1 Instructions

3. No references to libraries can be made (reference to any find, length, replace, concatenate, append, *etc.* is not allowed);
4. The candidate is required to implement all the utility functions in the exercises;
5. Strings should be treated as arrays.

2 Required output

For all four parts of the problem set the candidate should produce:

1. A console application that allows the user to enter the required inputs and produces the required outputs;
2. Unit-tests on the functions implemented. A high number of unit tests is encouraged to help the candidate assess code correctness. These unit tests can be run with a separate application;
3. A report discussing the code structure, the algorithm used, its complexity, and why this algorithm is optimal. A high level of detail is encouraged;

In addition:

1. The program should only accept strings that contain the English alphabet, no other characters can be used, and the candidate is required to display proper exception handling of wrong format of the input strings.
2. The program should only accept strings for S (and S only, not C) that are divisible by 3.
3. The candidate will turn in all code used to solve the problem set.

3 Part A

Write a console application that performs the following tasks:

1. asks the user for a string S of any length, a string C of any length, and a number N, a positive or negative natural number including zero.
2. based on C builds a new string C_{SHIFT}, which shifts C by N letters, for example the string $C = STRANGE$ shifted by $N = 1$ would become $C_{SHIFT} = TUSBOHF$.
3. switches occurrences of C in the first third of string S with occurrences of C_{SHIFT} in the remaining two thirds of string S. The function should only switch these occurrences until one of the sides runs out of their respective string. For example if $C = ABC$, and $N = 1$, then $C_{SHIFT} = BCD$. Given for example $S = ABCXXABCXXBCDXXBCD$, then the result should be $S = BCDXXABCXXABCXXBCD$

4 Part B

Solve the same problem as in Part A using recursion.

5 Part C

Same as in Part A, with the addition that now the characters in C_{SHIFT} do not need to be consecutive but the can be dispersed in the remaining two thirds of the string. For example if $C = ABC$, and $N = 1$, then $C_{SHIFT} = BCD$. Given for example $S = ABCXXABCXXBXXCXDXBCD$, then the result should be $S = BCDXXABCXXAXXBXCXXBCD$

6 Part D

Solve the same problem as in Part C using recursion.