NNSA HSAP Computational Laboratory 4

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

$$y = mx + b \tag{1}$$

and putting in $y = 0$ gives

$$0 = mx + b \tag{2}$$

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.

[EX: 4-1] Make a second version of your script which also figures out the place the line intersects the x axis.

[EX: 4-2] For the experts! When we drew pictures we noticed that $m = 0$ was a special case where the line never hit the x-axis. There was no solution. Will your code be able to figure that out? Do you have ideas (perhaps with using an 'if' statement) to make your code really robust?

[1] Recall the script to do the 'arithmetic sum'

$$\text{sum} = 1 + 2 + 3 + 4 + \cdots N$$

which figures out the sum of the first N integers:

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

is telling the computer to do. Let's review the whole script line-by-line. A key idea to master in this code was what

```
    sum = sum+i
```

is telling the computer to do.
What would happen if your code was written like this:

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

[2] Let's do another problem with sums. This one is called a 'geometric sum'

$$\text{sum} = x + x^2 + x^3 + \cdots x^N$$

Let's think about the steps the script *geometricsum.py* will need to do:
- Set sum=0 to start.
- Need to input the value of x.
- Need a loop.

$\Rightarrow$ What's inside the loop?!
- Need to write the answer at the end.

[EX: 4-3] Write the script!

[3] What do you get when you run your code for $x = 1/2$ (that is, $x = 0.5$) and $N = 4$?
What if $N = 10$?
What if $N = 20$?
What if $N = 50$?
What if $N = 100$?
What if $N = 200$?
What do you think *sum* is if $N = \infty$?
The result you get is the solution to "Zeno's paradox".

[EX: 4-4] What do you get when you run your code for $x = 0.1$ and $N = 4$?
Is there anything funny about your answer?
What should the compute have said the answer is?
All computers store numbers using a finite number of bits.
This leads to "round-off" errors. More on this later!!
What do you get for $x = 0.1$ and $N = 10$?
What do you get for $x = 0.1$ and $N = 50$?

[4] There is actually a much more simple way of writing

```
0.11111111111 ...
```

Does anyone know what it is?

[EX: 4-5] Use python in 'interactive mode' to compute 1/9.