# Week 4 Assignment

Design, build and test the following functions: *{updated 19 October 2021}*  
Due 3pm day before next lecture.  
Submit your python script with the filename ‘w4\_yourID.py’.

## signum(a\_num)

If a\_num is greater than zero return 1. If it’s less than zero return integer minus one. Otherwise, return integer zero.  
For example:

| parameter values | return value |
| --- | --- |
| 11 | 1 |
| 0.000702 | 1 |
| -1/2 | -1 |
| 0 | 0 |

## middle(p1, p2, p3)

Return the middle value.  
All three parameters are guaranteed to be pairwise comparable using the operators <, <= !=, ==, >=, and >.  
For example:

| parameter values | return value |
| --- | --- |
| 1 , 2 , 3 | 2 |
| 2 , 3 , 1 | 2 |
| 'a' , 'x' , 'c' | 'c' |
| 'abra' , 'cad' , 'abra' | 'abra' |

## isa\_triangle(len1, len2, len3)

The three parameters are guaranteed to be numbers. If they are interpreted as the lengths of the sides of a euclidean triangle, return an integer to indicate which type of triangle they would construct accoring to this table:

| Return Value | Meaning |
| --- | --- |
| **0** | No triangle possible |
| **1** | Scalene |
| **2** | Isosceles |
| **3** | Equilateral |

## robberlingo(a\_str)

Translate a\_str text into “rövarspråket” (Swedish for “robber’s language”). That is, double every consonant and place an occurrence of “o” in between. For example, translate(“this is fun”) should return the string ” tothohisos isos fofunon”

## merge2(str1, str2)

Interleave the successive characters of **two** strings of the same length into a single string and return the result.  
For example:

| parameter values | return value |
| --- | --- |
| '' , '' | '' |
| 'a' , 'x' | 'ax' |
| 'abc' , 'def' | 'adbecf' |

## merge3(str1, str2, str3)

Interleave the successive characters of **three** strings of the same length into a single string and return the result.  
For example:

| parameter values | return value |
| --- | --- |
| '', '', '' | '' |
| 'a', 'b', 'c' | 'abc' |
| 'abc', 'def', 'ghj' | 'adgbehcfj' |

## mergen\_short()

Interleave the successive characters of **any number** of strings of **any length** into a single string and return the result.  
Interleaving is halted when the shortest string is exhausted.  
For example:

| parameter values | return value |
| --- | --- |
| '', '', '' | '' |
| 'abc', 'd', 'ef' | 'ade' |
| 'abc\_', 'defg', 'hjkmn', 'pqr', 'stuv' | 'adhpsbejqtcfkru' |

## mergen\_long()

Interleave the successive characters of **any number** of strings of **any length** into a single string and return the result.  
As individual strings become exhausted, continue interleaving with the remaining strings until all are exhausted.  
For example:

| parameter values | return value |
| --- | --- |
| '', '', '' | '' |
| 'abc', 'd', 'ef' | 'adebfc' |
| 'abc\_', 'defg', 'hjkmn', 'pqr', 'stuv' | 'adhpsbejqtcfkru\_gmvn' |

## pangram(a\_str):

Answer true if a\_str is a pangram and false if not. Ignore capitalisation and punctuation, only consider the characters a-z.  
For example:

| parameter values | return value |
| --- | --- |
| 'the quick brown fox jumps over the lazy dog' | True |
| 'the quick brown fox jumped over the lazy dog' | False |

## letter\_count(a\_str)

Return a dictionary with every character (including spaces and punctuation) in a\_str as a key with a value equal to the number of occurrences of that character in a\_str. Only characters in a\_str are keys in the returned dictionary.

For example:

| parameter values | return value |
| --- | --- |
| '' | {} |
| 'a' | {'a' : 1} |
| 'aaa' | {'a' : 3} |
| 'abbabab' | {'a' : 3, 'b': 4} |
| 'abracadabra' | {'a' : 5, 'b': 2, 'c': c, 'd': c} |

## runup(a\_str)

The parameter a\_str is guaranteed to be a string made entirely of alphanumeric ([a..z][0..9]) characters. Answer the starting position and length of the longest non-decreasing substring within a\_str. If there are multiple such substrings, report the first. If a\_str is empty return (-1, 0).  
For example:

| parameter values | return value | substring |
| --- | --- | --- |
| '' | -1, 0 | '' |
| 'z' | 0, 1 | 'z' |
| 'ababcb' | 2, 3 | 'abc' |
| 'bus 27 here now' | 1, 4 | 'us27' |
| 'bus 21 here now' | 10, 4 | 'enow' |
| '27991' | 0, 4 | '2799' |

–end–