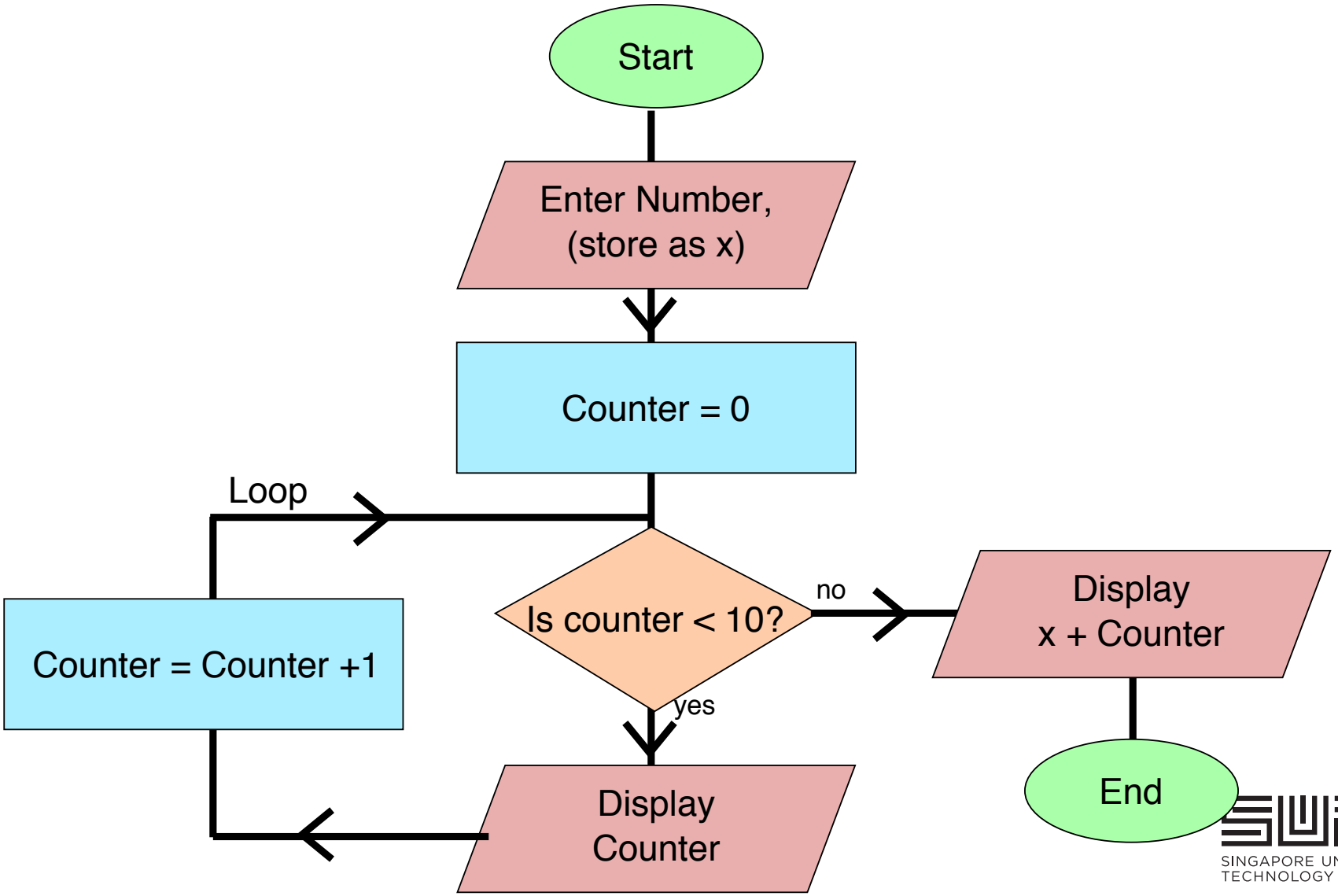
Plotting lists of variables

```
 >>> import matplotlib.pyplot as plt
>>> x=np.linspace(-1, 1, 21)
>>> y=x**2
>>> print y
[ 1.  0.81 0.64 0.49 0.36 0.25 0.16 0.09 0.04 0.01 0.  0.01 0.04
0.09 0.16 0.25 0.36 0.49 0.64 0.81 1.  ]
>>> plt.plot(x,y)
```



Case problem 3

Write a script to run the algorithm shown below:



Case problem 3 solution

```
print "Enter␣a␣number"
x=input()
for counter in range(10):
    print counter
print counter+x
```

Case Problem 4

- Make a list called `my_x` that goes from -4π to $+4\pi$
- Compute $sin(x)$ of the values in the list my_x [hint: lookup up `np.sin`]
- Add Labels [hint: lookup pyplot xlabel]
- Change the plot style [hint: lookup pyplot linewidth=2, color='r']

Case Problem 4 Solution

```
#import the numpy library
import numpy as np

#create 100 x values
my_x=np.linspace(-4*np.pi,4*np.pi,100)

#  list of y values
y=np.sin(my_x)

# plot x vs y
plt.plot(my_x,y, linewidth=1, color='r', marker='x')
```

Functions

A function is a named sequence of statements that performs a computation. They are useful because often you want to run the same sequence of statements at different parts of the program. Thus if you define a function, you can simply **call** it when needed. This makes the program simpler and easier to debug.

Defining a new function

Functions are defined using:
`def function_name(input1, input2, ...):`
The inputs are variables, strings, lists, or arrays that are used by the function. They are *local variables* that only are stored when the function is called.

As an example:

```
>>>def squareroot(in_num):
      sqrt=in_num**0.5
      return(sqrt)

>>> print squareroot(9)
3
```

Case Problem 5

Write a function to compute the hypotenuse of a right-angled triangle with sides of length a and b. Run the code with a=3, and b=4.

Case Problem 5 Solution

```
def hypotenuse(a,b):
    print "the␣hypotenuse␣is"
    c=(a**2 +b**2)**0.5
    return c

print hypotenuse(3,4)
```

Summary

- Discussed the basic elements of a program
- Shown how algroithms can be depicted using flow diagrams
- Learnt how to use the python interactive environment
- Learnt about data types
- Learnt the basic plotting function
- Learnt how to write a basic function