

### Introduction to sorting

Dr Tom Ridge

2018 Q1

### A sorted string

Suppose  $s$  is a string of characters.

We say that  $s$  is *sorted* if the characters appear in ascending order.

For example, "abc" is sorted, but "cab" is not.

### Challenge: insert

Given a **sorted** string  $s$ , and a character  $c$ , write a function

```
String insert(char c, String s)
```

This should insert  $c$  in the “correct” place according to the order.  
For example, `insert('c', "abde")` should produce the string "abcde".

## Solution: insert

```
1  String insert(char c, String s) {
2      String to_return = "";
3      while(true) {
4          if(s.equals("")) return to_return+c;
5          char c2 = s.charAt(0);
6          if(c>c2) { to_return = to_return + c2; }
7          else { return to_return+c+s; }
8          s=s.substring(1);
9      }
10 }
```

Demo

## Example execution

## Challenge: sort a string of characters

How can we use insert to sort a string s of characters?

1. Start with an empty string to\_return, which is trivially sorted
2. Look at the first character in s. Put it in the correct place in to\_return, using insert.
3. Drop the first character from s, and repeat from (1) until s is empty.

## Code: insertion sort on strings

```
1  String ins_sort(String s) {
2      String to_return = "";
3      while(true) {
4          if(s.equals("")) return to_return;
5          char c = s.charAt(0);
6          to_return = insert(c,to_return);
7          s=s.substring(1);
8      }
9  }
```

Demo

## Challenge: insert an integer into a list of integers

Can we generalize the code to sort a list of numbers?

## Insert on integer lists

```
1 List insert(Integer i, List l0) {
2     List to_return = nil();
3     while(true) {
4         if(l0.isEmpty()) return append1(to_return,i);
5         Integer i2 = (Integer)(hd(l0)); // note cast!!!
6         if(i>i2) { to_return = append1(to_return,i2); }
7         else { return append(append1(to_return,i),l0); }
8         l0=tl(l0);
9     }
10 }
```

Demo

## Insertion sort for integer lists

```
1 List ins_sort(List l0) {
2     List to_return = nil();
3     while(true) {
4         if(l0.isEmpty()) return to_return;
5         Integer i = (Integer)(hd(l0)); // note cast!!!
6         to_return = insert(i,to_return);
7         l0=tl(l0);
8     }
9 }
```

Demo

## NOTE

The following slides are optional for CO1005 2018-Q1

## Can we generalize further?

Clearly, the code for sorting a string of characters, and sorting a list of integers, is almost the same.

Can we write a version of insertion sort that works for any ordered collection of objects? (strings, lists, vectors, arrays etc)

We need the concept of an **INTERFACE**

## Interfaces

See slides on “Java interfaces”.

## A simple class

```
class Person {  
    public String first=""; public String last="";  
    Person(String f, String l) {first=f; last=l; }  
    public String toString() { return first+" "+last; }  
}
```

```
t = new Person("Tom","Ridge");  
u = new Person("Alf","Bloggs");  
Collections.sort(cons(t,cons(u,nil())));
```

What happens when you call the sort method? Something goes wrong (exception)

## Recap

- ▶ We can sort things using insertion sort!
- ▶ We can sort strings of characters
- ▶ We can sort lists of integers
- ▶ But what happens if, say, we want to sort a list of people by first name?
- ▶ Or last name?
- ▶ Or age?
- ▶ Do we have to write the insertion sort code all over again each time? Fortunately not.

## Recap: interfaces

What are they again?

...

```
1 String ins_sort(String s) {
2   String to_return = "";
3   while(true) {
4     if(s.equals("")) return to_return;
5     char c = s.charAt(0);
6     to_return = insert(c,to_return);
7     s=s.substring(1);
8   }
9 }
```

## Recap: string insertion sort

```
1 String insert(char c, String s) {
2   String to_return = "";
3   while(true) {
4     if(s.equals("")) return to_return+c;
5     char c2 = s.charAt(0);
6     if(c>c2) { to_return = to_return + c2; }
7     else { return to_return+c+s; }
8     s=s.substring(1);
9   }
10 }
```

## Recap: integer list insertion sort

```
1 List insert(Integer i, List l0) {
2   List to_return = nil();
3   while(true) {
4     if(l0.isEmpty()) return append1(to_return,i);
5     Integer i2 = (Integer)(hd(l0)); // note cast!!!
6     if(i>i2) { to_return = append1(to_return,i2); }
7     else { return append(append1(to_return,i),l0); }
8     l0=tl(l0);
9   }
10 }
```

(And similar for ins\_sort)

## Generalizing insertion sort

It is possible to generalize the insertion sort code

- ▶ to handle strings, lists, arrays, etc.

If you are interested, there are slides on “generic insertion sort”

- ▶ but this material will not be examined in class tests.