USP – ICMC – SSC

SSC0603 – Estrutura de Dados I (ED1) – 2024/2

TRABALHO TP03 – BRAIN System

Digital Circuit Simulator + Calculator (Math) + Decision (IF)

BRAIN System / Circuit Simulator + Calculator (Math) + Decison (IF) – Trabalho INDIVIDUAL
****

Versão 1.0r00 - VER sempre o PDF com descrição completa!
ATENÇÂO: DATA LIMITE DE UPLOAD 15/12/23 23:55:55
(Sistema fecha automaticamente a submissão na data/hora marcada no RunCodes)

> Usando Árvores Binárias [TAD Backes, FOsorio, ou outro de ArvBin]

> Criar uma árvore que permita usar ao MESMO tempo nodos para cálculo e simulação de
(A) Circuitos Digitais, (B) Calculadoras Matemática e (C) Regras de Decisão (IFs).

OPERADORES DO BRAIN System

(A) Criar um Simulador de Circuitos Digitais (circuitos sequenciais e combinacionais)
com portas lógicas e valores binários (representados por floats ou double – ponto flutuante).
>> Os circuitos devem simular as portas lógicas AND, OR, XOR e NOT
As portas lógicas possuem 2 entradas e 1 saída (sendo que o NOT só usa a 1a. entrada, ignora a 2a.)
Mais abaixo vamos detalhar cada uma destas operações lógicas binárias.

(B) Crias uma Calculadora Matemática básica (expressões aritméticas, operações de calculadora)
com expressões representadas na forma de nodos de uma árvore binária (valores float ou double).
>> As expressões devem aceitar os seguintes operadores: ADD, SUB, MULT, DIV, MOD (Resto),
PERC (Percentual), SQRT (Raiz), POW (Potência ao Quadrado), BIN (Binarizador),
MEM-Store (1,2,3), MEM-Recall (1,2,3), CONST_Zero, CONST_Um
Mais abaixo vamos detalhar cada uma destas operações numéricas/aritméticas.

(C) Criar uma estrutura de Decisão Condicional (expressões condicionais com IFs)
com expressões (decisões) representadas na forma de nodos de uma árvore binária (IFs).
>> As decisões condicionais vão receber valores de ponto flutuante (float ou double) e gerar
valores de falso (0.00 em pto. flutuante) ou de verdadeiro (1.00 em pto. flutuante) no caso dos
operadores: IFGT (“Greater”: Maior que => True/False), IFLT (“Less”: Menor que => True/False)
IFEQ (“Equal”: Iguais => True/False).
>> As decisões condicionais “de passagem” vão repassar o valor de entrada caso ele satisfaça
a condição, por exemplo: IPGT: Greater – Se A maior que B, passa o valor de A, senão passa B
IFLT: Less – Se A menor que B, passa o valor de A, senão passa o valor de B.
Mais abaixo vamos detalhar cada uma destas operações condicionais.

1. Construção da Árvore Binária (Ordenada) representando a Expressão do Circuito Lógico Combinacional, a Expressão Numérica-Aritmética e a Expressão Condicional.

> O simulador deve CRIAR o circuito com a descrição fornecida em um arquivo TEXTO de entrada. O arquivo "brain.txt" descreve o circuito+cálculos+decisões e permite que seja construída a árvore que representa ele.

> O arquivo "brain.txt" possui nodos com IDs (identificadores com valores inteiros) que vão permitir que seja construída uma Árvore Binária Ordenada (usando os IDs) EXATAMENTE do modo como foi projetada e descrita no arquivo "brain.txt" (descrição/detalhes do arquivo indicados mais abaixo).

2. Inserção dos Dados de Entrada da Árvore Binária colocando valores nos Nodos Folhas

> Inserção dos Dados de Entrada da Árvore, alimentando as entradas que são os nodos folhas com os valores de entrada da árvore (representados por valores de ponto flutuante.

> Para efeito dos circuitos lógicos digitais, 0.0 é ZERO (binário) e 1.0 é UM (binário), onde caso um valor diferente de ZERO seja entrado em uma porta lógica, ele deve ser considerado UM, ou seja: 0.0 é ZERO, qualquer outro valor que não seja 0.0 é considerado UM nas portas lógicas.

> O arquivo texto "input.txt" descreve as entradas da Árvore Binária (dados dos nodos folha). O arquivo "input.txt" permite que a Árvore seja alimentada com os dados de entrada que serão "propagados até a raiz", obtendo assim a "resposta final da simulação da expressão da árvore". Descrição/detalhes do arquivo de entrada "input.txt" serão indicados mais abaixo.

3. Simulação: Propagação das entradas através dos Nodos da Árvore Binária ("executa a expressão representada pela árvore, simulando e propagando os dados")

> A propagação dos sinais de entrada vem "subindo" pelos nodos da árvore, dos nodos folhas até a raiz, que irá armazenar o valor final da expressão representada pela árvore binária.
> Este valor final da execução da expressão da árvore será exibido na tela no formato: x.xx (valor float ou double com 2 casas após a vírgula, podendo se um valor binário ou não (pto. flutuante)
A saída poderá ser: 0.00 ou 1.00 quando o nodo raiz obter um valor final de saída binária.
A saída poderá ser: x.xx (valor de pto. flutuante) quando o resultado do nodo raiz for um valor resultante de uma operação de cálculo ou que resulte em uma saída numérica qualquer.

Nas aulas AULA21 e AULA22 foram discutidos os principais conceitos deste trabalho (circuito).
Na aula AULA24 foram discutidos os principais conceitos deste trabalho (cálculos/decisões IF).

O TP03 é uma "evolução" (nova versão) do TP02 onde foram incluídos mais "operadores" no TP03.
O TP03 é uma continuação do TP02 do BRAIN (Versão compatível e aperfeiçoada do BRAIN).

Modo de operação:

- O programa carrega o arquivo "brain.txt" e constrói a árvore binária, ordenada pelos IDs.
- O programa carrega as entradas do arquivo "inputs.txt" e "alimenta a árvore" (nodos folhas).
- O programa realiza a simulação (propaga sinais/valores) e obtém uma saída final.
- O programa exibe a saída final obtida da simulação realizada.
- O programa TERMINA.

(A) Descrição dos Arquivos de Definição da Árvore do Circuito: **brain.txt e input.txt**
EXEMPLO EXATAMENTE IGUAL AO EXEMPLO DO TP02

A.1. Descrição do Arquivo do Circuito: brain.txt

O arquivo é composto por pares de linhas contendo o ID seguido na linha abaixo do tipo de NODO.

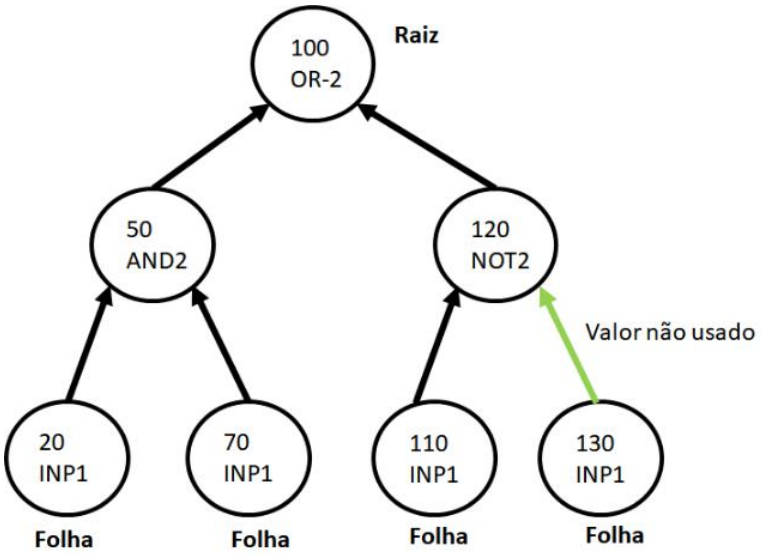
O ID é um número INTEIRO e serve para determinar a ordenação dos nodos da árvore, e assim, garantir que a árvore construída (definida pelo arquivo brain.txt, definido pelo professor) seja exatamente do modo como foi planejada (cria o circuito do modo descrito e previsto).
O arquivo termina com um ID contendo um valor negativo (-1).

O Tipo_do_Nodo é uma string que contém um texto indicando o tipo do nodo (4 Letras).
Os tipos de nodos (portas lógicas) disponíveis neste trabalho são:

AND2 - AND com 2 entradas
OR-2 - OR com 2 entradas
XOR2 - XOR com 2 entradas
NOT2 - NOT com 2 entradas, porém apenas a primeira é usada.
INP1 - Nodo de Entrada (Input) com 1 entradas (não realiza processamento, só guarda valor). Somente os nodos folhas tem apenas uma entrada (um valor)

EXEMPLO: CIRCUITO: ([20] **AND** [70]) **OR** (**NOT** [110])

```
100
OR-2
50
AND2
20
INP1
70
INP1
120
NOT2
110
INP1
130
INP1
-1
```



A.2. Descrição do Arquivo de Entradas do Circuito: inputs.txt

O arquivo é composto por pares de linhas contendo o ID de um nodo folha e um valor.
O valor é um valor de ponto flutuante, podendo ser 0.0 ou 1.0
O ID é sempre de um nodo folha.
O arquivo deve ter pares de linhas ID + Valor para cada nodo folha, sendo que também termina com o valor do ID negativo (-1).

EXEMPLO: SAÍDA FINAL DESTA SIMULAÇÂO: 1.00

```
20
1.0
70
1.0
110
1.0
130
1.0
-1
```

ERROS na Construção da Árvore ou na inserção das Entradas,
Devem gerar uma saída do programa indicando: FAIL na tela.
Exemplo de erros: Nodo com Tipo Inválido, Inserção de entrada em nodo que não é folha.

Sugere-se que cada nodo armazene: ID, TIPO_NODO, INPUT_1, INPUT_2 e OUTPUT

==========

(B) Descrição dos Arquivos de Definição da Expressões da Calculadora: **brain.txt e input.txt**
EXPANDINDO O BRAIN do TP02 para executar operações e expressões aritméticas.

B.1. Descrição do Arquivo com Operações e Expressões Aritméticas: brain.txt

O arquivo é composto por pares de linhas contendo o ID seguido na linha abaixo do tipo de NODO.

O ID é um número INTEIRO e serve para determinar a ordenação dos nodos da árvore, e assim, garantir que a árvore construída (definida pelo arquivo brain.txt, definido pelo professor) seja exatamente do modo como foi planejada (cria o circuito do modo descrito e previsto).
O arquivo termina com um ID contendo um valor negativo (-1).

O Tipo_do_Nodo é uma string que contém um texto indicando o tipo do nodo (4 Letras).

Os tipos de nodos (operações e expressões aritméticas) disponíveis neste trabalho são:

ADD2 - ADD com 2 entradas. Operação Soma: Saída = A + B (valores de pto. flutuante: A, B e S)

SUB2 - SUB com 2 entradas. Operação Subtrai: Saída = A – B (valores de pto. flutuante: A, B e S)

MLT2 - MULT com 2 entradas. Operação Multiplica: Saída = A * B (pto. flutuante: A, B e S)

DIV2 - DIV com 2 entradas. Operação Divide: Saída = A / B (pto. flutuante: A, B e S - Saída)

MOD2 – MOD com 2 entradas. Operação “Mod” (resto da divisão inteira entre 2 números), considerando a parte inteira dos números. Saída = A % B (pto. flutuante: A, B e S - Saída)

PERC – PERC com 2 entradas. Operação “Percentual” (obtêm B percentual de A). Saída: B perc A. Por exemplo: se A vale 200.0 e B vale 10.0, S=20.0 ou seja 10 porcento de 200.

SQR2 – SQRT com 2 entradas. Operação Raiz Quadrada: S = RaizQuadrada (A+B) com pto. flut. Note que é calculada a raiz da soma de A+B, se B for zero é calculada a raiz de A (A+Zero), se A for zero é calculada a raiz de B (B+Zero), ou, soma A+B e tira a raiz quadrada.

POW2 – POW com 2 entradas. Operação Eleva ao quadrado: S = Quadrado (A+B) com pto. flut.

Note que é calculada o quadrado da soma de A+B, se B for zero é A ao quadrado $(A+Zero)^2$, se A for zero é calculada o quadrado de B $(B+Zero)^2$, ou, soma A+B e eleva ao quadrado.

BIN2 – Binariza a soma (A+B). Se (A+B) igual a zero => S=0.00, senão (A+B) não é zero => S=1.00

Se A é zero, binariza B, Se B é zero, binariza A.

Operações de Memória da Calculadora: 3 posições de memória que permitem armazenar valores.

MV12 – Posição V1 da Memória de um Valor, com 2 entradas (B=1.00: Store ; ou ; B=0.00: Recall)
Se B for 1.00: Armazena A na memória 1; Se B for 0.00: Recupera A da memória 1.

MV22 – Posição V2 da Memória de um Valor, com 2 entradas (B=1.00: Store ; ou ; B=0.00: Recall)
Se B for 1.00: Armazena A na memória 2; Se B for 0.00: Recupera A da memória 2

MV32 – Posição V3 da Memória de um Valor, com 2 entradas (B=1.00: Store ; ou ; B=0.00: Recall)
Se B for 1.00: Armazena A na memória 3; Se B for 0.00: Recupera A da memória 3

CTE0 - Não interessa os valores das entradas A e B, sempre sai na Saída com 0.00 (valor constante)

CTE1 - Não interessa os valores das entradas A e B, sempre sai na Saída com 1.00 (valor constante)

ATENÇÃO:

As expressões numéricas podem ser combinadas com os nodos de circuitos lógicos e vice-versa.

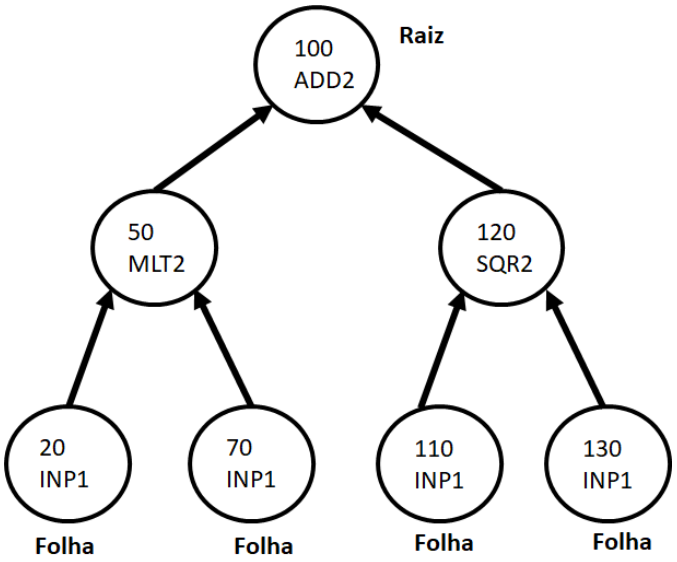
As expressões numéricas podem receber valores de nodos de entrada com os mesmos nodos "INP1"

INP1 - Nodo de Entrada (Input) com 1 entradas (não realiza processamento, só guarda valor).
Somente os nodos folhas tem apenas uma entrada (um valor)

EXEMPLO:

```
100
ADD2
50
MLT2
20
INP1
70
INP1
120
SQR2
110
INP1
130
INP1
-1
```

ÁRVORE: ([20] ***** [70]) **+** (**Raiz** [[110] + [130]])



B.2. Descrição do Arquivo de Entradas da Árvore: inputs.txt

O arquivo é composto por pares de linhas contendo o ID de um nodo folha e um valor. O valor é um valor de ponto flutuante, podendo ser qualquer valor de pto. flutuante. O ID é sempre de um nodo folha. O arquivo deve ter pares de linhas ID + Valor para cada nodo folha, sendo que também termina com o valor do ID negativo (-1).

EXEMPLO: SAÍDA FINAL DESTA SIMULAÇÂO: (10.0*5.0) + Raiz(4.0) = 52.00

```
20
10.0
70
5.0
110
2.0
130
2.0
-1
```

==========

(C) Descrição dos Arquivos de Definição da Expressões da Condicionais: **brain.txt e input.txt**
EXPANDINDO O BRAIN do TP02 para executar operações e expressões condicionais (IF).

C.1. Descrição do Arquivo com Operações e Expressões Condicionais (IF): brain.txt

O arquivo é composto por pares de linhas contendo o ID seguido na linha abaixo do tipo de NODO.

O ID é um número INTEIRO e serve para determinar a ordenação dos nodos da árvore, e assim, garantir que a árvore construída (definida pelo arquivo brain.txt, definido pelo professor) seja exatamente do modo como foi planejada (cria o circuito do modo descrito e previsto).
O arquivo termina com um ID contendo um valor negativo (-1).

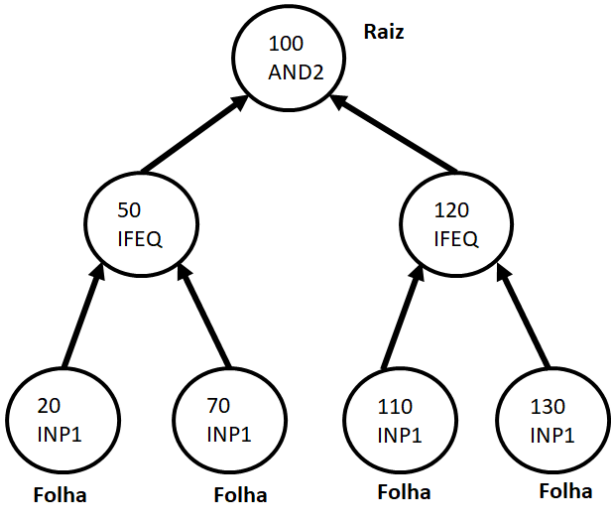
O Tipo_do_Nodo é uma string que contém um texto indicando o tipo do nodo (4 Letras).
Os tipos de nodos (operações e expressões aritméticas) disponíveis neste trabalho são:
IFGT – IF Greater com 2 entradas. Operação A Maior que B: Verdadeiro (1.00) ou Falso (0.00)
SE A > B Então Saída = 1.00 - Valores de entrada de pto. flutuante: A, B; Saída Binária.
SE A não é maior que B Então Saida = 0.00 (Falso)
IFLT – IF Less com 2 entradas. Operação A Menor que B: Verdadeiro (1.00) ou Falso (0.00)
SE A < B Então Saída = 1.00 - Valores de entrada de pto. flutuante: A, B; Saída Binária.
SE A não é menor que B Então Saida = 0.00 (Falso)
IFEQ – IF Equal com 2 entradas. Operação A Igual a B: Verdadeiro (1.00) ou Falso (0.00)
SE A igual B Então Saída = 1.00 - Valores de entrada de pto. flutuante: A, B; Saída Binária.
SE A não igual B Então Saida = 0.00 (Falso)

Vamos ter também expressões condicionais "de passagem" (passa o valor condicionalmente)
IPGT – IF Pass Greater com 2 entradas. Operação Passar A se A Maior que B, Senão Passar B
SE A maior que B Então Passa A para a Saida (IF A > B THEN S = A)
SE A não é maior que B Então Passa B para a Saída (IF A <= B THEN S = B)
IPLT – IF Pass Less com 2 entradas. Operação Passar A se A Menor que B, Senão Passar B
SE A menor que B Então Passa A para a Saida (IF A < B THEN S = A)
SE A não é menor que B Então Passa B para a Saída (IF A >= B THEN S = B)
Note que este operador (nodo) recebe 2 valores numéricos de entrada e sai um valor numérico.
Os operadores IPGT e IPLT **não são operadores de saída binária**, a saída é A ou B conforme o IF, podendo ser um valor numérico qualquer e não apenas um valor binário, como nos outros IFs.

ATENÇÃO:
As expressões lógicas podem ser combinadas com os demais nodos de circuitos lógicos e nodos de expressões aritméticos e vice-versa.
As expressões lógicas podem receber valores de nodos de entrada com os mesmos nodos "INP1"
INP1 - Nodo de Entrada (Input) com 1 entradas (não realiza processamento, só guarda valor).
Somente os nodos folhas tem apenas uma entrada (um valor)

EXEMPLO: ÁRVORE: **IFEQ** ([20] **,** [70]) **AND IFEQ** ([110] , [130])

```
100
AND2
50
IFEQ
20
INP1
70
INP1
120
IFEQ
110
INP1
130
INP1
-1
```



C.2. Descrição do Arquivo de Entradas da Árvore: inputs.txt

O arquivo é composto por pares de linhas contendo o ID de um nodo folha e um valor. O valor é um valor de ponto flutuante, podendo ser qualquer valor de pto. flutuante. O ID é sempre de um nodo folha. O arquivo deve ter pares de linhas ID + Valor para cada nodo folha, sendo que também termina com o valor do ID negativo (-1).

EXEMPLO: SAÍDA FINAL: Se as quatro entradas são iguais => 1.00

```
20
123.0
70
123.0
110
123.0
130
123.0
-1
```

```
Criar um MAKEFILE (Zip) para compilar o programa.
 Pode ser feito em um ".c" , mas com desconto na nota final se não for usado MAKEFILE.
 Os arquivos "brain.txt" e "input.txt" estarão carregados e disponíveis junto aos
 casos de teste (não use path e use exatamente o nome indicado dos arquivos).


BOM
TRABALHO!

========
 F.Osório
 Nov.2024
========
```