Programming Assignment 4

All functions should be in a main.cpp file. Only include: **`iostream, string, and Inputs.h.`**

Task 1: (Two Functions)

- Create a function called LSRight() that takes a const string reference parameter and an unsigned int parameter. The string parameter represents a binary number. Return a string that represents the binary number shifted to the right by the number of positions equal to the second parameter. The shift is a logical shift. You can assume the string parameter is a binary number consisting of only 0s and 1s.

  Example: LSRight("1101", 2) => "0011", LSRight("101", 1) => "010", LSRight("101", 3) => "000"

- Create a function called ASRight() that takes a const string reference parameter and an unsigned int parameter. The string parameter represents a signed binary number. Return a string that represents the binary number shifted to the right by the number of positions equal to the second parameter. The shift is an arithmetic shift. You can assume the string parameter is a binary number consisting of only 0s and 1s.

  Example: ASRight("1001", 2) => "1110", ASRight("101", 1) => "110", ASRight("101", 4) => "111"

Task 2: (Two Functions)

- Create a function called LSLeft() that takes a const string reference parameter and an unsigned int parameter. The string parameter represents a binary number. Return a string that represents the binary number shifted to the left by the number of positions equal to the second parameter. The shift is a logical shift. You can assume the string parameter is a binary number consisting of only 0s and 1s.

  Example: LSLeft("1101", 2) => "0100", LSLeft("011", 1) => "110", LSLeft("101", 3) => "000"

- Create a function called ASLeft() that takes a const string reference parameter and an unsigned int parameter. The string parameter represents a signed binary number. Return a string that represents the binary number shifted to the left by the number of positions equal to the second parameter. The shift is an arithmetic shift. You can assume the string parameter is a binary number consisting of only 0s and 1s.

  Example: ASLeft("10101", 2) => "10100", ASLeft("101", 1) => "010", ASLeft("101", 4) => "000"

Task 3:

- Create a function with the header: `bool AND(const Inputs& inputs)`
  Treat each element of the inputs object as an input to an AND operation. Without using &&, return the appropriate truth value. You can treat all Inputs objects as strings. Hint: a truth table may help you visualize the outputs.

  Example: AND(Inputs(“000”)) => 0, AND(Inputs(“010”)) => 0, AND(Inputs(“111”)) => 1

Task 4:

- Create a function with the header: `bool OR(const Inputs& inputs)`
  Treat each element of the inputs object as an input to an OR operation. Without using ||, return the appropriate truth value. You can treat all Inputs objects as strings. Hint: a truth table may help you visualize the outputs.

  Example: OR(Inputs(“000”)) => 0, OR(Inputs(“010”)) => 1, OR(Inputs(“111”)) => 1

Task 5:

- Create a function with the header: `bool XOR(const Inputs& inputs)`
  Treat each element of the inputs object as an input to an OR operation. Without using ||, return the appropriate truth value. You can treat all Inputs objects as strings. Hint: a truth table may help you visualize the outputs.

  Example: XOR(Inputs(“000”)) => 0, XOR(Inputs(“010”)) => 1, XOR(Inputs(“101”)) => 0, XOR(Inputs(“111”)) => 1