# Introduction to Programming W8

Functions I.

2022/05/27

## 1 Exercise 8.1

Complete the code below by writing three functions:

- `flatten_list`: takes a nested list as an argument, and flattens it (i.e., makes a normal list from a nested list). The function returns the flattened list. Most of this function is already done: it iterates through the the elements of the small lists using a nested for loop structure. Note the levels of indentation.
- `filter_numbers`: takes a list of numbers as one of its arguments (`number_list`), and puts the positive numbers inside two lists `evens` or `odds`, based on if the positive number is even or odd. The default values for these arguments are `even_numbers` and `odd_numbers`, respectively. You do not need to change these default values when calling the function. This function does not need to return anything. Tip: use a nested if statement structure (first check if the number is positive or not, then check if it is odd or even).
- `remove_dups`: takes a list as an argument, and returns a list with duplicates removed. The order needs to be preserved. Tip: to preserve the order, a dictionary method needs to be used before the list constructor.

```python
num_lst = [[1,1,4,66],[-55,7],[-1,43,55,-6,12,12,4]]

even_numbers = []
odd_numbers = []

def flatten_list(a_list):
    """docstring"""
    flat_num_lst = []
    for lst in a_list: #iterating through the three smaller lists
        for element in lst: #iterating through the elements of the small lists
            #append the elements to the flattened list

    return flat_num_lst #return the flattened list


def filter_numbers(number_list, evens=even_numbers, odds=odd_numbers):
    """docstring"""
    #complete the function
    for i in number_list:



def remove_dups(a_list):
    """docstring"""
    #complete the function
    dups_removed =
    return dups_removed #return the list with duplicated removed (order preserved)


print("Original nested list:", num_lst)
```

```
flat_num_lst = flatten_list(num_lst) #function call
#next line should print [1, 1, 4, 66, -55, 7, -1, 43, 55, -6, 12, 12, 4]
print("Flattened list:", flat_num_lst)

filter_numbers(flat_num_lst) #function call
print("The even numbers:", even_numbers) #should print [4, 66, 12, 12, 4]
print("The odd numbers:", odd_numbers) #should print [1, 1, 7, 43, 55]

uniq_even_numbers = remove_dups(even_numbers) #function call
uniq_odd_numbers = remove_dups(odd_numbers) #function call
print("The unique even numbers:", uniq_even_numbers) #should print [4, 66, 12]
print("The unique odd numbers:", uniq_odd_numbers) #should print [1, 7, 43, 55]
```

Any technique can be used to complete the exercises, but do not change the variable/function names and definitions already provided. Do not forget to write the docstrings as well, not just for the program but for the functions individually (you can use line breaks in docstrings to make them easier to read). Submit the exercise to Manaba R+. Name the file the following way:
ip_ex81_<student id number>.py, for example: ip_ex81_012345678-9.py