HOMEWORK 6

Solve the following problems using loops. Try to avoid using Python built-in functions unless absolutely necessary.

Problem 1. Write a function filter_positive_even(numbers) that takes a list of integers and returns a new list containing only the positive even numbers.

```
filter_even([1, -2, -3, 4, 5, 6])
```

should return [4, 6]. Note that -2 does not work because it is negative.

Problem 2. Write a function named multiply_factor(a_list, factor) that returns a list where each element in the original list is multiplied by the given factor. For example

```
multiply_factor([1, 2, -3], 2)
```

should return [2, 4, -6].

Problem 3. Write a function named numeric_values(a_list) that takes a list as input and returns a new list with only the numeric elements. Numeric values include both integers and floating-point numbers. For example:

```
numeric_values(["1", "apple", 1, 1.2, -4])
```

should return $[1, 1.2, -4]$. For this problem, you might need to use the `type` function.

Problem 4. Write a function named is_member(a_list, element) that takes a list and an element as input and returns True if this element is a member of a_list. Otherwise, return False. For example

```
removed_element([0, "test", 1, "apple", 0, 1.1], 0)
```

should return True. On the other hand

```
removed_element([0, "test", 1, "apple", 0, 1.1], 2)
```

should return False.

Problem 5. Write a function uppercase_strings(a_list) that takes a list of strings and returns a new list with all the strings in uppercase. For example

```
uppercase_strings(["Anne", "Ben", "David"])
```

should return [“ANNE”, “BEN”, “DAVID”].

Problem 6. Write a function last_names(full_names) that takes a list of full names and returns a new list containing all last names. Here, we assume that each full name consists of a first name followed by a last name, separated by a space. For example,

```
last_names(["Anne Nguyen", "David Hilbert", "Michael Jordan", "
    Alan Turing"])
```

should return [“Nguyen”, “Hilbert”, “Jordan”, ”Turing”]. All of these individuals are famous, except for the first one (at least for now!).

Problem 7. Write a function join_strings(a_list) that takes a list of strings and joins them into a single string, separated by spaces.

```
join_strings(["I", "Love", "Lake", "Forest", "College"])
```

should return “I Love Lake Forest College”.

Problem 8. Write a function longest_string(a_list) that takes a list of strings and returns the longest string. For example

```
longest_string(["apple", "banana", "mango"])
```

should return “banana”.

Problem 9. Given a list of circle radii, create a new list containing the corresponding areas of the circles, rounded to one decimal place. For example, if the list is

```
radii = [1,2 ,3]
```

then, the corresponding areas should be [3.1, 12.6, 28.3].

Problem 10. From a list of strings, create a new list that contains the first letter of each string. For example, if the given list is [“Apple”, “Banana”, “Mango”], the new list should be [“A”, “B”, “M”].

Problem 11. From a list of integers, create a new list that includes only the odd numbers using list comprehension. For example, if the given list is [1, 2, 3, 4, 5, 6], then the new list should be [1, 3, 5].

Problem 12. Write a function named three_largest(alist) that takes a list of numbers as input. The function should return a list containing the three largest values from alist, sorted from lowest to highest. If the list contains fewer than three numbers, return the original list sorted from lowest to highest. For example

```
three_largest([8, 3, 1, 9, 12])
```

should return [8, 9, 12].

For the following problems, you might need to sort the list.

Problem 13. Write a function named find_median(alist) that takes a list of numbers alist as input. The function should return the median value of this list. Recall that the median is defined as follows

- If the list has an odd number of elements: The median is the middle value in the sorted list.
- If the list has an even number of elements: The median is the average of the two middle values.

For example

```
print(find_median([7, 1, 4]))
```

should return 4 and

```
print(find_median([7, 1, 4, 6]))
```

should return $\frac{4+6}{2} = 5$.