Programming Assignment 7

In the **Programming/Programming07** folder of your class repository on GitHub, create a main.cpp file. Only include: `iostream, string, cmath, stdexcept,` and `"../Programming06/util.h"`

Use the string variable and functions from **util.h** to assist with the following function definitions:

Task 1: (2 points)

Create a function called **toDecimal()** that takes a constant string reference parameter and an integer parameter. The string parameter represents a signed binary, octal, or hexadecimal number. The integer parameter represents the base of the number. Return an integer that represents the string number as a signed decimal number. Perform the weighted sum technique for all digits of the number. Afterwards, if the string number is negative for the respective base, subtract the base raised to the power of the number of characters in the string from the weighted sum. Ensure the string is valid for the base. If the string is invalid, throw an exception.

Example: $\mathbf{1010_2} = \mathbf{-2^4 + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)}$

$\mathbf{= (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) - 2^4}$

$\mathbf{= 8 + 0 + 2 + 0 - 16}$

$\mathbf{= -6}$

Example: $\mathbf{01010_2} = \mathbf{(0 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)}$

$\mathbf{= 0 + 8 + 0 + 2 + 0}$

$\mathbf{= 10}$

Example: $\mathbf{4321_8} = \mathbf{-8^4 + (4 \times 8^3) + (3 \times 8^2) + (2 \times 8^1) + (1 \times 8^0)}$

$\mathbf{= (4 \times 8^3) + (3 \times 8^2) + (2 \times 8^1) + (1 \times 8^0) - 8^4}$

$\mathbf{= 2{,}048 + 192 + 16 + 8 - 4{,}096}$

$\mathbf{= -1{,}832}$

Example: $\mathbf{12345_8} = \mathbf{(1 \times 8^4) + (2 \times 8^3) + (3 \times 8^2) + (4 \times 8^1) + (5 \times 8^0)}$

$\mathbf{= 4{,}096 + 1{,}024 + 192 + 32 + 5}$

$\mathbf{= 5{,}349}$

Example: $\mathbf{FACE_{16}} = \mathbf{-16^4 + (F \times 16^3) + (A \times 16^2) + (C \times 16^1) + (E \times 16^0)}$

$\mathbf{= (15 \times 16^3) + (10 \times 16^2) + (12 \times 16^1) + (14 \times 16^0) - 16^4}$

$\mathbf{= 61{,}440 + 2{,}560 + 192 + 14 - 65{,}536}$

$\mathbf{= -1330}$

Example: $\mathbf{3BA17_{16}} = \mathbf{(3 \times 16^4) + (B \times 16^3) + (A \times 16^2) + (1 \times 16^1) + (7 \times 16^0)}$

$\mathbf{= (3 \times 16^4) + (11 \times 16^3) + (10 \times 16^2) + (1 \times 16^1) + (7 \times 16^0)}$

$\mathbf{= 196{,}608 + 45{,}056 + 160 + 16 + 7}$

$\mathbf{= 241{,}847}$

Task 2: (2 points)

Create a function called **add()** that takes two constant string reference parameters and an integer parameter. The string parameters represent signed binary, octal or hexadecimal numbers. The integer parameter represents the base of the numbers. Return a string that represents the signed sum of the two string parameters. Ensure the string parameters are valid for the base. If the strings are invalid, throw an exception.

Example: `add("01010","01100",2) => "010110"`
`add("23400751","73476230",8) => "17077201"`
`add("AA1A","989ACD0B",16) => "989A7725"`

Task 3: (2 points)

Create a function called **onesComplement()** that takes a constant string reference parameter and an integer. The string parameter represents a signed binary, octal, or hexadecimal number. The integer parameter represents the base of the number. Return the one's complement of the string parameter. Ensure the string is valid for the base. If the string is invalid, throw an exception.

Example: `onesComplement("01010",2) => "10101"`
`onesComplement("2025",8) => "5752"`
`onesComplement("954F",16) => "6AB0"`

Task 4: (1 point)

Create a function called **twosComplement()** that takes a constant string reference parameter and an integer. The string parameter represents a signed binary, octal, or hexadecimal number. The integer parameter represents the base of the number. Using the functions from tasks 2 and 3, return the two's complement of the string number.

Example: `twosComplement("10101",2) => "01011"`
`twosComplement("6547",8) => "1231"`
`twosComplement("4A2E",16) => "B5D2"`

Task 5: (1 point)

Create a function called **subtract()** that takes two constant string reference parameters and an integer parameter. The string parameters represent signed binary, octal or hexadecimal numbers. The integer parameter represents the base of the numbers. Return a string that represents the signed difference of the two string parameters. Ensure your string parameters are of the same base and are valid strings for the base.

Example: `subtract("01010","11110",2) => "01100"`
`subtract("23400751","73476230",8) => "27702521"`
`subtract("AA1A","989ACD0B",16) => "6764DD0F"`

Task 6: (2 points)

Create a function called **toBase()** that takes two integer parameters. The first parameter represents the decimal number that should be converted to the base indicated by the second parameter. Use repeated division treating the decimal number as positive. If the decimal number is negative, return the two's complement of the result of the repeated division. Otherwise, return the result of the repeated division.

Example: `toBase(25,2) => "011001"`
`toBase(-25,2) => "100111"`
`toBase(2625,8) => "05101"`
`toBase(-2625,8) => "72677"`
`toBase(2625,16) => "0A41"`
`toBase(-2625,16) => "F5BF"`