CS 61A Summer 2017 — Containers & Data Abstraction

Mentoring 5: July 5, 2017

1 Python Lists

1.1 What would Python display? Draw box-and-pointer diagrams to find out.

(a)
```
a = [1, 2, 3]
a
```

(b)
```
a[2]
```

(c)
```
b = a
a = a + [4, 5]
a
```

(d)
```
b
```

(e)
```
c = a
a = [4, 5]
a
```

(f)
```
c
```

(g)
```
d = c[0:2]
c[0] = 9
d
```

1.2 Draw the environment diagram that results from running the code.

```
def reverse(lst):
    if len(lst) <= 1:
        return lst
    return reverse(lst[1:]) + [lst[0]]

lst = [1, [2, 3], 4]
rev = reverse(lst)
```

1.3 What would Python display? Draw box-and-pointer diagrams to find out.

(a)
```
L = [1, 2, 3]
B = L
B
```

(b)
```
A = L[1:3]
L[0] = A
L = L + A
B
```

(c)
```
B[0] = A[:]
L[0][0] = A
L[0][0][0][0][1]
```

(d)
```
B
```

1.4 Write a function that takes in a list of `nums` and returns a new list containing only the prime numbers from `nums`. Assume that `is_prime(n)` is defined.

Challenge: Write the function body in one line.

```
def filter_primes(nums):
```

2 Data Abstraction

2.1 Your boss, Elaine, is building a database to keep track of all the elephants in her zoo. She's written all the business logic, but she needs your help to implement the data abstractions. Here's some of the code she's written.

```
elephants = [elephant(name="Timothy", age=2,  rating=5.8, can_fly=False),
             elephant(name="Voldie",  age=17, rating=2.1, can_fly=False),
             elephant(name="Eleanor", age=5,  rating=7.2, can_fly=True)]

def eldest_elephant(elephants):
    eldest = None
    eldest_age = -1
    for e in elephants:
        if age(e) > eldest_age:
            eldest = e
            eldest_age = age(e)
    return eldest

def num_parachutes_needed(elephants):
    return len([e for e in elephants if can_fly(e)])

def best_elephant(elephants):
    return max(elephants, key=rating)
```

Define the *constructor* and *selectors* so that the code will run.

Challenge: Define the data abstraction without using containers.