1. Write a function `alternate1` which when given two lists combines them by alternating the elements (if one is shorter then just add the rest of the other at the end — The length of the answer is always the sum of the lengths of the two lists). For example:

```
>>> alternate1([1,2],[3,4])
[1,3,2,4]
>>> alternate1([1,2], [3,4,5])
[1,3,2,4,5]
```

2. Write a function `alternate2` which when given two lists and two integers i and j, merge them but now i at a time from the first list and j at a time form the second list (the previous problem is the special case when i = j = 1). For example:

```
>>> alternate2([1,2,3,4],[5,6],2,1)
[1,2,5,3,4,6]
>>> alternate2([1,2,3,4],[5,6],2,2)
[1,2,5,6,3,4]
```

3. Write a recursive function `sum` to compute the sum of the elements of a list. The input is a list of integers. Examples:

```
>>> sum([])
0
>>> sum([1,2,3])
6
```

4. Write a recursive function `num_digits` which returns the number of digits in the given input number. Examples:

```
>>> num_digits(1)
1
>>> num_digits(12345)
5
```

5. Write a function `flatten` which flattens the given list. Examples:

```
>>> flatten([1,2,3,4])
[1,2,3,4]
>>> flatten([1,2,[3,4],[5,6]])
[1,2,3,4,5,6]
```

   Hint: Iterate over each element and check the type of the element to choose the action necessary.

6. Write a function `findLongest` that does the following: given an input, with "word" defined as in "split", find the longest word in the list and print it. For example,

```
>>> findLongest('abcd efgh ijkl     mn,op qr')
'mn,op'
>>> findLongest('abcd123 efgh ijkl     mn,op qr')
'abcd123'
```

7. Write a function `takeinput` that takes as input a string, which is a comma separated list of integers, and returns a list containing the integers present in the string. If the input string contains any object other than integers (and commas) then it should print `'invalid input'`. Below is how the function is expected to behave:

```
>>> takeinput('1')
[1]
>>> takeinput('1,2,3')
[1,2,3]
>>> takeinput('1,2 and 3')
'invalid input'
>>> takeinput('1,2.1,3')
'invalid input'
```

   Hint: You can use the function `split` to split a string based on commas as well. It works in the following way:

```
>>> 'abc,2,3'.split(',')
['abc','2','3']
>>> 'abc,def,g'.split(',')
['abc','def','g']
```

   Once you get this list of strings, try checking the types of the elements and do what is necessary.