**The CS 5 Times**

"Zach and Geoff to be Replaced by Two Penguins," says HMC Dean

Claremont (AP): Beginning next week, CS 5 will be taught by a pair of penguins, announced an HMC dean. "Penguins are very smart," said the dean, "and they are also quite adorable, which is more than we can say about the CS 5 faculty. Plus, we won’t need to heat their offices in the winter."

Photos of the two new CS 5 penguin professors

**This Week**

Homework 2:
- Reading on computer interfaces
- Lab: Fractal art
- Problem 2: Higher-order functions
- Problem 3: 42andme
- Problem 4: RNA folding

**Comparing DNA via Longest Common Subsequence (LCS)**

```
AGGACAT
ATTACGAT
>>> LCS("AGGACAT", "ATTACGAT")
5
>>> LCS("can", "man!")
2
```

Please e-mail the prof as soon as you feel bad!

**Speaking of Illness…**

**Arts:** New band “The Recursions” performs songs from first album.

**Sports:** HMC Web surfing team advances to Nationals
Recursive Approach…

def LCS(s1, s2):
    if BASE CASE:
        ???
    else:
        LCS("spam", "sam!")

Turtle Graphics

Turtle graphics are built into Python!

>>> import turtle
>>> turtle.forward(50)
>>> turtle.right(90)
>>> turtle.backward(50)

Problem 2 has a link to the turtle documentation
Fractals

“I Wonder About Trees” – Robert Frost

We wonder about Robert Frost - Trees

>>> svTree(128, 6)

100 long!

svTree(100, 3)

>>> svTree(100, 3)

recursion level

trunk length

>>> svTree(100, 3)

100 long!

svTree(25, 1)

svTree(50, 2)
Tuples ("Immutable Lists")

```python
>>> foo = (42, 'hello', (5, 'spam'), 'penguin')
>>> foo
(42, 'hello', (5, 'spam'), 'penguin')
>>> foo[0]
42
>>> foo[-1]
'penguin'
>>> foo[0:2]
(42, 'hello')
>>> foo[0:1]
(42,)
```

Dictionaries

```python
>>> D = {}
>>> D["Geoff"] = "spam"
>>> D["Zach"] = "donuts"
>>> D["Alien"] = 42
>>> D["Geoff"]
'spam'
>>> D["Alien"]
42
>>> D["Suicide Squad"]
BARF!
```

Tuples ("Immutable Lists")

```python
>>> foo = (42, 'hello', (5, 'spam'), 'penguin')
>>> foo
(42, 'hello', (5, 'spam'), 'penguin')
>>> foo[0]
42
>>> foo[-1]
'penguin'
>>> foo[0:2]
(42, 'hello')
>>> foo[0:1]
(42,)
>>> foo[0] = 100
BARF!!!
```

Dictionaries

```python
>>> D = {}
>>> D["Geoff"] = "spam"
>>> D["Zach"] = "donuts"
>>> D["Alien"] = 42
>>> D["Geoff"]
'spam'
>>> D["Alien"]
42
>>> D["Suicide Squad"]
BARF!
>>> D
{'Geoff': 'spam', 'Zach': 'donuts', 'Alien': 42}
```
Finding Shortest Paths

Sometimes We Need to Make More Than 2 Recursive Calls!

Shortest path? Is greed good? How does the use-it-or-lose-it idea get used here?
We Admit It’s Tricky

def shortestPath (cities, dists):
    '''Returns the length of the shortest path from the leftmost to the rightmost city in the cities list.'''
    if len(cities) <= 1:
        return 0
    else:
        return min(map(
            lambda hop: dists[cities[0], cities[hop]]
            + shortestPath(cities[hop:], dists),
            range(1, len(cities)))))

SnowFlake Fractals

The Koch Snowflake Fractal:

![Koch Snowflake Fractals Diagram]
Snowflake Fractals

The Koch Snowflake Fractal:

level 0  level 1  level 2  level 3  level 4  level 5

level 0

level 1

1/3  1/3  1/3