“My CS 5 Green Lecture Notes Were Stolen By Aliens” claims distraught CS 5 Professor.

Claremont, CA: A Harvey Mudd CS Professor claims that his lecture notes were stolen by aliens early on Thursday morning. “I wrote up some notes for my lecture and, while I went out to get a cup of coffee before class, aliens snuck into my office and stole my notes to eat for breakfast,” claimed the distraught professor. “We’re obligated to investigate every claim,” said a campus security officer, “but we’re skeptical of this one. Aliens are not known to eat class notes for breakfast. They much prefer Spam-flavored Pop Tarts and Red Bull. We’re investigating, but we suspect that squirrels were the actual perpetrators of this terrible crime.” Why are you reading this drivel? Don’t you have anything better to do?
The Secret of Happiness is…
(in programming)

• Don’t memorize!
• Look at examples of similar problems
• Experiment
• Syntax that looks weird now will become second nature soon
Looking up stuff on the web...

Looking up Python documentation (e.g., Googling “python documentation”) is fine!
Types of data in Python

```python
>>> goodNum = 42  # Integer
```

```python
>>> pi = 3.1415926  # “Floating point” number
```

```python
>>> special = [2.718, 3.141, 42]  # List
```

```python
>>> bestFood = "spam"  # String (single or double quotes work)
```

```python
>>> okFood = 'chocolate'
```

```python
>>> 42 = goodNum  # Python barf?
```

```python
>>> 'chocolate' = okFood
```
>>> 3 == 1+2
True

>>> 1+2 == 3
True

>>> 42 == "spam"
False

>>> [1, 2, 3] == [1, 2, 3]
True

>>> [1, 2, 3] == [3, 2, 1]
False

>>> 42 > 5
True

>>> 42 != 5
True

>>> not 42 == 5
True

Notice the ==
if, else...

def special(x):
    """This function demonstrates the use of if and else""
    if x == 42:
        return "Nice!"
    else:
        return "Yuck!"

>>> special(42)
"Nice!"

>>> special(43)
"Yuck!"
The Collatz Function

```python
def collatz(n):
    """Returns n/2 if n is even and returns 3n+1 otherwise"""
    if n % 2 == 0:  # if n is even...
        return n/2
    else:
        return 3*n + 1
```

The docstring

A comment
def special(x):
    """This function demonstrates the use of if, elif, and else"""
    if x < 42:
        return "Silly little number!"
    elif x == 42:
        return "Secret to all happiness!"
    else:
        return "Wow, that’s big!"
def special(x):
    """This function demonstrates the use of if, elif, and else"""
    if x < 42:
        if x % 2 == 0:
            return "Silly small even number"
        else:
            return "Silly small odd number"
    elif x == 42:
        return "Secret to all happiness!"
    else:
        if x % 2 == 0:
            return "Big & even"
        else:
            return "Big & odd"
Getting things in order…

def schedule(hour):
    """ Describes your day in military time"""
    if hour >= 2 and hour <= 9:
        return "Sleep"
    elif hour <= 17:
        return "In class"
    elif hour <= 20:
        return "Hang out"
    else:
        return "Do CS homework"

You really otta get more sleep!
Getting things in order…

def schedule(hour):
    """ Describes your day in military time"""
    if hour >= 2 and hour <= 9:
        return "Sleep"
    elif hour <= 17:
        return "In class"
    elif hour <= 20:
        return "Hang out"
    else:
        return "Do CS homework"
A Cheesy Example…

```python
def special(x):
    if x == 42:
        dinner = "spam!"
    else:
        dinner = "mac & cheese"
    return dinner
```

Alternatively??

```python
def special2(x):
    if x == 42:
        dinner = "spam"
    dinner = "mac & cheese"
    return dinner
```

```python
def special3(x):
    if x == 42:
        dinner = "spam"
    return dinner
```

return is strong stuff! When Python sees it, the function is done!

Worksheet: Which of these two alternatives does the same thing as the `special` function above?
Python Gets Loopy!

The Python,
Busch Gardens Florida
Python Gets Loopy!

loopy is a function, but it doesn’t expect any input!

We choose the name of a variable…

def loopy():
    sum = 0
    for myNumber in [20, 17, 5]:
        sum = sum + myNumber
    return sum

>>> loopy()
42
Carl Friedrich Gauss
1777-1855
If Gauss had Python...

**Goal:** Add up $1 + 2 + \ldots + n$

```python
def gauss(n):
    ''' returns $1 + 2 + \ldots + n$ '''
    sum = 0
    for num in list(range(1, n+1)):
        sum = sum + num
    return sum
```

This is the list $[1, 2, 3, \ldots n]$
The Anatomy of a `for` loop

```python
for myVariable in list:
    Do all the stuff that's
    Indented beneath the for loop
```

Stuff at this level of indentation
is done afterwards!
Factorial...

Goal: Take an integer n as input and return n!

def factorial(n):
    ''' Returns the factorial of n '''

Try this (and the next two problems) in your notes.
Goal: Take an integer \( n \) as input and return \( n! \)

```python
def factorial(n):
    ''' Returns the factorial of n '''
    product = 1
    for factor in range(1, n+1):
        product = product * factor
    return product
```

In your notes
Approximating e…

Goal: Take a value $n$ as input and return $1 + 1/1! + 1/2! + \ldots + 1/n!$

```python
def e(n):
    '''Returns $1 + 1/1! + 1/2! + \ldots + 1/n!$
    Assume that factorial(n) is available '''
```

In your notes
Approximating e…

Goal: Take a value n as input and return
1 + 1/1! + 1/2! + ... + 1/n!

def e(n):
    '''Returns 1 + 1/1! + 1/2! + ... 1/n!
    Assume that factorial(n) is available '''
    sum = 1  # Why 1?
    for denominator in range(1, n+1):
        sum = sum + 1/factorial(denominator)
    return sum
A mystery…

def mystery(n):
    for d in range(2, n):
        if n % d == 0:
            return False
    return True

What is this function saying about its input n?
A perfect worksheet problem!

\[6 = 1 + 2 + 3\quad 28 = 1 + 2 + 4 + 7 + 14\]

\[6, 28, 496,\]

Known to ancient Greeks
A perfect worksheet problem!

6 = 1 + 2 + 3

28 = 1 + 2 + 4 + 7 + 14

6, 28, 496, 8128,

Known to ancient Greeks Nicomachus, 100 AD!
A perfect worksheet problem!

$$6 = 1 + 2 + 3$$  $$28 = 1 + 2 + 4 + 7 + 14$$

6, 28, 496, 8128, 33,550,336

Known to ancient Greeks  Nicomachus, 100 AD!  1456!
A perfect worksheet problem!

\[ 6 = 1 + 2 + 3 \quad 28 = 1 + 2 + 4 + 7 + 14 \]

6, 28, 496, 8128, 33,550,336

Known to ancient Greeks Nicomachus, 100 AD! 1456!

8,589,869,056 137,438,691,328

Pietro Cataldi, 1588!
>>> perfect(6)
True

>>> perfect(7)
False

def perfect(n):
    sod = 0
    for d in range(1, n):
        if n % d == 0:
            sod = sod + d

What’s this doing? A few more lines of code in your notes and we’re done!

Achieving perfection!
>>> perfect(6)
True

>>> perfect(7)
False

def perfect(n):
    sod = 0
    for d in range(1, n):
        if n % d == 0:
            sod = sod + d
    if n == sod:
        return True
    else:
        return False
>>> perfect(6)
True

>>> perfect(7)
False

def perfect(n):
    sod = 0
    for d in range(1, n):  # [1,2,3,...n-1]
        if n % d == 0:
            sod = sod + d
        if n == sod:
            return True
    return False
>>> perfect(6)
True

>>> perfect(7)
False

def perfect(n):
    sod = 0
    for d in range(1, n):
        if n % d == 0:
            sod = sod + d
    return n == sod

What!?
I think that the prof is delusional!

Demo perfect.py