

## CSE 102 Homework Assignment 7 (Due: November 27, 11:55 pm) (Grade weight is twice as much as of a regular assignment)

You are going to write a complete C program which implements the following functionality:

- your program reads two files:
  - `circuit.txt`
  - `input.txt`
- According to content in `circuit.txt`, the program **dynamically** creates necessary structures for a logic circuit and evaluates the cases listed in `input.txt`.
- Your program prints the output to `stdout`. After each output there should be a newline.

### `circuit.txt`

- Each line starts with a **keyword**. Possible keywords:
  - INPUT
  - AND
  - OR
  - NOT
  - FLIPFLOP
- The first line specifies input labels. Labels are separated by spaces. Example:

```
INPUT a input2 c3 k
```
- Here there are 4 inputs are defined. Each has an identifier. `a`, `input2`, `c3`, `k`.
- AND keyword specifies that there is an **and** gate defined. AND keyword follows the identifier for the gate and two other identifiers for the inputs. Example:

```
AND gate_A c3 another_id
```
- Here the **and** gate is identified by the string `gate_A`. Its inputs are identified `c3` and `another_id`. These identifiers can be input identifiers or identifiers for other gates.
- OR keyword specifies that there is an **or** gate defined. OR keyword follows the identifier for the gate and two other identifiers for the inputs. Example:

OR gate\_B ck id3

- Here the **or** gate is identified by the string **gate\_B**. Its inputs are identified **ck** and **id3**. These identifiers can be input identifiers or identifiers for other gates.
- NOT keyword specifies that there is an **not** gate defined. NOT keyword follows the identifier for the gate and one other identifier for its input. Example:

NOT gate\_C c5

- Here the **not** gate is identified by the string **gate\_C**. It has only one input and it is identified by the string **c5**.
- FLIPFLOP keyword specifies that there is an **flip-flop** gate defined. FLIPFLOP keyword follows the identifier for the gate and one other identifier for its input. Example:

FLIPFLOP gate\_F c6

- Here the **flip-flop** gate is identified by the string **gate\_F**. Its input is identified by **c6**.

### **input.txt**

- Each line is a list of 1 and 0. Example:

```
1 0 1 1
0 1 1 1
0 0 1 0
1 0 0 1
```

### **Example:**

- Suppose that **circuit.txt** is has the following content:

```
INPUT a b c d
AND and1 a b
OR or1 and1 c
NOT n1 d
FLIPFLOP f1 n1
AND a2 or1 f1
```

- **input.txt** has the following content:

```
1 1 0 1
1 0 1 0
1 1 1 0
```

- Assume that initially **former-out** of any FLIPFLOP is 0.
- Any FLIPFLOPs should preserve the state throughout the evaluation of the whole `input.txt`.
- Each line in `input.txt` is assigned to identifiers `a`, `b`, `c`, `d`, defined in `circuit.txt`. According to the truth tables, outputs of gates are calculated.
- For the `input.txt` given, the output of your program should be:

0  
1  
0

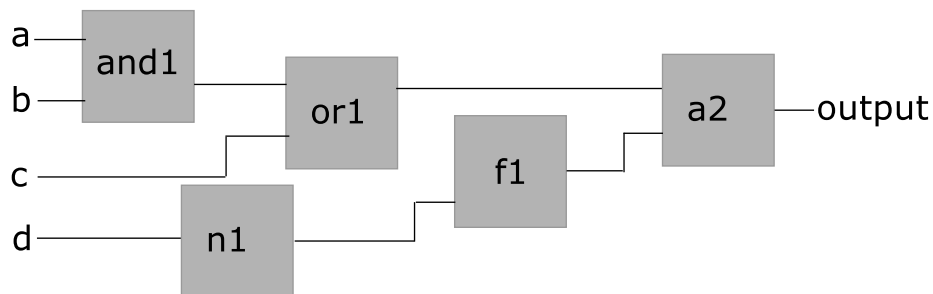


Figure 1: Example Logic Circuit

## Remarks

- Output is not defined explicitly. It is your job to figure out the output pin. There will always going to be one output pin.
- Each identifier is unique
- There won't be any errors in the files.
- You have to use **dynamical memory allocation** and **struct**.

## Turn in:

A complete C program `<assignment_7_name_id.c>` which can be compiled using the following command:

```
gcc -std=c99 assignment_7_name_id.c -o assignment_7_name_id
```

If your program requires additional compile and link options, state that requirement at beginning of your source code as a comment.

### Caution:

- Read and apply “Assignment Submission Rules and Other Related Information” document which available on the class e-learning system.
- You may or may not get partial credit depending on how you structured or documented your code.

### Truth Tables:

- AND

a	b	out
0	0	0
0	1	0
1	0	0
1	1	1

- OR

a	b	out
0	0	0
0	1	1
1	0	1
1	1	1

- NOT

a	out
0	1
1	0

- FLIPFLOP

a	former_out	out
0	0	0
0	1	1
1	0	1
1	1	0