Métodos para a construção de algoritmo

- Compreender o problema
- Identificar os dados de entrada e objetos desse cenário-problema
- Definir o processamento
- Identificar/definir os dados de saída
- Construir o algoritmo
- Testar o algoritmo

Paradigmas de programação

- Paradigma $\rightarrow$ algo que serve de exemplo geral ou de modelo (padrão).
- Em programação $\rightarrow$ relacionado à **forma de pensar** do programador e como ele **busca a solução** para os problemas apresentados utilizando linguagem de programação.
  - Paradigma *versus* utilização do ferramental disponível.
  - Exemplos: estruturado, orientado a objetos, lógico, funcional, etc.

Paradigmas de programação

- Estruturado (imperativo) → **divide** um **problema** complexo em **pequenas partes** mais simples.
  - Qualquer problema pode ser resolvido utilizando três: **sequencial**, **condicional, iterativa** (repetição).
  - Modularização e parametrização.
- Orientado a objetos (OO) → o **problema** como uma **coleção de objetos** interagindo por meio de **trocas de mensagem**.
  - Procurar objetos → determinar características e responsabilidades → estabelecer como ocorrerá a interação entre os objetos

Instruções de entrada e saída

- Instrução de entrada – uma forma de **obtenção** dos dados de entrada do problema.
- Instrução de saída – uma forma de comunicação da saída por ele **produzida pelo algoritmo**.

- Na linguagem Portugol,
  - a instrução de leitura é **leia**
    - exemplo: **leia (var)**
  - a instrução de saída é **escreva**,
    - exemplo: **escreva (var)**

      escreva (“Olá mundo!”)

Estrutura de um algoritmo

- **Linha de cabeçalho**
- **Declaração de variáveis**
- **Corpo do algoritmo**
- **Linha final**

Estrutura de um algoritmo

- **Linha de cabeçalho –** 1ª linha do algoritmo contém a palavra **algoritmo**.
- **Declaração de variáveis –** seção em que as variáveis são declaradas.
- **Corpo do algoritmo –** contém as instruções que, de fato, fazem o "trabalho" descrito pelo algoritmo. É iniciado pela palavra **início** e terminado pela palavra **fim**.
- **Linha final –** obrigatoriamente a última linha do algoritmo, contém a palavra **fimalgoritmo**.

Particularidades dos algoritmos escritos em PORTUGOL

- **Cada instrução do algoritmo** é escrita em uma **linha dedicada** apenas à instrução.
- As **declarações de variáveis** e o **conteúdo do corpo do algoritmo** são **identados** com relação à **linha de cabeçalho** e a **linha final**.
- Além disso, pode ser que haja linhas em “branco” entre uma seqüência de linhas.
    - Essas linhas em branco servem para separar partes do algoritmo que estão menos relacionadas.

Particularidades dos algoritmos escritos em PORTUGOL (continuação)

- O algoritmo pode conter comentários (altamente recomendado).
  - Comentários são linhas não “executáveis”, ou seja, elas não fazem parte do processo de produção da saída do algoritmo e servem apenas para nos auxiliar na leitura e entendimento da solução do algoritmo.
  - As linhas de comentário se iniciam obrigatoriamente com o símbolo //.
    - Exemplos:

```
// Este algoritmo exemplifica as instrucoes de entrada e saida
// Escreve uma mensagem para indicar o que deve ser lido pelo algoritmo
```

Expressões

- Um conjunto de variáveis e/ou constantes numéricas que se relacionam por meio de operadores compondo uma fórmula.
- Dividem-se em:
  - **Aritméticas**
  - **Relacionais**
  - **Lógicas ou booleanas**

Regra de precedência de operadores

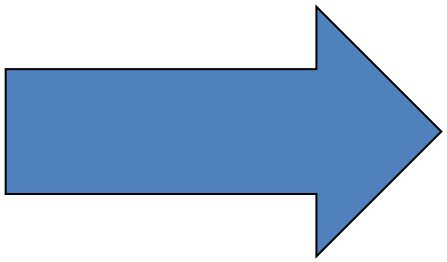
- Regras para definir a ordem em que os operadores aritméticos serão aplicados em uma expressão aritmética qualquer.

- Alterando a prioridade:
  - Utilizam-se os parênteses, pois todo operador possui uma prioridade menor do que a dos parênteses.

Expressões aritméticas

- Qualquer sequência de símbolos formadas apenas por constantes numéricas, variáveis numéricas, operadores aritméticos e parênteses.

Operador	Tipo	Descrição	Precedência
+ -	unários	constante positiva, troca de sinal	1 (máxima)
* / %	binários	multiplicação, divisão e resto da divisão	2
+ -	binários	adição e subtração	3 (mínima)



Observações

- **Multiplicação**: matemática: **x** ou **.** $\rightarrow$ computação: ***
- **Divisão**: matemática: $\div \rightarrow$ computação: **/**
- Normalmente, as linguagens de programação assumem que a **DIVISÃO** é uma operação que retorna um valor **REAL**.
- Se dois operandos de um operador binário possuem tipos distintos (um do tipo **inteiro** e o outro do tipo **real**), o valor do tipo inteiro é convertido para o valor do tipo **real** equivalente.

Expressões relacionais

- Comparação entre dois valores de um mesmo tipo.
- Operadores relacionais da linguagem Portugol.

Operador	Descrição
=	igual a
< >	diferente de
>	maior que
<	menor que
>=	maior ou igual a
<=	menor ou igual a

Observações

- O resultado de operações relacionais é expressa em valor lógico (falso, verdadeiro).

Exemplos:

- a < b
- “alessandra” < “alessandro”
- c >= (5.2 – a)

Expressões lógicas ou booleanas

- É formada por uma ou mais proposições. Uma **proposição** é qualquer **sentença declarativa** que pode ser valorada com o valor **verdadeiro** ou **falso**.
  - Imperativa: “Multiplique 2 por 3.”
  - Exclamativa: “Que cerveja gelada!”
  - Interrogativa: “Está chovendo lá fora?”
  - Declarativa: “Todo aluno da UFRN é maior de idade.”

Expressões lógicas ou booleanas

(continuação)

- Em lógica, assumimos que as proposições satisfazem dois princípios:

1. **Terceiro excluído**: uma proposição **ou é verdadeira ou é falsa**, isto é, **não existe uma terceira possibilidade**.
2. **Não-contradição**: uma proposição **não pode ser**, **ao mesmo tempo**, **verdadeira** e **falsa**.

Proposições compostas

- A linguagem utilizada em lógica para **representar proposições** e **realizar cálculos sobre elas** foi cuidadosamente desenvolvida para **evitar ambiguidades**.
  - Este não é o caso da língua portuguesa, que nos permite facilmente construir sentenças ambíguas:
    - Grandes carros e aviões.
      - O que é grande? Só os carros ou carros e aviões?
    - José está vendo o jogo em cima das dunas.
      - Quem está em cima das dunas? O jogo? José? Ambos?

Conectivos lógicos

- Utilizam-se **conectivos lógicos** para criar **novas proposições**, proposições compostas, a partir de **proposições existentes**.

Conectivo	Operador	Descrição	Prioridade
¬	**não**	negação	mais alta
∧	**e**	conjunção	↑
∨	**ou**	disjunção	mais baixa

Tabelas verdades

- Os valores verdades das proposições podem ser descritos através de tabelas verdades.
- A **tabela verdade** de uma *proposição* ***p*** definida a partir das proposições $p_1, \ldots, p_n$ lista todas as possibilidades de valores lógicos de $p_1, \ldots, p_n$, com **V** denotando **verdadeiro** e **F** denotando **falso**, e para cada combinação desses valores lógicos, a tabela verdade lista o valor lógico de ***p***.

Tabela verdade da conjunção (e)

- O valor do resultado da conjunção de relações será verdadeiro se, e somente se, o resultado das duas relações for o valor verdadeiro.

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Tabela verdade da disjunção (ou)

- O valor do resultado da disjunção de relações será verdadeiro se, e somente se, o resultado de uma ou de ambas as relações for o valor verdadeiro.

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Tabela verdade da negação (não)

- O valor do resultado da negação de avaliação da expressão lógica será verdadeiro se, e somente se, o resultado da avaliação da relação for o valor falso.

p	$\overline{p}$
V	F
F	V

Prioridade de operadores aritméticos, relacionais e lógicos, unários e binários

Operador	Tipo	Precedência
+ -	unários	1 (máxima)
*** / %**	binários	2
+ -	binários	3
= <> >= <= > <	binários	4
nao	unário	5
e	binário	6
ou	binário	7 (mínima)

Exercícios propostos

1. Avalie as seguintes expressões lógicas:

a) não (7 <> 15 / 2) ou verdadeiro e (4 - 6 > 4 - 20)

b) (2 * 5 > 3) ou (5 + 1 < 2) e (2 < 7 - 2)

c) (2 > 3) e (3 < 2) ou (2 < 3)

d) não (8 * 2 > 10) ou (8 - 3 <> 4) ou (3 > 8 * 4)