Homework 11:

Exercise 1. Do the exercises 8.12, 8.13, 8.14 in the textbook.

Exercise 2. Create two functions to convert a string to a number and vice versa, as shown in the following table:

	A	B	C	D	…	Z
10	11	12	13	14	…	36

Exercise 3. Get two primes number p,q with more than 10 digits (https://bigprimes.org/) and create:

a) The function RSA_encrypt to encrypt the message: DSEB.

b) The function RSA_decrypt to decrypt the number in a).

Exercise 4. Write the function: To make the first letter in each word upper case (do not use the .title() method).

Exercise 5. Write the function: count(string, value, start, end) returns the number of times a specified value appears in the string (do not use the .count method)

Parameter Values

Parameter	Description
value	Required. A String. The string to value to search for
start	Optional. An Integer. The position to start the search. Default is 0
end	Optional. An Integer. The position to end the search. Default is the end of the string

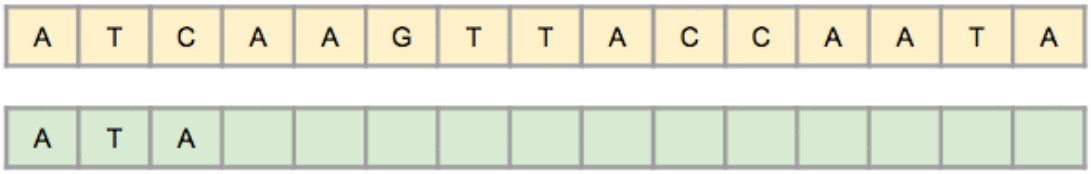
Exercise 6. Write the function: `replace(string, oldvalue, newvalue, count)` replaces a specified phrase with another specified phrase.

Parameter Values

Parameter	Description
oldvalue	Required. The string to search for
newvalue	Required. The string to replace the old value with
count	Optional. A number specifying how many occurrences of the old value you want to replace. Default is all occurrences

Exercise 7. Write the function: **text-searching(p,t)**

- Searches for an occurrence of the pattern p in text t.
- It returns the smallest index i such that p occurs in t starting at index i.
- If p does not occur in t, it returns 0.



Exercise 8. Write the function: **insertion_sort(s)** to sort the sequence $s = [s_0, s_1, \ldots, s_n]$ in nondescreasing order.

6 5 3 1 8 7 2 4

Exercise 9. Write the function: **combination(r,n)** to lists all r-combiations of {1,2,…,n} in increasing lexicographic order (https://en.wikipedia.org/wiki/Lexicographic_order).

Example: **combination(3,4)** returns 123,124, 134, 234.

Exercise 10. Write the function: **permutation(n)** to lists all r-combiations of {1,2,…,n} in increasing lexicographic order.

Example: **permutation(3)** returns 123, 132, 213, 231, 312, 321.