Expressões Computáveis

1. Introdução

O cálculo numérico é uma metodologia para resolver problemas matemáticos por intermédio do computador, sendo amplamente utilizado por engenheiros e cientistas. Uma solução via Cálculo Numérico é sempre numérica, enquanto os métodos analíticos usualmente fornecem um resultado em termos de funções matemáticas. Muito embora uma solução numérica seja uma aproximação do resultado exato, ela pode ser obtida em grau crescente de exatidão.

Para computar um resultado numérico, são necessárias operações aritméticas (adição, subtração, multiplicação e divisão) e lógicas (comparação, conjunção, disjunção e negação). Considerando que essas são as únicas operações matemáticas que os computadores são capazes de realizar, então os computadores e os Cálculo Numérico formam uma combinação perfeita.

2. Operações Aritméticas

Realizar cálculos aritméticos no computador não é muito diferente de usar um calculadora de mesa. Deve-se, no entanto, ficar atento com prioridade das operações. Vamos recorda-las:

Prioridade	Operação
Alta	Potenciação e Radiciação
Média	Multiplicação e Divisão
Baixa	Adição e Subtração

Se operações tiverem igual prioridade então a resolução deve ocorrer no sentido da leitura, ou seja, da esquerda para a direita. A alteração das prioridades das operações é possível com o uso de parêntesis. Note, no entanto, que não há colchetes ou chaves na notação o computacional de operações algébricas, logo a prioridade de parêntesis também deve ser observada – resolvendo-os dos mais internos para os mais externos.

A notação computacional de operações aritméticas foi pensada de modo a não ser redundante, como a notação matemática por vezes pode ser. Observe o exemplo: 2^{3+7}. Qual operação deve ser feita primeiro? Acerta quem diz que a adição deve ser resolvida previamente a potenciação, o que leva a crer que existe uma exceção nas regras de prioridades. Na verdade, não há. O que ocorre é que a maneira como a notação matemática induz o leitor enxergar a prioridade na adição dos expoentes. O computador não terá esse mesma percepção! Observe os operadores abaixo.

Operação	Operador	Exemplo
Adição	+	2 + 3 = 5
Subtração	-	5 – 3 = 2
Multiplicação	*	3 * 2 = 6

Divisão	/	6 / 2 = 3
Potenciação	^	2^3 = 8

Para exemplificar vamos converter o exemplo anterior da notação algébrica convencional para a notação computacional: 2^{3+7} = 2^**(**3+7**)**. Note que o uso de parêntesis nesse caso precisa ser explicitado, caso contrário as regras de prioridade determinarão que o computador resolva primeiramente a potencia (2^3) e posteriormente a adição (+7).

Dica: Sempre faça uso de parêntesis na notação matemática computacional, mesmo se for redundante em relação as regras de prioridade. Isso demonstra que você estava ciente das prioridades envolvidas naquela operação no ato da escrita.

Os operadores apresentados acima são padrão na computação (salvo alguma exceções), diferentemente de funções específicas de linguagens de programação usadas em outras operações matemáticas tais quais: raiz quadrada, tangente, modulo, etc. Essas funções serão tratadas nas aulas práticas dependendo da linguagem que estivermos usando para a resolução do problemas que as demande. Em princípio, usaremos um notação livre similar a da matemática. Ex. raiz_q(9) = 3;

3. Operações Lógicas

Para o computador, as expressões lógicas são sempre uma pergunta cuja a resposta pode ser somente "sim" ou "não", mais precisamente, "verdadeiro" ou "falso". Os operadores relacionais usados nessas expressões também já são em sua maioria nossos velhos conhecidos da matemática elementar:

Operação	Operador	Exemplo
Igualdade	=	10=10
Diferença	<>	5 <> 10
Maior	>	10 > 5
Menor	<	10 < 5
Maior igual	>=	10 >= 10
Menor igual	<=	10 <= 10

Fique atento! Os operadores relacionais não são tão padronizados como os aritméticos, ou seja, são mais dependentes da sintaxe da linguagem utilizada, especialmente os operadores de igualdade e diferença.

Apesar de nós já conhecermos e trabalharmos essas expressões lógicas na matemática elementar, é comum iniciantes se confundirem com essa notação. Por exemplo, na notação matemática convencional a expressão *x = 10* é uma afirmação e não uma interrogação. Em certas linguagens de programação (como a que usaremos), *x = 10* é uma pergunta e será respondida com *verdadeiro* ou *falso* dependendo do valor da variável *x*. No entanto, se desejamos afirmar que a

variável *x* deve ser equivalente ao valor numérico de *10* então dizemos que *x←10,* onde ler-se *x* recebe *10*. Esse tipo de operação é denominada atribuição.

As "perguntas" representadas pela expressões lógicas podem ser agrupadas da mesma forma que fazemos em linguagem natural, ou seja, utilizando os conectores lógicos "E" e "OU". Exemplo: Se tiver dinheiro **E** o chefe liberar então vou viajar. Note que nessa frase, só obteremos sucesso em viajar se ambas as condições forem satisfeitas. Isso é diferente de dizer: Se tiver dinheiro **OU** o chefe liberar então vou viajar. Na segunda frase basta que uma das condições seja verdadeira para que o objetivo seja alcançado.

Na lógica, conhecemos o "E" como conjunção e o "OU" como disjunção. Há ainda a negação (NÃO) e a disjunção exclusiva (XOU). Observe a tabela verdade das operações lógicas:

P1	P2	Resposta
V	V	V
V	F	F
F	V	F
F	F	F

Tabela 1: Conjunção Lógica (E)

P1	P2	Resposta
V	V	V
V	F	V
F	V	V
F	F	F

Tabela 2: Disjunção Lógica (OU)

P1	P2	Resposta
V	V	F
V	F	V
F	V	V
F	F	F

Tabela 3: Disjunção Exclusiva Lógica (XOU)

A negação, como o próprio nome já diz, inverte o resultado de uma expressão lógica. NAO(*Verdadeiro*) = *falso*.

Assim como nas operações aritméticas, as operações lógicas também possuem regras de prioridade que devem ser seguidas. Observe a seguinte expressão lógica, onde X = 0;

$$P \leftarrow 3 + 3 < 10 \text{ E } 11 > 10 \text{ OU NAO}(X < 1)$$

Nessa expressão não houve qualquer alteração de prioridade por parêntesis logo a prioridade padrão deve ser observada com de acordo com a tabela abaixo:

Prioridade	Operação
Alta	Negação (NAO)

Média	Conjunção (E)
Baixa	Disjunção (OU, XOU)

Note, no entanto, que operações lógicas possuem uma prioridade mais baixa que operações aritméticas logo, no exemplo acima teríamos que resolver primeiramente a operação aritmética: 3 + 3. Ao explicitarmos as prioridade com o uso de parêntesis a expressão lógica P ficaria assim:

$$P \leftarrow (((3 + 3) < 10)\ E\ (11 > 10))\ OU\ (NAO(\ X < 1))$$

Desta forma resolveríamos essa expressão na seguinte ordem:

i. 3 + 3 = 6
ii. 6 < 10 = V
iii. 11 > 10 = V
iv. X = 0 < 1 = V
v. NAO (iv) = F
vi. V (ii) E V(iii) = V
vii. V (vi) OU F(v) = V

Logo, P = *Verdadeiro*.

4. Variáveis

Na Matemática, uma variável é a representação simbólica dos elementos de um certo conjunto que não possui um valor quantitativo fixo. Nos algoritmos, cada variável corresponde a uma posição de memória, a qual nos damos um "apelido" (identificadores). Mas para que servem efetivamente?

Variáveis são usadas sempre que precisamos guardar uma informação para processamento futuro. Exemplo: suponhamos que precisamos calcular a velocidade escalar de um objeto em movimento. Sabemos que:

$$velocidade\ escalar\ \left(\frac{km}{h}\right) = 3.6\ \left(\frac{posição\ inicial - posição\ final\ (metros)}{tempo\ final - tempo\ inicial\ \ (segundos)}\right)$$

Seria impossível proceder esse processamento sem saber os valores de posição inicial, posição final, tempo inicial e tempo final. Logo, esses valores precisam de alguma forma serem informados ao computador para que o mesmo efetue os cálculo. Nós já vimos como montar essa equação na notação computacional de modo que o computador consiga processá-la, no entando essa tarefa só é possível se também determinarmos os valores das variáveis de entrada: posição inicial, posição final, tempo inicial, e tempo final. Observe o exemplo abaixo:

```
...
posicaoInicial ← 0
posicaoFinal ←100
tempoInicial ← 0
```

```
tempoFinal ← 60

velocidadeEscalar ← 3.6 * ((posicaoFinal – posicaoInicial)/ (tempoFinal – tempoInicial))
```

No exemplo acima, as variáveis foram inicializadas com um valor qualquer. Desta forma o computador poderá realizar o cálculo da velocidade escalar. Se precisarmos alterar esses valores basta alterar o valor das variáveis sem nos preocuparmos em acidentalmente alterar também a formula.

Lembre-se: a diferença de um algoritmo para um simples cálculo é que o algoritmo deve ser genérico, ou seja, deve funcionar para qualquer situação e qualquer valor de variáveis. Esses valores quase nunca são conhecidos previamente, caso contrário, seria um mero cálculo. Algoritmo não é uma conta!

Em linguagens de programação fortemente tipadas, tais como Java e o próprio VisuALG, precisamos declarar previamente as variáveis que vamos usar. Isso serve a dois propósitos: a) reservar o espaço de memória apropriado ao tamanho da informação que precisamos armazenar; b) permitir ao computador processar corretamente esses dados, pois operações aritméticas com números de reinos diferentes resultam em valores distintos. Ex: O resto da divisão (7 % 2 = 1) só é possível no reino dos número inteiros.

Durante esse curso trabalharemos em PORTUGOL com 4 tipos primitivos de variáveis:

1. **Inteiro**: Reino dos números inteiros;
2. **Real**: Reino dos números reais (com casas decimais);
3. **Lógico**: Resposta de uma expressão lógica (Verdadeiro/Falso);
4. **Caracter**: Texto, que deve ser identificado por aspas.

Explicação: A grande maioria das linguagens estruturadas, como o VisuALG, definem uma sessão de declaração de variáveis logo no inicio do programa. Já as linguagens orientada-a-objetos, como o Java, trabalham com um conceito de escopo que foge ao entendimento que queremos focar em variáveis nesse momento (será visto em TP II).

Nota: No VisuALG, as variáveis inteiro e real podem ser condensadas um único tipo chamado numérico apesar desse junção não ser recomendada pelo autor.

É importante observar algumas regras e padrões ao nomear variáveis:

i. O nome deve ser representativo! Nada de preguiça e ficar usando *x, y, z*. Esses nomes não revelam para que serve a variável e só servem quando fazemos cálculos específicos de escopo conhecido, como nas disciplinas de física e álgebra por exemplo.
ii. O nome da variável (identificador) não pode:
    a. conter espaço em branco;
    b. caracteres especiais (acentos, cedilha, operadores);

c. Ser uma palavra reservada a outro comando;
d. Começar com números;

iii. Recomenda-se:
a. iniciar com letra minúscula e a cada nova palavras, a primeira letra em Maiúsculo; Exemplos: nomeCliente, corCarro, tempoEstimado
b. Nao deve conter artigos ou preposições;