Recursion Problems #1.

1. Write a function that takes two lists as its args & returns the longest as its result. (hint: nothing too elaborate, this one does not need to be recursive, it is just to get you writing a function which contains an "if" form)

2. Write a function which takes a list of numbers as its arg & returns the list with a numbers incremented, eg:
```
(inc-list1 '(5 2 7 4 1)) => (6 3 8 5 2)
```


3. Write a function which takes a mixed list (numbers and other items) & returns the list with all numbers incremented, other items should be left as is, eg:
```
(inc-list1 '(cat 4 dog 3 x 5)) => (cat 5 dog 4 x 6)
```


4. Write a function that takes a list as its only argument & returns that list after enclosing every item in it in an extra set of brackets. eg:
```
(brack-lis '(b l o b))  =>  ((b) (l) (o) (b))
```


5. Develop a function which takes a list of integers as its only argument and returns a list of the factorials of those numbers. (hint: look at inc-list in the notes).
```
(fac-lis '(2 4 3))  =>  (2 24 6)
```

Keep the numbers small and try tracing both this function and the factorial function.

6. Write a function del-item which deletes all occurrences of an item within a list, eg:
```
(del-item 'spam '(egg spam chips spam beans and spam))
=> (egg chips beans and)
```


4.1 Write a Fn, my-replace, which globally replaces some specified item in a list. Replace should take 3 args:
    (a) the item to be replaced
    (b) the new item
    (c) the list

  eg:
```
(my-replace 'the 'a '(the cat sat on the mat))
==>  (a cat sat on a mat)
```

NB: avoid calling this function "replace" because Lisp already has a function with this name.