SLIST REFERENCE

UMD CS DEPARTMENT

1. Public Interface for the SList class

For many projects and labs we will use a common "list" implementation. The idea is to offload you by providing some standard minimal implementation that you are expected to use in order to provide your own implementations, e.g., use this class to write your recursive sorts.

Method Signature	Description
`isEmpty( list )`	Returns `true` iff the `list` is empty.
`first( list )`	Returns the value of the first element in the `list`. Note, an exception is generated if `list` is empty.
`rest( list )`	Returns a list containing the elements following the first element in `list`. Note, an exception is thrown if `list` is empty.
`cons( item, list )`	Creates a new `SList` with `item` as its first element and the contents of `list` the remaining elements.
`length( list )`	Returns a non-negative integer indicating the number of items in `list`. The empty list contains 0 items.
`list( T ... )`	Takes an arbitrary number of arguments of type `T` and returns a new list where these elements appear in the order in which they were given to this function.
`NULL`	Not a method: this is a special constant that is used in place of Java's `null` object to denote the empty `SList`.

Table 1. SList class reference

1.1. Examples of using this class.

Consider writing a method that computes the `length` of an `SList<T>` object:

```
public static <T> int length( SList<T> list ) {
    if( isEmpty( list ) ) return 0;
    else return 1 + length( rest( list ) );
```

Observe that no iteration was necessary, and certainly no use of any external Java Collections classes are helpful here; and, that's the idea.