UNISINOS - **UNIVERSIDADE DO VALE DO RIO DOS SINOS**
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS (C6/6) – Curso: Informática

LABORATÓRIO II

Disciplina: Linguagem de Programação PASCAL
Professor responsável: ***Fernando Santos Osório***
Semestre: 2000/2
Horário: 31

E-mail: ***osorio@exatas.unisinos.br***
Web:
http://www.inf.unisinos.br/~osorio/lab2.html
Xerox : *Pasta 54* – LAB. II (Xerox do C6/6)

TRABALHO PRÁTICO 2000/2 – GRAU B (Versão 1.0)

Faça um programa para "simular" um circuito lógico usando árvores de um tipo similar as árvores binárias (sugestão: árvores ternárias), com alocação dinâmica (usando ponteiros), de acordo com a descrição dada abaixo. O programa deve possuir um menu com as seguintes opções:

1. Ler circuito de um arquivo em disco
2. Exibir a árvore do circuito lógico
3. Verificar o circuito
4. Exibir a função implementada pelo circuito
5. Simular o circuito lógico – Entrada de um arquivo
6. Simular o circuito lógico – Entrada do teclado
7. Salvar o circuito em um arquivo em disco
8. Terminar a execução do simulador

1. Ler arquivo e criar circuito: Esta opção deve permitir ao usuário ler um arquivo texto do disco, cujo nome será informado por ele. Este arquivo contem a descrição de um circuito lógico, conforme descrição informada mais abaixo. O arquivo irá permitir que seja criada uma nova árvore descrevendo o circuito lido, e posteriormente simular este circuito. Exemplo:

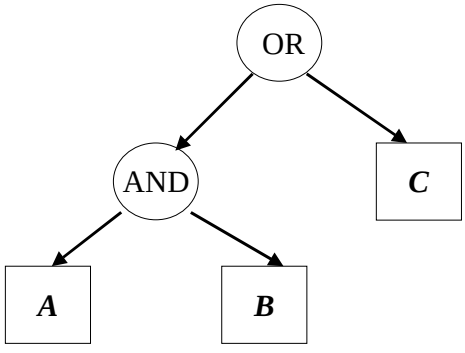
   Entre com o nome do circuito: circuito.txt
   >> Lendo arquivo...
   >> Circuito lido, estrutura de dados criada.

2. Exibir o circuito lógico na tela: exibir a árvore de modo a permitir que o usuário possa visualizar os diferentes níveis desta, com a descrição dos seus respectivos nodos (tipo de operador). Exemplo: (A and B) or C <=> Exibido da forma indicada a esquerda

Saída na Tela:

```
OR  AND  A
         B
    C
```

```
OR | AND | A
   |     | B
   | C
```



3. Verificar se o circuito lógico está correto: verificar se todos os nodos internos da árvore possuem definida uma função lógica correta (suas ligações de entrada e saída estão bem conectadas), verificar se todos os nodos folhas são do tipo entrada, verificar se o circuito possui uma única saída bem definida (raiz única). O circuito não deve ter laços.

>> Verificando Circuito...	***>> Verificando Circuito...***
>> Portas Lógicas: OK.	***>> Portas Lógicas: ERRO – Falta entrada na porta S2***
>> Entradas do Circuito: OK.	***>> Entradas do Circuito: ERRO – Porta em nodo folha***
>> Saídas do Circuito: OK.	***>> Saídas do Circuito: OK.***
>> Circuito Verificado: OK!	***>> Circuito Verificado: COM ERROS!***

4. Exibir a função lógica, conforme indicação dada no final do arquivo que descreve o circuito (circuito final informado no arquivo). Exibir em seguida, em modo pós-fixado (EDV), a função lógica do circuito carregado na memória, conforme os dados contidos na árvore. Exemplo:

   Circuito final conforme descrito no arquivo: ((A and B) and C) or (D and E)
   Circuito armazenado na árvore (pós-fixado): A B and C and D E and or

5. Simular o circuito lógico baseado nos dados de entrada lidos de um arquivo em disco, obtendo o valor da saída final deste circuito. Esta opção irá ler um arquivo texto, cujo nome é informado pelo usuário, e usar os dados lidos para realizar a simulação (dados de entrada = valores dos nodos folhas da árvore). Como resultado da simulação, vamos obter um valor de saída que deve ser exibido na tela - uma única saída, com valor 0 ou 1. Exemplo: considerando o seguinte circuito ((E1 and E2) and E3) or (E4 and E5)

   Entre com o nome do arquivo de entrada: input.txt
   Valores de entrada lidos do arquivo:
   E1: 0 E2: 1 E3: 0 E4: 0 E5: 1
   Valor de saída do circuito: 0 (zero)

6. Simular o circuito lógico baseado nos dados de entrada lidos do teclado. Esta opção realiza a mesma operação que é feita peloa opção anterior, só que os dados de entrada são obtidos através da leitura destes pelo teclado. Esta opção irá ler valores 0 ou 1 correspondendo a cada uma das entradas do circuito – nodos folhas (dados de entrada). Como resultado da simulação, vamos obter um valor de saída que deve ser exibido na tela, sendo uma saída única, com um valor igual à 0 ou 1. Exemplo:

   Entre com os valores das entradas:
   E1: 0 E2: 1 E3: 0 E4: 0 E5: 1
   Valor de saída do circuito: 1 (um)

7. Salvar o circuito em um arquivo em disco com o nome especificado pelo usuário: gerar um arquivo texto contendo a descrição de todos os dados contidos na árvore do circuito. Salvar o arquivo de forma que este possa ser lido posteriormente, reconstruindo exatamente a mesma árvore que foi salva.

8. Sair do programa.

Os arquivos usados pelo programa de simulação de circuitos possuem seus dados organizados de acordo com o formato que vamos especificar aqui. O programa deve respeitar o formato especificado para que possam ser lidos os seguintes arquivos:

1. Arquivo de Descrição de Circuitos
2. Arquivo de Entradas

⇨ Arquivo de Descrição de Circuitos:

- Cada linha descreve uma sub-função = porta lógica (operador e seus operandos – um, dois ou três operandos), onde a linha poderá conter os seguintes dados:

  Operador_Unário Operando_Esquerdo Saída
  ou
  Operador_Binário Operando_Esquerdo Operando_Direito Saída
  ou
  Operador_Ternário Operando_Esquerdo Operando_Centro Operando_Direito Saída

  Operadores Unários: NOT (Not = Negação, Xor = Exclusive Or)
  Operadores Binários: AND, OR2, XOR, NOR (Or2 = Or 2 entradas, NOR = Not Or)
  Operadores Ternários: OR3, XR3 (OR3 = Or 3 entradas, XR3 = Xor 3 entradas)

- O arquivo termina com duas linhas especiais, uma contém "# Circuito Final" que indica o final das descrições das sub-funções, e a linha seguinte que contém uma descrição do circuito lido (esta é a função lida que será apresentada na opção 4). A linha com a descrição do circuito lido é dada pronta e devemos "acreditar" que ela representa corretamente o circuito que foi descrito no arquivo.

  Exemplo:

  ```
  OR2  S2  S3  S4
  AND  S1  E3  S2
  AND  E4  E5  S3
  AND  E1  E2  S1
  # Circuito Final
  ((E1 and E2) and E3) or (E4 and E5)
  ```

  Árvore que deve ser lida:

  ```
                    S4-Or2
            S2-And                  S3-And
    S1-And          E3          E4          E5
  E1      E2
  ```

  A ordem das linhas que descrevem as sub-árvores no arquivo, são colocadas obrigatoriamente de modo que a árvore será construída de cima para baixo, começando pela raiz. Isso deverá simplificar a construção da árvore do circuito.

- Os operadores possíveis são:

  AND (E) , **OR2 / OR3** (Ou) , **XOR /XR3** (Ou exclusivo), **NOR** (E negado), **NOT** (Negação)
  No caso da negação, deve ser gerado um nodo especial que terá somente a árvore esquerda, sem ter uma árvore direita associada. Este nodo deverá ser tratado de modo especial, sendo indicado no arquivo assim: NOT E1 S1. Os nodos OR3e XR3 devem ser tratados como nodos especiais, pois possuem 3 entradas, o nodos OR3 E1 E2 E3 S1 eqüivale a S1= E1 or E2 or E3, e o nodo XR3 E1 E2 E3 S1 eqüivale a S1= E1 xor E2 xor E3.

- Os operandos/parâmetros de uma porta lógica são:

  1. Entradas (nodos folhas). Estes nodos possuem um nome que começa pela **letra E** seguida de um número que indica qual é esta entrada.
  2. Saídas (nodos internos). Estes nodos são identificados pela **letra S** seguida de um número que indica qual é esta saída.

Um nodo da árvore deve portanto conter as seguintes informações:

- *Identificador*. Exemplo: E1, E2, E3, ... (nas folhas) ou S1, S2, S3, ... (nas portas)
- *Tipo do nodo*. Exemplo: Entrada (folha) ou Operador (And, Or2, Xor, ...)
- *Valor resultado*. Exemplo: um nodo And com folhas E1 eE2, o resultado é o valor do cálculo de E1 and E2, podendo resultar em 0 ou 1. Se for um nodo folha o valor é o próprio valor de entrada lido.
- *Ponteiros* para a árvore esquerda, árvore central e árvore direita.
- Estas informações indicadas aqui são o mínimo aconselhado, podendo ser adicionadas outras informações se julgar necessário.

⇨ Arquivo de Entradas:

- Cada linha descreve uma entrada, onde a linha contém os seguintes dados:

Entrada Valor

Exemplo:

```
E1  1
E2  0
E3  0
E4  1
E5  0
```

Atenção: uma mesma entrada pode ser usada repetidas vezes dentro do circuito, podendo existir mais de um nodo folha com a mesma identificação da entrada. Neste caso, o valor da entrada deve ser copiado nos diversos nodos folhas que fazem referência aquela mesma entrada. Exemplo: (E1 and E2) or (E1 and E3)

OBSERVAÇÕES FINAIS:

- O programa deve ser **entregue até o dia 21/11** (aula anterior a prova do Grau B).
- **Entregar** o **programa fonte por e-mail** para o professor, **juntamente com uma listagem impressa** deste programa.
- Este programa deverá simular os circuitos lógicos, baseando-se em uma estrutura de dados do tipo árvore (do tipo das árvores binárias e/ou genéricas). **Esta estrutura de dados deve ser baseada (similar) ao que foi estudado em aula. As rotinas de manipulação de estruturas de dados devem ser criadas de maneira modular.** A boa estruturação e modularidade do programa irá contar na avaliação! *Evite usar variáveis globais*!
- Exemplos de arquivos de circuito usados para testar o simulador serão colocados na Internet junto a página Web da disciplina: http://www.inf.unisinos.br/~osorio/lab2.html

---

BOM TRABALHO!