Programming Assignment 5

All functions should be in the provide main.cpp file. Do not include any libraries, other than the one already there.

Task 1: (1 point)

- Define a function for the prototype: **`LogicPair 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. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function.

  Example: **`AND(Inputs("000")) => ("AND", 0)`**

  `AND(Inputs("0101")) => ("AND", 0)`

  `AND(Inputs("11111")) => ("AND", 1)`

Task 2: (1 point)

- Define a function for the prototype: **`LogicPair 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. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function.

  Example: **`OR(Inputs("000")) => ("OR", 0)`**

  `OR(Inputs("0101")) => ("OR", 1)`

  `OR(Inputs("11111")) => ("OR", 1)`

Task 3: (1 point)

- Define a function for the prototype: **`LogicPair XOR(const Inputs& inputs);`**
  Treat each element of the inputs object as an input to an XOR operation. Without using &&, ||, or != , return the appropriate truth value. You can treat all Inputs objects as strings. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function.

  Example: **`XOR(Inputs("000")) => ("XOR", 0)`**

  `XOR(Inputs("0111")) => ("XOR", 1)`

  `XOR(Inputs("11101")) => ("XOR", 0)`

Task 4: (1.5 point)

- Define a function for the prototype: **`LogicPair NOT(const LogicPair& pair);`**
  Using the getName() and getValue() methods for LogicPair, return the NOT of the given value along with its appropriate name.

  Example: **`NOT(LogicPair("XOR", 1)) => ("XNOR", 0)`**

  `NOT(LogicPair("XNOR", 0)) => ("XOR", 1)`

  `NOT(LogicPair("AND", 1)) => ("NAND", 0)`

  `NOT(LogicPair("NAND", 1)) => ("AND", 0)`

  `NOT(LogicPair("OR", 1)) => ("NOR", 0)`

  `NOT(LogicPair("NOR", 0)) => ("OR", 1)`

Task 5: (0.5 point)

- Define a function for the prototype: **`LogicPair NAND(const Inputs& inputs);`**
  Treat each element of the inputs object as an input to a NAND operation. Without using &&, return the appropriate truth value. You can treat all Inputs objects as strings. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function. Hint: Use prior functions.
  Example: **`NAND(Inputs("000")) => ("NAND", 1)`**
  `NAND(Inputs("0101")) => ("NAND", 1)`
  `NAND(Inputs("11111")) => ("NAND", 0)`

Task 6: (0.5 point)

- Define a function for the prototype: **`LogicPair NOR(const Inputs& inputs);`**
  Treat each element of the inputs object as an input to a NOR operation. Without using ||, return the appropriate truth value. You can treat all Inputs objects as strings. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function. Hint: Use prior functions.
  Example: **`NOR(Inputs("000")) => ("NOR", 1)`**
  `NOR(Inputs("0101")) => ("NOR", 0)`
  `NOR(Inputs("11111")) => ("NOR", 0)`

Task 7: (0.5 point)

- Define a function for the prototype: **`LogicPair XNOR(const Inputs& inputs);`**
  Treat each element of the inputs object as an input to an XNOR operation. Without using &&, ||, or !=, return the appropriate truth value. You can treat all Inputs objects as strings. The return value should be a LogicPair constructor with two arguments: the name of the operation as a string in all caps , and the result of the function. Hint: Use prior functions.
  Example: **`XNOR(Inputs("000")) => ("XNOR", 1)`**
  `XNOR(Inputs("0111")) => ("XNOR", 0)`
  `XNOR(Inputs("11101")) => ("XNOR", 1)`

Task 8: (4 points)

- Define a function for the prototype:
  `void truthTable(Inputs inputs, LogicPair (*logicFunc)(const Inputs&));`
  Treat each element of the inputs object as an input to the second parameter. The second parameter is a LogicPair function that represents: (AND, OR, XOR, NAND, NOR, XNOR) An example of using the caller truthTable(Inputs(4), XNOR), calling logicFunc() would go as follows:
  `logicFunc(inputs) => ("XNOR", 1)`
  Ensure the size of the first parameter is no greater than 5.
  You can assume the inputs parameter is either all 0s or all 1s.
  Display the header for the table by calling the table_header() function with the size of inputs and the name of the function parameter as arguments. Using a nested for loop (only two), display the truth table for each logical operation.
  If the parameter is all 0s, you can add 1 on each iteration of the loop to get the proper inputs for the row.
  If the parameter is all 1s, you can subtract 1 on each iteration of the loop to get the proper inputs for the row.
  The number of rows is equal to 2^n, where n is the number of inputs. The number of input columns is n.
  Your output should be as follows:

If you start with all 0s.

in1	in2	in3	AND
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

in1	in2	in3	OR
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

in1	in2	in3	XOR
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

in1	in2	in3	NAND
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

in1	in2	in3	NOR
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

in1	in2	in3	XNOR
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

If you start with all 1s.

in1	in2	in3	AND
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

in1	in2	in3	OR
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	0

in1	in2	in3	XOR
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	1
0	0	0	0

in1	in2	in3	NAND
1	1	1	0
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	1

in1	in2	in3	NOR
1	1	1	0
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	1

in1	in2	in3	XNOR
1	1	1	0
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	1