

Lists

Dr Tom Ridge

2018 Q1

What is a list?

- ▶ Informal notions
- ▶ Writing down lists
- ▶ Notation which makes the empty list apparent

List syntax

We probably also need some concrete syntax for lists. 1,2,3 is good; [1,2,3] is better (because you can write the empty list as []). With these in place, we are ready to write lots of those interesting functions we did last week, but using lists.

How are lists and strings related?

A string is either empty, or it is not empty and there is a first character, and the rest of the string.

A list is either empty, or it is not empty and there is a first element and the rest of the list.

So a string **is** a list of characters.

The essential list operations

Recall the basic string operations.

The essential list operations

Recall the basic string operations.

The list equivalents of the basic operations are:

Building up

`nil()`

`cons(obj,l)`

Breaking down

`l.isEmpty()`

`hd(l), tl(l)`

(notice how the two entries in each row correspond to each other)

We use `cons` for adding an element at the front of the list.

Head (`hd`) and tail (`tl`) are easy to remember. `nil` is also OK. `cons` (“construct”) is a bit difficult to remember if you have not encountered it before.

Experimenting with lists

Let's see what Java makes of various expressions involving lists. . .

```
1  nil();  
2  
3  world = cons("world!",nil());  
4  
5  hello_world = cons("hello ", world);  
6  
7  hd(world);  
8  hd(world); // note, world is not changed  
9  tl(hello_world);  
10 tl(world); // what does this return
```

What about lists containing strings *and* ints? What about lists containing other lists?

Challenge

Recall the program to calculate the length of a string:

```
int somefun(String s) {  
    int to_return = 0;  
    while(true) {  
        if(s.equals("")) return to_return;  
        to_return++;  
        s=s.substring(1);  
    }  
}
```

How can we write the function that computes the length of a list?

List length

```
int somefun(List s) {  
    int to_return = 0;  
    while(true) {  
        if(s.equals("")) return to_return; // change this  
        to_return++;  
        s=s.substring(1); // change this  
    }  
}
```

Challenge: list reverse

How can we write a function to reverse a list? Change the following function which reverses a string.

```
String reverse(String s) {  
    String to_return = "";  
    while(true) {  
        if("").equals(s)) return to_return;  
        to_return=s.charAt(0)+to_return;  
        s=s.substring(1);  
    }  
}  
  
reverse("hello");
```

Append at end of list

Recall that we allowed to join a character on the end of a string. For lists, we allow to join an element on the end of a list, via `append1(List l0, Object o)`.

```
// add at end of list  
List append1(List l0, Object o) {  
    List l = copy(l0);  
    l.add(l.size(),o);  
    return l;  
}
```

Append two lists

Joining two strings is easy. We also allow the equivalent for lists, `append`, to join two lists together.

// join two lists together

```
List append(List l01, List l02) {  
    List to_return = copy(l01);  
    List l2 = copy(l02);  
  
    while(true) {  
        if(l2.isEmpty()) return to_return;  
        to_return=append1(to_return,hd(l2));  
        l2=tl(l2);  
    }  
  
}
```

```
append1(hello_world," again!");  
append(hello_world,hello_world);
```

Recap basic list functions

Building up

`nil()`

`cons(obj,l)`

`append1(l,obj)`

`append(l1,l2)`

Breaking down

`l.isEmpty()`

`hd(l), tl(l)`

List contains

How to implement the following?

```
boolean contains(Object o, List l) {  
    ...  
}
```

List contains

```
boolean contains(Object o, List l) {  
    boolean to_return = false;  
    while(true) {  
        if(l.isEmpty()) return to_return;  
        if(o.equals(hd(l))) { to_return = true; }  
        l=tl(l);  
    }  
}
```

```
List l0 = cons("jen",cons("tom",nil()));
```

```
contains("tm",l0);
```

