NNSA HSAP Computational Laboratory 3 - 2026 Update

Goals: Python scripts; More python syntax (loops, etc)

[1] It's awkward to use python in interactive mode unless you want to do something really simple (ie. just a few lines). For longer tasks one write a python 'script'. This is just a file containing the same python commands you would have typed in interactively. Then you import the script into python. This of course has the **big** advantage that you can make changes to, or re-use, the script without typing it all over again.

The computers in this lab use the linux 'operating system' (OS). Does anyone know what an OS is? Do you know other OS examples? Over the next few weeks we will teach you about linux. For now just be aware that it is an incredibly useful OS to know. Most sophisticated computer clusters use a linux or related OS.

In linux, there are a lot of 'editors' you can use to write stuff Max and Alexandria will explain the one we will use: gedit.

[2] We'll now just write a bunch of scripts to practice editing, and to learn more about how python does things.

Use an editor to create a file named 'add.py' containing:

```
x = float(input("Please enter a number: "))
y = float(input("Please enter a second number: "))
z=x+y
print("The sum of the two numbers is:")
print(z)
```

Then (make sure you are in the same directory/folder where your file is) type:
python ./add.py
You will be queried for two numbers, and then python will tell you their sum. Try it for a few different inputs.

It's pretty obvious, but let's 'walk through the code' and discuss what it is doing line-by-line.

[EX: 3-1] Use an editor to create a file named 'subtract.py' which reads in two numbers and subtracts them.
To run it, type:
python ./subtract.py

[EX: 3-2] Use an editor to create a file named 'multiply.py' which reads in *three* numbers and multiplies them. To run it, type:
python ./multiply.py

Let's discuss a few useful linux commands to make your workflow easier and more transparent:
ls, cp, rm, mv

Next, use an editor to create a file named 'ifelse.py' containing:

```
n = int(input("Please enter an integer: "))
if n%2 ==0:
    print('n is even')
else:
    print('n is odd')
```

To run it, type:
python ./ifelse.py
You will be queried for a value for the integer, and then python will tell you whether the integer is odd or even. Try it for a few different inputs.

It's time to learn about loops!

Write this python script (you might call it 'arithmeticsum.py'):

```
N = int(input("Please enter an integer: "))
sum = 0
for i in range (1,N+1):
    sum = sum+i
    print(sum)
```

Then execute it with python:
python ./arithmeticsum.py
What does it do?
Run your script for N=30. What do you get?
Run your script for N=1000. What do you get?

This code is a first example of a 'loop' in python. Loops are among the most powerful of all language elements. Let's 'walk through the code' and discuss what it is doing line-by-line.

Important comment: Make especially sure you understand the line

```
    sum = sum+i
```

Actually, there is a trick to sum arithmetic series withou a computer. Have you learned it?

Discuss different methods.

[EX: 3-3] Write a python script to sum the squares of the first N integers:

$$1^2 + 2^2 + 3^2 + 4^2 + \cdots N^2$$

Challenge question: Is there a magic formula for this, similar to the summing of the first N integers without squaribng?!?!

[EX: 3-4] Now write this python script (you might call it 'fact.py')

```
N = int(input("Please enter an integer: "))
fact=1
for i in range (1,N+1):
    fact=fact*i
    print(fact)
```

Then execute it with python:
python ./fact.py
What does it do if you run your script for $N = 10$?
What does it do if you run your script for $N = 30$?
Why did we call this 'fact.py'?
It turns out the programming language C would give the wrong answer for $N = 30$ unless you do something special. Can anyone guess why?

Discuss base two and how computers store numbers!

Change your script to this alternate. The only difference is the indentation in the last line.

```
N = int(input("Please enter an integer: "))
fact=1
for i in range (1,N+1):
    fact=fact*i
print(fact)
```

What happens? Explain.

Change your script to this third version. Again, the only differences are indentations.

```
N = int(input("Please enter an integer: "))
fact=1
for i in range (1,N+1):
fact=fact*i
print(fact)
```

What happens?

[3.] Type in this python script. What does it compute?

```
N=input('Enter N:  ')
M=input('Enter M:  ')
Nfact=1
for i in range (1,N+1):
    Nfact=Nfact*i
Mfact=1
for i in range (1,M+1):
    Mfact=Mfact*i
NminusMfact=1
for i in range (1,N-M+1):
    NminusMfact=NminusMfact*i
result=Nfact/(Mfact*NminusMfact)
print('what is this mystery quantity?!  ')
print(result)
```

[4.] Type in this python script. What does it compute?

```
A=input('Enter A:  ')
x=1.
for i in range (1,10):
    x=x/2.+A/(2.*x)
    print(x)
```

[5.] Type in this python script. What does it compute?

```
A=input('Enter A:  ')
x=1.
for i in range (1,10):
    x=2.*x/3.+A/(3.*x*x)
    print(x)
```

[6.] This script computes the 'Fibonacci numbers'. Type it in and run it.

```
N=input('Enter N  ')
fib1=1
fib2=1
for i in range (1,N):
    fib3=fib1+fib2
    print(fib3)
    fib1=fib2
    fib2=fib3
```

[EX: 3-5] Clearly explain the logic of the script in [6.]. What are the last two lines doing and why are they needed? (This is tricky!)

The Fibonacci sequence appears in the book *Liber Abaci* (1202) by Fibonacci. Fibonacci considered the growth of an idealized (biologically unrealistic) rabbit population, assuming that: a newly born pair of rabbits, one male, one female, are put in a field; rabbits are able to mate at the age of one month so that at the end of its second month a female can produce another pair of rabbits; rabbits never die and a mating pair always produces one new pair (one male, one female) every month from the second month on. The puzzle that Fibonacci posed was: how many pairs will there be in one year?

[EX: 3-6] Explain why your script is answering Fibonacci's puzzle.

According to wikipedia: Fibonacci numbers appear unexpectedly often in mathematics, so much so that there is an entire journal dedicated to their study, the *Fibonacci Quarterly*. Applications of Fibonacci numbers include computer algorithms such as the Fibonacci search technique and the Fibonacci heap data structure, and graphs called Fibonacci cubes used for interconnecting parallel and distributed systems. They also appear in biological settings, such as branching in trees, the arrangement of leaves on a stem, the fruit sprouts of a pineapple, the flowering of an artichoke, an uncurling fern and the arrangement of a pine cone's bracts.

Although named after an Italian mathematician, the sequence had been described earlier in Indian mathematics. More from wikipedia: The Fibonacci sequence appears in Indian mathematics in connection with Sanskrit prosody $\cdots$ In the Sanskrit poetic tradition, there was interest in enumerating all patterns of long (L) syllables of 2 units duration, juxtaposed with short (S) syllables of 1 unit duration. Counting the different patterns of successive L and S with a given total duration results in the Fibonacci numbers: the number of patterns of duration m units is F_{m+1}.

Knowledge of the Fibonacci sequence was also expressed as early as Pingala (c. 450 BC-200 BC). Singh cites Pingala's cryptic formula misrau cha ("the two are mixed") and scholars who interpret it in context as saying that the number of patterns for m beats (F_{m+1}) is obtained by adding one (S) to the F_m cases and one (L) to the F_{m-1} cases.

[1] Use gedit to type in the following python script. It helps us practice with 'logical statements'. Notice that in python you can comment your code using the # symbol at the start of a line. Comments are always a good idea in coding!

```
# Read two numbers from the user
a = float(input("Enter the first number: "))
b = float(input("Enter the second number: "))
# Compare them
if a > b:
    print("The first number is larger.")
elif b > a:
    print("The second number is larger.")
else:
    print("The two numbers are equal.")
```

Run it and make sure it works. **Warning:** The indentations are important!

[2] The function

$$y = a\,x^2 + b\,x + c, \tag{1}$$

is called a parabola. Let's draw pictures and do some exambles to see what's going on.

- $y = x^2 - 4$
- $y = x^2 - 25$.
- $y = -x^2 + 16$.

In these examples the middle term $b = 0$. What do you notice about the shape of y? What does the sign in front of the x^2 control? What about

- $y = 3x^2 - 4$

What do you think the value of a controls?
Can you guess what b might do?
Let's try something with a b term present: • $y = x^2 - 2\,x - 4$

The quadratic formula tells you the values of x where $y = 0$

$$x = \frac{-b \pm \sqrt{b^2 - 4\,a\,c}}{2\,a} \tag{2}$$

Geometrically, $y = 0$ means the curve intersects the x axis. This turns out to be a very important thing to know!

This python script implements the quadratic formula. Please type it in.

```
# First code to solve the quadratic equation.
# To use the sqrt function we need to import a library

import math

# Read in a,b,c

a = float(input("Enter a: "))
b = float(input("Enter b: "))
c = float(input("Enter c: "))

# Get the first root
root1 = ( -b + math.sqrt(b**2 - 4.0*a*c) ) / (2.0*a)
print("The first root is",root1)

# Get the second root
root2 = ( -b - math.sqrt(b**2 - 4.0*a*c) ) / (2.0*a)
print("The second root is",root2)
```

Test your code for
- $y = 2\,x^2 - 11\,x - 21$

Do you know another way to figure out when $y = 0$ besides the quadratic formula?

In all the examples we did, the parabolas intersected the x axis. Does this *always* happen?

Let's draw $(a = 1, b = 0, c = 9)$
- $y = x^2 + 9$

Isn't this a contradiction? We decided some parabolas don't intersect the x axis but the quadratic formula always gives you two values!

Run your code for this problem. What happens?

So your code *knows* something is going wrong. Can you see what is happening in the quadratic formula?

[EX: 3-7] Write a second version of your quadratic formula code which checks to see if $b^2 - 4\,a\,c$ is negative and prints a warning instead of trying to take the square root of a negative number.

[EX: 3-8] For the experts! What happens if $b^2 - 4\,a\,c = 0$? Are there still roots? See if you can write a third quadratic equation code which deals with this situation correctly.

[3] You are told that a line goes through the point $(3, 5)$. Write a code which prints the "y-intercept" (the place the line intersects the y-axis) if you enter the slope of the line m.

It's always good to draw some pictures before trying to do math. We'll do some examples at the board (with pictures), e.g. $m = 0$ and $m = 1$ and $m = -1$.

Now let's remember the math. The equation of a line is

$$y = mx + b \tag{3}$$

where m is the slope (which the user of your code will enter) and b is the y-intercept, which is what your code will figure out. The extra information is that the point (3,5) is on the line, which means it obeys Eq. 3.

$$5 = 3m + b \tag{4}$$

You need to solve this equation for b (remember you have entered what m is) and have the code print out b.

[EX: 3-9] Write the python script for this!

So far we're thinking about where the line hits the y axis. What about hitting the x axis?

As before, it's good to draw some pictures before trying to do math. We'll do some examples at the board (with pictures), e.g. and $m = 1$ and $m = -1$. What happens if $m = 0$?

Now do the math. A line intesects the x axis when $y = 0$. Using

$$y = mx + b \tag{5}$$

and putting in $y = 0$ gives

$$0 = mx + b \tag{6}$$

You know m (the user inputs the value) and your first code computed b for you. If you solve Eq. 5 for x then that's the value you need where you hit the x axis.