Intro to programming I

Week 11 - recursion

- A *return* statement inside a function terminates the function, just as if it had reached the end of function. For example,

```
def half_poem():
    print("Roses are red")       # prints Roses are red
    print("Violets are blue")    # prints Violets are blue
    return                       # returns
    print("Sugar is sweet")      # Python cannot reach this print statement
    print("And so are you")      # nor this one
```

- *Recursion* is a technique in which we solve a problem by solving ever smaller versions of the same problem. Many recursive functions follow a template similar to this:

```
def foo():
    if <base condition>:    # base case: is the problem very simple?
        solve problem       # if yes.. solve it...
        return              # ...and we are done!
    else:                   # otherwise: recursive case
        reduce problem      # make the problem smaller...
        foo()               # ...and try again
        return
```

All recursive functions are divided in a base case and a recursive case. The base case is when the solution to our problem is so simple that we can just solve it; the recursive case is when our problem is not simple enough yet so we have to make the problem smaller and then try to solve it again

- We are going to use recursion to take Karel to a wall. The base case is when we are already in front of the wall; in this case we do not need to do anything so we just return. In the recursive case we are not yet near the wall so we take a single step forward, making the problem a little bit smaller, and then try again.

```
def go_to_wall():
    if not front_is_clear():  # base case
        pass                  # do nothing
        return                # we are done!
    else:                     # recursive case
        move()                # reduce problem
        go_to_wall()          # ...and try again
        return
```