❖ An Introduction to Scientific computing using Python
  ➢ Python Fundamentals
    Ref: MITx: 6.00.1x Introduction to Computer Science and Programming Using Python

❖ A program:
 a sequence of definitions and commands.

✚ Data Objects:

Each data object has a type that defines the kind of things the programmes can do to it.

❖ Objects:
  ➢ Scalar: cannot be subdivided.
  ➢ Non-Scalar: can be subdivided

✚ Scalar objects:
  ✓ Int: represent integers
  ✓ float: represent real numbers
  ✓ bool: represent boolean values

❖ Using Shell
  ✓ You can type: 5, 5.0 or True

✚ type command

❖ Simple calculations with ints and floats
➢ 5+8, 5.0+4, 5*4
➢ 5/4, 5.0/4
➢ (5+4)*2
➢ 2+3*4

  ✓ Precedence: In the absence of parentheses, execution is made from left to right first using ** then * and/ and then + and -

✚ Comparing ints and floats:
  ✓ i>j, i<j,i>=j, i<=j, i==j, i!=j
  ✓ 3>5, 3<5, 3>3, 3>=3, 4==4, 5!=5

✚ Operations on bools
  ✓ a and b
  ✓ a or b
  ✓ not a or not b
  ✓

- Type Casting (Type Conversion)
- ✓ float(3), int(4.9)

- ❖ Variables: simple means of abstraction
- ✓ Variable Assignment:  " = "
  - ➤ L=0.3
  - ➤ A=15
  - ➤ k=0.9
  - ➤ R=L/(k*A)

what if:

  - ➤ k=10
  - ➤ R=?

- Non-Scalar Objects:
- ❖ Strings (str): sequence of charcters
- ❖ either a single ('  ')or double quotes(" ")
  - ➤ name1="abc"
  - ➤ type(name1)
  - ➤ type("123")
  - ➤
- ❖ String Operations:
  - ➤ 3*"a"
  - ➤ 3*"ab"
  - ➤ ab+str(12)
  - ➤ len("name")

- ❖ Extracting parts of strings
- ➤ indexing
- ➤ "res"[1] : Starts from 0
- ➤ "res"[3]:?

- ✓ Going backwards:
- ➤ "res"[-1]

- ✓ Slicing
- ➤ "res"[a:b]   ----> starts at a  and <u>stops just before b</u>
- ➤ "resistance"[1:3]

❖ Using IDLE Shell is not desirable
   ✓ we need to save our code in a separate script

   ✓ Print Command
   ➢ print("Resistance")
   ➢ print(3*"R")
   ➢ print(R)
   ➢ print("The resistance is determined to be " +str(R)+ "degC/W")


   ✓ raw_input Command
   ➢ input_k=raw_input("Please Enter the layer's conductivity")
   ➢ print(input_k)
   ➢ print("You just told me that the conductivity is "+k+ " W/(m.degC)")
   ➢ k=float(input_k)
   ➢ print("The resistance is determined to be " +str(R)+ "degC/W")


✚ A simple script:

➢ L=raw_input("Please Enter the length of the layer (in m): ")
➢ A=raw_input("Please Enter the area of the wall (in m2): ")
➢ k=raw_input("Please Enter the condictivty of the layer (in W/(m*K)): ")
➢ print("\n you just said "+ "L= " + L+ " m "+ "A= " + A+ " m2 "+ "k= "+ k +" W/(m*K) \n")
➢ R=float(L)/(float(k)*float(A))
➢ print("Well the Thermal Resistnace is "+ str(R)+ " degC/W")

❖ Comments:
   ✓ starting with #

➢ Till now we just executed commands line by line!


✚ Branching programs:
❖ Conditioning
   ✓ evaluating  a boolean :
   ✓ a script executing if the boolean is True
   ✓ (optional) a script executing if the boolean is False

❖ A not so intelligent way of extracting material properties!:

➤ L=raw_input("Please Enter the length of the layer (in m): ")
➤ A=raw_input("Please Enter the area of the wall (in m2): ")
➤ material=raw_input("Please Enter the material of the layer: ")
➤ if  material=="glass":
➤        k="1.145" #W/(mK)
➤ else:
➤        print("I do not have the properties of this material")
➤        k=(raw_input("Please Enter the conductivity off the layer(in W/(m K): "))
➤ print("\n you just said "+ "L= " + L+ " m "+ "A= " + A+ " m2 "+ "k= "+ k +" W/(m*K) \n")
➤ R=float(L)/(float(k)*float(A))
➤ print("Well the Thermal Resistnace is "+ str(R)+ " degC/W")


❖ Nested if

➤ L=raw_input("Please Enter the length of the layer (in m): ")
➤ A=raw_input("Please Enter the area of the wall (in m2): ")
➤ material=raw_input("Please Enter the material of the layer: ")
➤ if material=="glass":
➤        type_glas=raw_input("which type of glass do you mean:window=1, wool          insulation=2 ")
➤        if int(type_glas)==1:
➤                k=str(0.96) #W/mK
➤        else:
➤        k= str(0.04) #W/mK
➤ elif material=="brick":
➤        k=str(0.8) #W/mK
➤ else:
➤        print("I do not have the properties of this material")
➤        k=(raw_input("Please Enter the conductivity off the layer(in W/(m K): "))
➤ print("\n you just said "+ "L= " + L+ " m "+ "A= " + A+ " m2 "+ "k= "+ k +" W/(m*K) \n")
➤ R=float(L)/(float(k)*float(A))
➤ print("Well the Thermal Resistnace is "+ str(R)+ " degC/W")


🞢 Still Each statement gets executed just once!

- Loops and iterations

- x=4
- itersLeft=x
- Sum=0
- while(itersLeft!=0):
-    Sum=Sum+x
-    itersLeft=itersLeft-1
-    print(Sum)
-    print(itersLeft)
- print(str(x)+"**"+" 2 = "+str(Sum))

- The While loop example with Resistances

- For Loop
- In order to iterate over a sequence of choices
- for <identifier> in <sequence>:
  - o   <code book>
- Break: stops the iteration process

- ✓ Generating a sequence of integers:
- ✓ range(n)
- ✓ range(n,m)

Function in python

def <function name> (function parameters):

<function body>

- function Maximum

The idea of Environment:

formalism for the binding of values and environments

- ✓

- ✓ Pay attentions that the variables inside the function are always local
- ✓ localvariables function


- ⬦ Modules:
- ➤ import module
- ➤ module.function structure


- ➤ from module import *
- ➤ from module import function


- ⬦ Compound data types:
- ➤ tuples
- ➤ lists
- ➤ Dictionaries


- ⬦ Tuples:
  t1=(1,"yes",2)
  t2=(t1,"si")
  t3=t1+t2
  t1[3]

  t4=(5,)

- ⬦ Lists
- ⬦ L1=[2,3,4,5]
  L1.append(8)
  L2=[3,5,6,7,8]
  L2.remove(5)


- ⬦ Dictionaries
- ⬦ layer1=["area":15, "length":0.3, "conductivity":0.9]
- ⬦ layer1["area"]