Revisão e Discussão da Norma ISO 5807 - 1985 (E) Proposta para Padronização Formal da Representação Gráfica da Linha de Raciocínio Lógico Utilizada no Desenvolvimento da Programação de Computadores a ser Definida no Brasil.

JOSÉ AUGUSTO NAVARRO GARCIA MANZANO

Professor na área de Tecnologia da Informação (TI) leciona as disciplinas de Algoritmos, Lógica de Programação, Estrutura de Dados, Linguagens de Programação (PASCAL, C, C++, JavaScript, entre outras), Tópicos Avançados de Processamento de Dados, Engenharia de Software, Arquitetura de Computadores, Engenharia da Informação. Atua profissionalmente nas instituições de ensino superior: Faculdade Cantareira (São Paulo – SP), Faculdades Interação Americana (São Bernardo do Campo – SP) e Universidade Mogi das Cruzes (São Paulo – SP). Possui mestrado em Administração de Empresas e é autor de vários títulos na área de TI publicados pela Editora Érica Ltda.
Email : augusto_manzano@hotmail.com

RESUMO

Este texto apresenta uma proposta de modelo gráfico para a representação da linha de raciocínio lógico do conceito de programação de computadores. Tem por objetivo discutir, revisar e ampliar alguns detalhes existentes na norma ISO 5807-1985 (E), que não são apresentados de forma aprofundada.

ABSTRACT

This paper presents a graphic model representing the computer logical reasoning of programming concept. Its objective is to discuss, revise and enlarge some existent details in the norm ISO 5807-1985 (E) that are not presented in a deep way.

PALAVRAS-CHAVE : fluxograma, diagrama de bloco, diagrama de blocos, , diagramas de blocos, diagrama de fluxo de programa, lógica de programação de computadores, programação de computadores.

INTRODUÇÃO

Este artigo tem por objetivo discutir e propor um formato padronizado para a elaboração e documentação da representação gráfica do processo de desenvolvimento lógico de programação de computadores para linguagens de programação de computadores de primeira, segunda e terceira gerações, tomando-se

por base os parâmetros apresentados na norma ISO[1] 5807-1985 (E): Information processing - Documentation symbols and conventios for data, program and system flowcharts, programa network charts and system resources charts (Processamento de Informação – Documentação de símbolos e convenção de dados, diagramas de fluxo de programa e sistemas, diagramas de programas de redes e diagramas de recursos de sistemas).

Deseja-se com este trabalho sugerir algumas regras (não definidas na referida norma) que venham a formalizar e facilitar o processo de documentação gráfica da representação da linha de raciocínio lógico utilizada na elaboração da programação de computadores, segundo à luz da norma ISO 5807-1985 (E). Neste trabalho, não se questiona o fato de a norma estar correta ou incorreta (ou mesmo desatualizada), mas, sim, o fato de apresentar alguns pontos a serem considerados pelos profissionais da área de desenvolvimento de *software*, principalmente pelos programadores de computador.

O fator que incentivou a elaboração desta proposta de trabalho decorre de dois pontos básicos: o primeiro pela ocorrência do fato de a última revisão da norma ISO 5807 ter ocorrido em 1985 e não ter sido mais reavaliada; o segundo pelo fato de apesar de, existir uma norma internacional, esta se apresenta de forma bastante genérica e não foca de maneira clara os critérios para a elaboração de diagramas que representem a linha de raciocínio lógico a ser utilizada por um programador de computadores. Assim sendo, o escopo de discussão deste artigo está focado nos pontos relacionados à elaboração dos diagramas *program flowcharts* (diagramas de fluxo de programa) e *data flowcharts* (diagrama de fluxo de dados), que serão utilizados em conjunto para a representação gráfica da linha de raciocínio a ser utilizada na elaboração da programação de computadores.

Os demais pontos existentes e comentados na norma ISO 5807-1985 (E), tais como: *system flowcharts* (diagrama de fluxo de sistema) *program network charts*

[1] International Organization for Standardization.

(diagrama de programas de rede) e *system resources charts* (diagrama de recursos de sistema) não serão aqui discutidos, por fugirem do escopo pretendido, deixando-os como uma proposta para um estudo mais aprofundado a ser realizado.

Sendo assim, neste artigo são apresentados pontos relacionados às técnicas de diagramação do processo de representação gráfica da lógica de programação de computadores no que tange o uso dos conceitos de: seqüência; tomada de decisão simples, composta e seleção; utilização de laços de repetição; uso dos conceitos de procedimentos / funções com ou sem passagens de parâmetro, tanto por valor quanto por referência; uso de operações com arquivos.

Espera-se com esta proposta fornecer aos profissionais da área de desenvolvimento de programas de computadores subsídios para a elaboração padronizada e formal da representação gráfica da lógica de programação de computadores.

HISTÓRICO

A norma ISO 5807-1985 (E) para a definição e elaboração de diagramas de fluxos para a área de desenvolvimento e projeto de software é a consolidação de duas normas anteriores: ISO 1028[2] e ISO 2636[3] (ISO 5807, 1985, p. 1), ambas publicadas em 1973. Em particular, a norma ISO 1028 foi editada a partir da norma ANSI[4] X3.5[5] de 1970.

Segundo informação da própria norma ISO 5807-1985 (E), a definição e elaboração de diagramas para a representação gráfica da linha de raciocino lógico a ser adotada não deve restringir o uso de aplicações ou soluções particulares, uma vez que pode ocorrer a existência de vários tipos de soluções para os vários problemas de processamento de informação (ISO 5807, 1985, p. 1). Nota-se, com base no exposto,

[2] Information processing - Flowchart symbols.
[3] Information processing - Conventions for incorporating flowchart symbols in flowcharts.
[4] American National Standards.

que a norma ISO 5807-1985 (E) sugere o uso de alguns critérios que devem ser adaptados segundo as necessidades existentes. Se de um lado, esta postura fornece a liberdade de trabalho desejada, por outro, acaba dando margem à existência de problemas de interpretação da própria norma.

Dentro deste aspecto e da liberdade de poder definir ou considerar situações particulares, é que este artigo propõe alguns padrões básicos na elaboração da representação gráfica, no sentido de que haja uma maior conformidade na sua elaboração e maior rigor na definição padronizada de sua apresentação visual. Dessa forma, ficará mais fácil a comunicação na elaboração de projetos, nos quais várias pessoas estarão envolvidas, mesmo que sejam de regiões geográficas diferentes.

Ë importante ressaltar que a norma ISO 5807-1985 (E) faz menção à definição de cinco tipos de representação gráfica: *program flowcharts* (diagrama de fluxo de programas), *data flowcharts* (diagrama de fluxo de dados), *system flowcharts* (diagrama de fluxo de sistemas) *program network charts* (diagrama de programas de rede) e *system resources charts* (diagrama de recursos de sistema) não serão discutidos.

Para a proposta deste artigo utilizo-se como base de estudo o uso de apenas dois dos cinco modelos propostos, a saber: *program flowcharts* (diagramas de fluxo de programa) e *data flowcharts* (diagrama de fluxo de dados). Os demais modelos não serão discutidos por estarem fora do contexto pretendido.

Os símbolos gráficos utilizados com os *program flowcharts* e *data flowcharts* possuem como característica demonstrar de forma clara a linha de raciocínio lógico utilizada por um programador de computadores, de forma que seja fácil a quem não efetuou a tarefa de programação entender o que se pretende que aquele programa faça.

Segundo (PRESSMAN, 1995, p. 453), "um quadro[6] vale mil palavras, mas é importante diferenciar qual quadro e quais mil palavras" se pretende realmente

[5] Flowchart Symbols and their Usage in Information Processing.
[6] O termo *quadro* está sendo utilizado no sentido figurado para representar o termo *símbolo.*

referenciar. A importância da representação gráfica da linha de raciocínio lógico é considerada também por BERG & FIGUEIRÓ (1998, p. 18), quando afirmam que as representações gráficas implicam ações distintas, deixando claro que "tal propriedade facilita o entendimento das idéias (...) e justifica a sua popularidade". A importância da representação gráfica da lógica de programação de um computador não é algo novo, pois já havia sido apresentada por GOLDSTEIN & VON NEUMANN (1947).

Verifica-se, que apesar da concordância de alguns autores no sentido de reconhecer o valor do instrumento gráfico na representação da linha de raciocínio lógico de um programa, não existe uma definição padronizada para seu desenvolvimento, o que acaba por gerar diversas outras formas de definição e leva a incorrer em muitos erros de definição. PRESSMAN (1995, p. 453 adverte que "se as ferramentas gráficas forem mal utilizadas, a figura errada pode levar ao *software* errado".

Parte da ocorrência da má interpretação e uso das ferramentas de representação gráfica decorrem do fato de a norma ISO 5807-1985 (E) ser um instrumento apenas informativo e não regulador do processo de criação e uso dos símbolos gráficos nos projetos de desenvolvimento da lógica de programação a ser utilizada.

DEFINIÇÃO CONCEITUAL

Um dos pontos que pode gerar conflito de definição entre os profissionais de desenvolvimento de *software* no nível de programação de computadores e no nível de análise de sistemas é o fato de utilizar o termo fluxograma para generalizar o uso de toda e qualquer forma de representação gráfica de atividades relacionadas ao projeto de *software.*

Esta generalização decorre, talvez, do fato de o termo *flowchart* ter sido traduzido para o idioma português como *fluxograma.* Nesse sentido, tem-se como *flow* o conceito mais puro associado ao termo *fluxo* e *chart* o conceito mais puro associado

ao termo *diagrama*. Dessa forma, a tradução mais adequada para o termo *flowchart*[7] seria *diagrama de fluxo*. Assim sendo, a definição na norma ISO 5807-1985 (E) de *flowcharts program* e *data flowcharts* poderia ser respectivamente em idioma português traduzida como: *diagrama de fluxo de programas* e *diagrama de fluxo de dados*.

No sentido de distanciar a definição do termo *fluxograma* do termo *diagrama de fluxo de programas* e *diagrama de fluxo de dados* e assim evitar conceituações equivocadas, fica estabelecido o termo *diagrama de blocos*, aceito e apontado por PRESSMAN (1998, p. 453).

O conceito do termo *bloco* do ponto de vista da programação de computadores é bem definido segundo GUIMARÃES & LAGES (1994, p. 22), quando afirmam que "um bloco pode ser definido como um conjunto de comandos com uma função bem definida".

Segundo definição da norma ISO 5807-1985 (E), os conceitos de *diagrama de fluxo de programas* (*flowcharts program*) e *diagrama de fluxo de dados* (*data flowcharts*) podem ser, resumidamente, entendidos como:

- **Data flowcharts** – representação gráfica do formato da estrutura de dados a ser utilizada na solução de um determinado problema (ISO 5807, 1985, p. 1);
- **Program flowcharts** – representação gráfica da seqüência de operações executada por um programa de computador (ISO 5807, 1985, p. 2).

No contexto prático das atividades de documentação gráfica da linha de raciocínio lógica de um programa, ocorre o fato de parte dos símbolos existentes para cada categoria de diagramas serem naturalmente utilizados em conjunto. Dessa forma, fica um pouco difícil no projeto lógico de um programa de computador

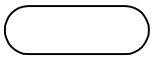
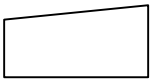
[7]Um sinônimo para o termo *flowchart* em inglês é o termo *flow diagram* (http://www.its.bldrdoc.gov/fs-1037 – Federal Standard 1037C – National Communications System).

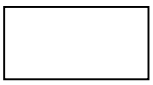
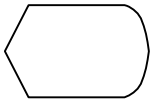
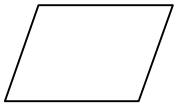
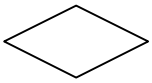
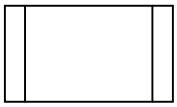
diferenciar as categorias de símbolos em separado, pois, além de estabelecer uma seqüência de operações a serem executadas (*flowcharts program*), um programa necessita efetuar o tratamento dos dados de entrada e saída (*data flowcharts*).

É neste enfoque operacional que este artigo visa a apresentar e sugerir uma conduta de documentação mais rígida e definida para a elaboração de *diagramas de blocos* direcionados para representar a linha de raciocínio lógico na elaboração de programas de computadores.

SIMBOLOGIA GRÁFICA

Na figura 1 são apresentados apenas os símbolos mais importantes que representam a seqüência de operações de um programa (*program flowcharts*) e o de acesso ao caminho dos dados (*data flowcharts*) na definição da estrutura gráfica de um diagrama de blocos. É com relação à utilização destes símbolos que está sendo feita esta contribuição.

Símbolo	Significado	Descrição
	Terminal *Terminato*r	Este símbolo representa a definição de início e fim do fluxo lógico de um programa (ISO 5807, 1985, p. 9). Também é utilizado na definição de sub-rotinas de procedimento ou função.
	Entrada manual *Manual input*	Este símbolo representa a entrada manual de dados, normalmente efetuada em um teclado conectado diretamente ao console do computador (ISO 5807, 1985, p. 3).

Símbolo	Significado	Descrição
	Processamento *Process*	Este símbolo representa a execução de uma operação ou grupo de operações que estabelecem o resultado de uma operação lógica ou matemática (ISO 5807, 1985, p. 3).
	Exibição *Display*	Este símbolo representa a execução da operação de saída visual de dados em um monitor de vídeo conectado ao console do computador (ISO 5807, 1985, p. 3).
	Documento *Document*	Este símbolo representa a execução da operação de saída de dados em um documento emitido por uma impressora na forma de relatório (ISO 5807, 1985, p. 3).
	Dados *Data*	Este símbolo representa oficialmente dados de uma forma genérica (ISO 5807, 1985, p. 2). Tipicamente é associado a operações genéricas de entrada e saída de dados, desde que identificados.
	Decisão *Decision*	Este símbolo representa o uso de desvios condicionais para outros pontos do programa de acordo com situações variáveis (ISO 5807, 1985, p. 4).
	Preparação *Preparation*	Este símbolo representa a modificação de instruções ou grupo de instruções existentes em relação à ação de sua atividade subseqüencial (ISO 5807, 1985, p. 4).
	Processo predefinido *Predefined process*	Este símbolo representa definição de um grupo de operações estabelecidas como uma sub-rotina de processamento anexa ao diagrama de blocos (ISO 5807, 1985, p. 4).
	Conector *Connector*	Este símbolo representa a entrada ou saída em outra parte do diagrama de blocos. Pode ser usado na definição de quebras de linha e na continuação da execução de decisões (ISO 5807, 1985, p. 9).

Símbolo	Significado	Descrição
----------	Linha tracejada *Dashed line*	Este símbolo representa uma alternativa na definição do relacionamento entre duas operações do diagrama de blocos. Também pode ser usado na definição de área de comentários (ISO 5807, 1985, p. 7).
————	Linha *Line*	Este símbolo representa a ação de vínculo existente entre os vários símbolos de um diagrama de blocos. Normalmente possui a ponta de uma seta indicando a direção do fluxo de ação (ISO 5807, 1985, p. 6).

Figura 1 – Tabela de símbolos para a definição de diagramas de blocos - Fonte: norma ISO 5807-1985 (E).

Os símbolos: dados, processamento, preparação, linha tracejada, linha, conector são categorizados segundo a norma ISO 5807-1985 (E) como símbolos para a utilização em diagramas de fluxo de programas e diagrama de fluxo de dados. Já os símbolos: armazém de dados, documento, entrada manual, exibição são utilizados apenas para a representação do fluxo de dados do programa. Os símbolos processo predefinido e decisão são utilizados para a representação do fluxo lógico de um programa. É da junção dessas duas categorias de símbolos que se define a criação e uso de um diagrama de blocos.

É importante ressaltar que para a definição do desenho de um diagrama de blocos pode-se utilizar um gabarito (figura 2) que normalmente possui o conjunto de símbolos básicos necessários para o devido uso. Os gabaritos possuem também, dependendo do modelo, alguns símbolos que não estão definidos na norma ISO 5807-1985 (E) ou, muitas vezes, não apresentam símbolos que estão definidos na norma ISO 5807-1985 (E).

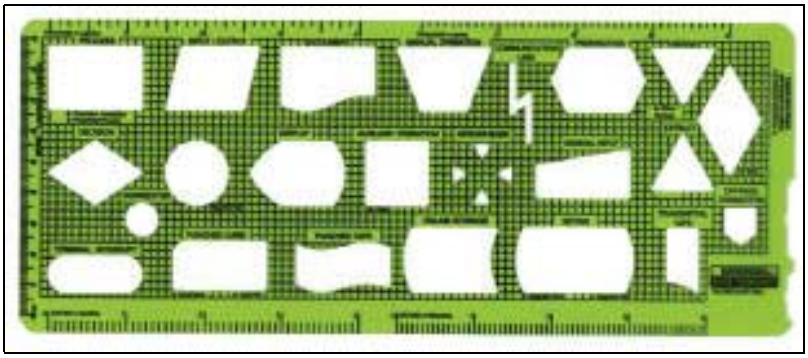
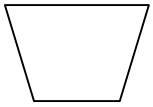


Figura 2 – Gabarito para o desenvolvimento de diagramas de blocos - Fonte: Rapidesign.

Para melhor ilustrar esta abordagem, serão apresentados os demais símbolos existentes no gabarito representado pela figura 2, os quais não foram comentados anteriormente, mas que são previstos de uso, segundo a norma ISO 5807-1985 (E). A figura 3 mostra os referidos símbolos.

A figura 2 possui alguns símbolos que não foram apresentados nas figuras 1 e 3, pois os mesmos não se encontram definidos na norma ISO 5807-1985 (E). Vale ressaltar um pequeno detalhe com relação ao símbolo *processo predefinido*, que para ser desenhado utiliza como base o símbolo *processamento*.

Símbolo	Significado	Descrição
	Operação manual *Manual operation*	Este símbolo representa a execução de qualquer operação que poça ser realizada por seres humanos, e que não estejam previstas em nenhum outro símbolo (ISO 5807, 1985, p. 4).

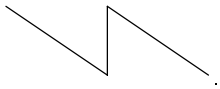
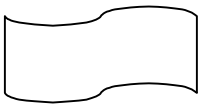
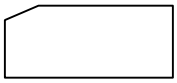
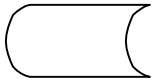
Símbolo	Significado	Descrição
	Linha de comunicação *Communication link*	Este símbolo representa a transferência automática de informações ou dados entre locais diferentes, por meio de linhas de comunicação (ISO 5807, 1985, p. 7).
	Fita perfurada *Punched tape*	Este símbolo representa o uso de fita de papel ou fita plástica utilizada como um meio de armazenamento de programa e dados. Mecanismo de armazenagem de pouco uso (ISO 5807, 1985, p. 3).
	Cartão *Card*	Este símbolo representa o uso de cartões perfurados de cartolina utilizado como meio de armazenamento de programa e dados. Mecanismo de armazenagem de pouco uso (ISO 5807, 1985, p. 3).
	Armazém de dados *Stored data*	Este símbolo representa a definição de acesso de abertura, fechamento, leitura e escrita em um determinado arquivo de dados a ser realizada por um programa (ISO 5807, 1985, p. 2).

Figura 3 – Tabela de símbolos para a definição de diagramas de blocos - Fonte: norma ISO 5807-1985 (E).

SUGESTÃO DE REGRAS PARA COMPLEMENTAÇÃO DA NORMA ISO 5807-1985 (E)

A norma ISO 5807-1985 (E) apresenta alguns parcos exemplos de representações gráficas da forma de uso das cinco categorias de diagrama que aborda. Nenhum dos exemplos indicados aprofunda-se em um uso mais formal dos símbolos. Com base nessa ocorrência, abre-se o espaço para a contribuição deste artigo.

Antes de exemplificar o uso específico de algumas situações relacionadas ao processo de representação gráfica da linha de raciocínio lógico de um programa de computador, é necessário considerar algumas convenções, baseada na própria norma ISO 5807-1985 (E), bem como por ela daptada:

- os símbolos de identificação gráfica representam sempre uma operação ou conjunto de operações similares, podendo ser identificados por um rótulo relacionado à própria ação do símbolo em uso, quando necessário;

- os símbolos devem ser conectados uns aos outros por linhas de setas que mostrem explicitamente a direção do fluxo de programa a ser utilizado;

- a estrutura visual do diagrama de blocos deve a princípio estar orientada no sentido de cima para baixo, da direita para a esquerda. No entanto, dependendo da situação este critério pode ser alterado, o que leva a necessidade de manter-se as linhas de seta indicando a direção do fluxo;

- a definição de inicialização e finalização de um diagrama de bloco ocorrerá com o uso do símbolo “terminal”, devidamente identificado;

- as operações de entrada de dados executadas por um ser humano serão genericamente representadas com o símbolo “entrada manual”;

- as operações de saída de dados serão genericamente definidas com o símbolo “exibição”, considerando o fato de estar em uso um monitor de vídeo. Exceção a esta regra ocorrerá quando da definição de saída de dados em formato de relatório, definidas pelo símbolo “documento”;

- as operações de entrada e saída não definidas com os símbolos “entrada manual”, “exibição” e “documento” serão definidas com o símbolo “dados”;

- a definição das variáveis nas operações de entrada e saída serão definidas nos símbolos apropriados. Quando houver mais de uma variável a ser utilizada, estas estarão definidas por vírgulas;

- as operações de processamento matemático e lógico estarão definidas com o uso do símbolo “processamento”. Quando houver mais de uma operação a ser definida em um mesmo símbolo, estas deverão estar separadas por linhas de ação;
- as operações de tomada de decisão para decisões simples, decisões compostas e laços condicionais serão representadas pelo símbolo “decisão”, que conterá internamente a definição da condição a ser avaliada logicamente. Cada símbolo de decisão poderá possuir a definição de apenas uma condição lógica. É considerada uma condição lógica a definição de uma condição ou de um conjunto de condições vinculadas por um operador lógico de conjunção (conceito e) ou disjunção (conceito ou);
- as operações de laços incondicionais (laços iterativos) serão representadas com o símbolo “preparação”, uma vez que este símbolo permite a realização de um grupo de tarefas relacionadas;
- em relação à definição de sub-rotinas será utilizado o símbolo “processo predefinido”. Este símbolo deverá estar identificado com um rótulo que estará associado a outro diagrama de bloco interdependente ao programa e relacionado por um símbolo “terminal”;
- as operações que necessitarem do uso de conexão utilizarão o símbolo “conexão”. Este símbolo poderá ser usado na finalização de operações de decisões ou na identificação de vínculos entre partes de um programa e, nesse caso, deverão estar identificados com rótulos alfanuméricos;
- fica eleito o símbolo “processamento” como um curinga, podendo representar qualquer ação lógica definida ou não, desde que a operação seja devidamente identificada por um rótulo descritivo. Exceções aos símbolos “decisão” e “preparação” que representam operações bem definidas e não serão substituídos por qualquer outro símbolo.

Além das sugestões anteriormente propostas, poderão surgir ainda alguns pontos a serem definidos. Demais critérios que se fizerem necessários e que não foram aqui sugeridos deverão ser analisados e definidos em particular pela gerência de cada equipe de desenvolvimento.

REPRESENTAÇÕES GRÁFICAS

Serão, a seguir, apresentadas algumas situações ilustrativas em relação ao uso do instrumento gráfico, de forma que possa vir a facilitar a montagem visual do processo da linha de raciocínio da lógica de programação a ser utilizada.

Não é foco deste artigo discutir a funcionalidade do princípio de execução lógica de um computador e de seus detalhes de operação. Objetiva-se apenas demonstrar como representar graficamente as ações lógicas de um computador na forma mais abrangente possível, de maneira que se possa utilizar a técnica de diagramação com qualquer tipo de linguagem de programação de computadores.

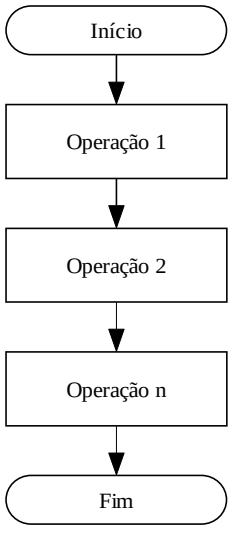


Figura 4 – Representação de uma seqüência simples de operações.

A figura 4 apresenta um modelo de execução seqüencial de passos a serem processados por um computador. Nota-se que para esta representação gráfica utiliza-

se o símbolo *terminal* com a identificação da ação de início e fim da ação de um programa de computador. O símbolo *processamento* é usado na figura 4 ocorre de forma genérica representando qualquer tipo de operação.

A identificação da operação de início e fim de um diagrama de bloco é obrigatória, pois o símbolo *terminal* também é utilizado para representar o início e o fim da execução de uma sub-rotina. Assim sendo, torna-se necessário identificar corretamente a sua ação. A identificação utilizada no símbolo *terminal* ou demais símbolos poderá ocorrer em qualquer idioma.

Outro ponto a ser considerado é a definição das linhas de ligação representadas por setas, as quais possuem a finalidade de indicar a direção que o fluxo do programa deverá seguir, além de estabelecer o vínculo de ação entre as várias operações que um programa de computador deve executar.

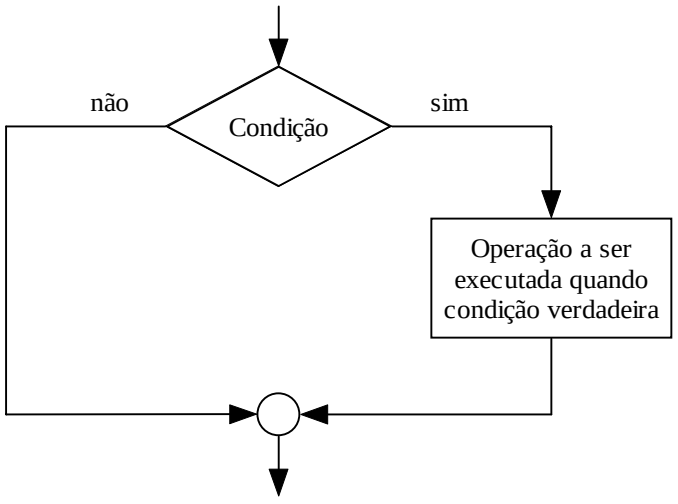


Figura 5 – Representação de uma tomada de decisão simples.

A figura 5 caracteriza a ação de uma tomada de decisão simples. Nesse caso, quando a condição for verdadeira, ocorrerá o desvio do fluxo do programa para a ação identificada como verdadeira e posterior ação de encontro com o símbolo *conector*. No caso de a condição ser falsa, o desvio do fluxo do programa ocorrerá pelo lado identificado como “não”, o qual automaticamente irá ao encontro do símbolo

conector para que o fluxo do programa tenha continuidade. Observe o uso do símbolo *conector* para estabelecer o ponto de continuidade do fluxo do programa.

Torna-se fundamental em operações de tomada de decisão simples ou composta identificar qual o lado do diagrama representa a ação verdadeira e a ação falsa. Isto se faz necessário, uma vez que não existe um lado predefinido para a ação verdadeira ou falsa.

A figura 6 caracteriza a ação de uma tomada de decisão composta. Nesse caso, quando a condição for verdadeira ocorrerá o desvio do fluxo do programa para a ação identificada como verdadeira. Caso a condição seja falsa, será executada a ação identificada como falsa.

Torna-se fundamental em operações de tomada de decisão simples ou composta identificar qual o lado do diagrama representa a ação verdadeira e a ação falsa. Isso se faz necessário, uma vez que não existe um lado predefinido para a ação verdadeira ou falsa.

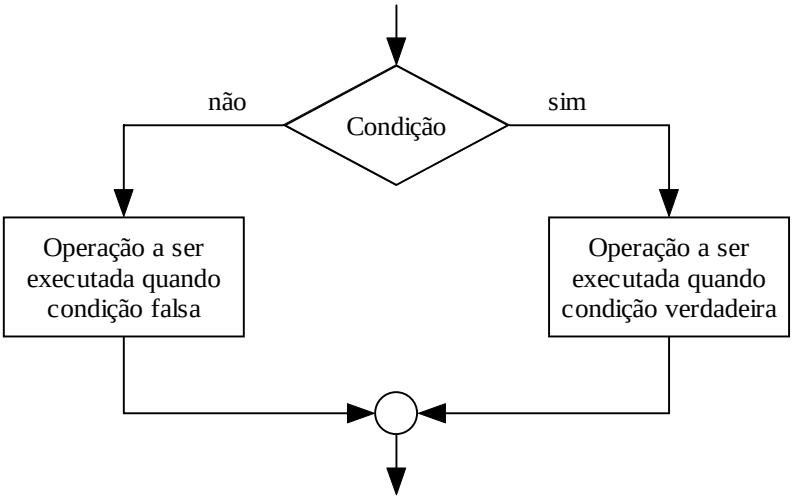


Figura 6 – Representação de uma tomada de decisão composta.

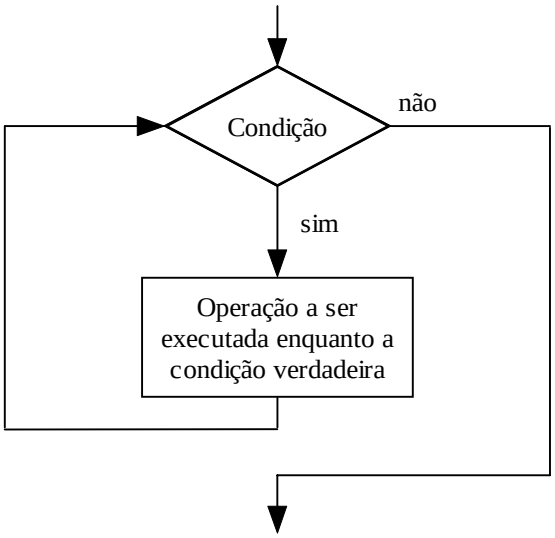


Figura 7 – Representação de laço condicional do tipo enquanto...faça.

A figura 7 apresenta o formato do laço de repetição condicional do tipo *enquanto...faça.* Este tipo de laço executa as operações a ele subordinadas enquanto a condição for verdadeira. Após a condição se tornar falsa, o laço é encerrado e fluxo de ação do programa continua sua execução fora do laço de repetição.

A figura 8 apresenta o formato do laço de repetição condicional do tipo *repita...até que.* Esse tipo de laço executa as operações a ele subordinadas até que a condição seja verdadeira. Após a condição se tornar verdadeira, o laço é encerrado e o fluxo de ação do programa continua sua execução fora do laço de repetição.

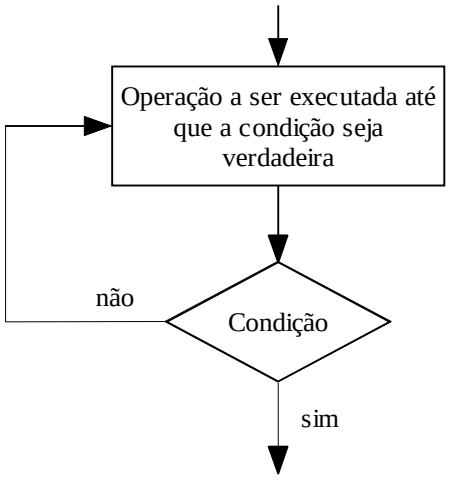


Figura 8 – Representação de laço condicional do tipo repita...até que.

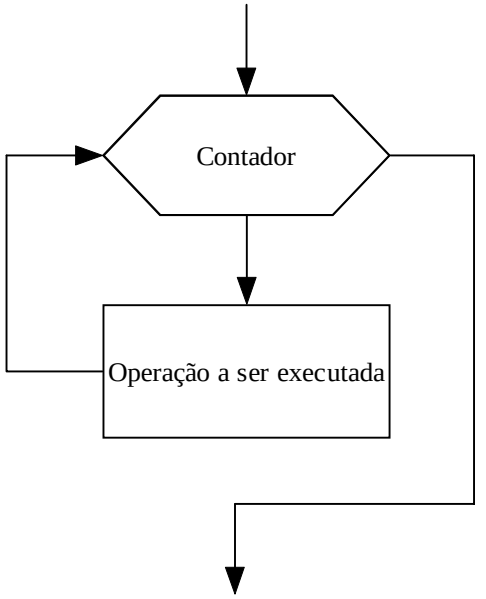


Figura 9 – Representação de laço incondicional do tipo para.

A figura 9 apresenta o formato do laço de repetição incondicional do tipo *para*. Este tipo de laço executa as operações a ele subordinadas por uma variável de controle iniciada por um valor, finalizado por outro valor diferente do valor inicial, podendo ser maior ou menor. A operação de incremento ou decremento é efetuada por um contador de passo. Supondo um contador crescente de 1 a 10 com passo 1 positivo, sugere-se escrever na parte interna do símbolo *preparação* a descrição I ← 1, 10, 1; no qual I é a variável contador e as demais informações são respectivamente a inicialização da contagem, a finalização da contagem e o incremento entre o início e o fim da contagem. Se for utilizado um contador decrescente de 10 a 1 com passo 1 negativo sugere-se utilizar a descrição I ← 10, 1, -1.

As figuras 7, 8 e 9 sugerem uma forma de representação de laços de repetição mais intimamente focada no contexto lógico da execução do código de programa em um computador. No entanto, na norma ISO 5807-1985 (E), é sugerido o uso da estrutura gráfica representada pela figura 10.

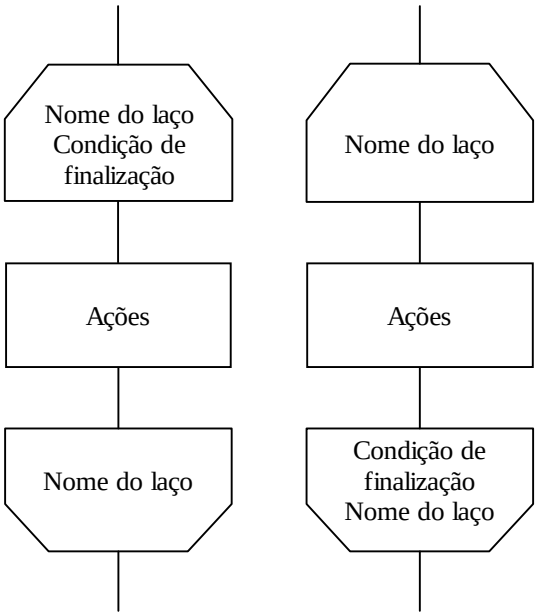


Figura 10 – Estrutura de laço, segundo a norma ISO 5807-1985 (E), pg 6.

A forma representada na figura 10 determina o uso da forma gráfica de representação dos laços do tipo *enquanto* ou tipo *para*[8] (desenho da esquerda) e do tipo *repita* (desenho da direita). Nota-se apenas uma forma de representação gráfica sem o real vínculo com a representação do funcionamento mecânico dos tipos de laços. Representar os laços na forma mecânica, possibilita uma definição focada na funcionalidade do laço e não em uma mera representação do laço em si.

A figura 11 demonstra o critério de definição da chamada de uma sub-rotina com o uso do símbolo *processo predefinido*, na qual é indicado o nome da sub-rotina no programa principal. Ao lado direito encontra-se a definição da rotina secundária. Note as identificações utilizadas nos símbolos de *terminal*.

A figura 12 demonstra o critério de definição da chamada de uma sub-rotina com o uso do conceito de passagem de parâmetro por valor, representado pelos dados indicados entre parênteses. Note no diagrama da direita a identificação no

[8] Para representar um laço do tipo *para* é necessário considerar no lugar da condição a definição de um contador.

símbolo de *terminal* do nome da sub-rotina e dos parâmetros definidos entre parênteses.

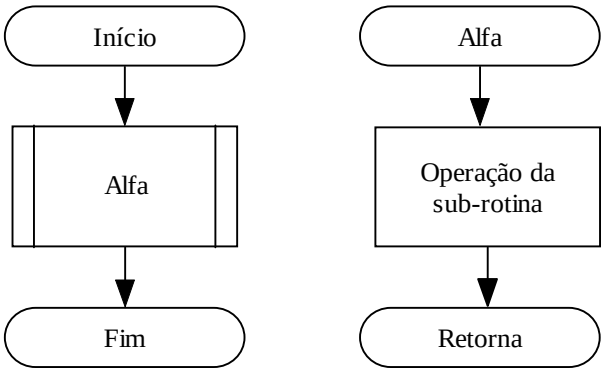


Figura 11 – Chamada e definição de uma sub-rotina simples.

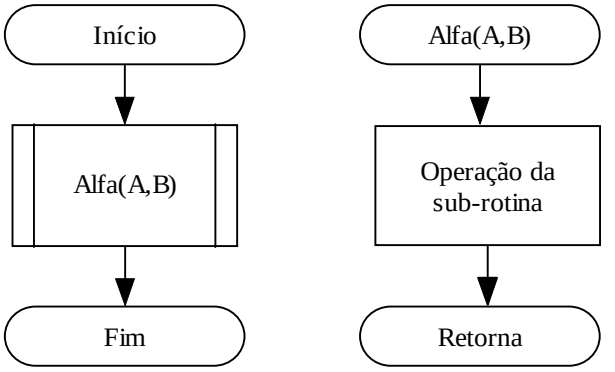


Figura 12 – Chamada e definição de uma sub-rotina com passagem de parâmetro por valor.

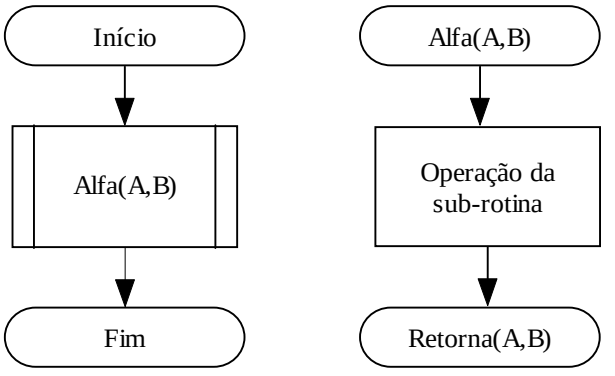


Figura 13 – Chamada e definição de uma sub-rotina com passagem de parâmetro por referência.

A figura 13 demonstra o critério de definição da chamada de uma sub-rotina com o uso do conceito de passagem de parâmetro por referência. Note no diagrama da direita, além da identificação no símbolo de *terminal* do nome da sub-rotina e dos

parâmetros definidos entre parênteses, a indicação da definição do rótulo Retorna(A,B), demonstrando a passagem de parâmetro por referência.

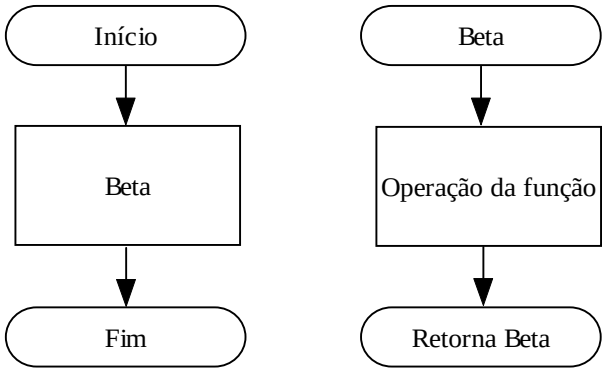


Figura 14 – Chamada e definição de uma função simples.

A figura 14 demonstra o critério de definição da chamada de uma sub-rotina de função, na qual é indicado o nome da função numa operação de processamento (por exemplo) e ao lado direito se encontra a definição da rotina secundária. Note as identificações utilizadas nos símbolos de *terminal*. Perceba que, ao indicar a finalização de uma função, utiliza-se junto do rótulo de identificação Retorna o nome da própria função.

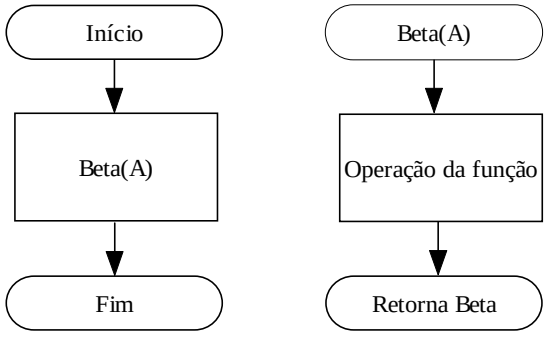


Figura 15 – Chamada e definição de uma função com passagem de parâmetro por valor.

A figura 15 demonstra o critério de definição da chamada de uma função com o uso do conceito de passagem de parâmetro por valor, representado pelo dado

indicado entre parênteses. Note no diagrama da direita a identificação no símbolo de *terminal* do nome da função e do parâmetro definidos entre parênteses.

A figura 16 demonstra o critério de definição da chamada de uma função com o uso do conceito de passagem de parâmetro por referência, representado pelo dado indicado entre parênteses. Note no diagrama da direita a identificação no símbolo de *terminal* do nome da função e do parâmetro definido entre parêntese, indicando a passagem de parâmetro por referência.

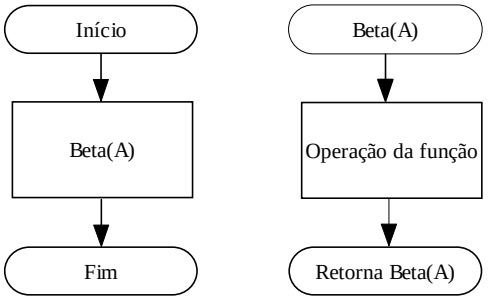


Figura 16 – Chamada e definição de uma função com passagem de parâmetro por referência.

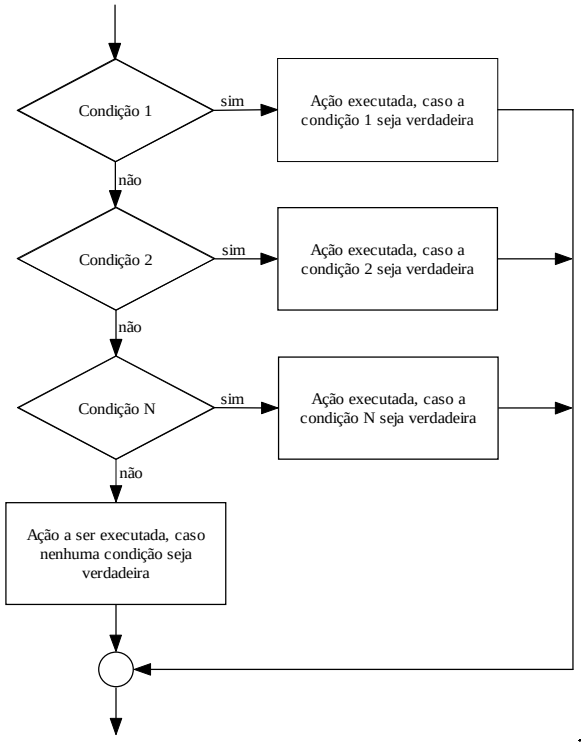


Figura 17 – Representação da estrutura de seleção case.

A figura 17 representa visualmente o funcionamento lógico da estrutura de controle lógico *case* existente em várias linguagens de programação de computadores. Esse tipo de estrutura lógica tem por finalidade desviar o fluxo de funcionamento do programa para uma ação e em seguida desviar o fluxo do programa para um ponto de conexão comum, representado pelo símbolo *conector*. Se a condição é válida (condição verdadeira), a ação definida após a condição é realizada e o fluxo de ação do programa é posicionado após a execução da ação estabelecida no ponto de conexão comum (representado pelo símbolo *conector*), que visa encerrar a ação da estrutura *case*. No caso de uma determinada condição ser falsa, o controle do fluxo de programa é desviado para a condição seguinte, que repetirá o mesmo processo de funcionamento até chegar no ponto de conexão comum.

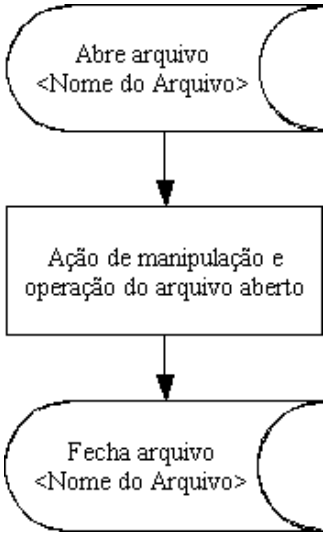


Figura 18 – Representação das operações de abertura e fechamento de arquivos.

A figura 18 demonstra as operações de abertura e fechamento de arquivos de dados, utilizando-se o símbolo *armazém de dados*. Em casos que envolvam a manipulação de mais de um arquivo de dados, utiliza-se sempre um símbolo de abertura e fechamento para cada arquivo em particular.

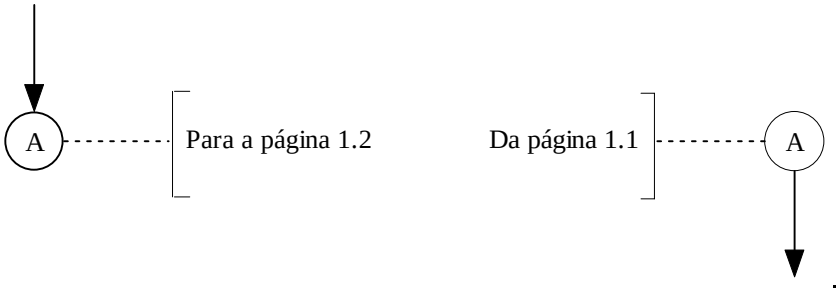


Figura 19 – Representação de indicações de conexão de páginas.

A figura 19 apresenta um exemplo de conexão da linha de raciocínio lógico entre páginas, sugerido na norma ISO 5807-1985 (E), que dispensa qualquer comentário, pois apresenta-se de forma clara e objetiva. Vale ressaltar que a indicação "A" dentro dos conectores é apenas uma referência ao ponto de conexão, podendo ser utilizada para esta representação outras letras e mesmo números. Outro detalhe com o uso de conectores é a possibilidade de representarem conexões na mesma página. Nesse caso, utiliza-se apenas os conectores, suprimindo a linha tracejada e o colchete de discrição do ponto de conexão.

ELEMENTOS AUXILIARES DE REPRESENTAÇÃO GRÁFICA

É muito comum o uso de alguns elementos auxiliares de representação gráfica ao contexto lógico do desenvolvimento da programação de computador, principalmente no que tange ao uso de operações lógicas e matemáticas. Nesse sentido, existem alguns operadores auxiliares, tais como:

- operadores aritméticos;
- operadores relacionais;
- operadores lógicos.

Além dos operadores aritméticos, relacionais e lógicos existentes, há também a definição do uso de valores lógicos, quando do uso de avaliação condicional ou processamento não matemático, tais como:

- verdadeiro;
- falso.

A norma ISO 5807-1985 (E) não faz menção em relação ao uso desses elementos auxiliares de representação gráfica. Assim sendo, cada profissional define critérios particulares, o que, muitas vezes, dificulta a comunicação entre elementos do mesmo grupo, ou em grupos distintos. Nesse sentido, é necessário definir um procedimento padrão para o uso desses elementos de forma mais organizada e padronizada, conforme apresenta-se a seguir.

OPERADORES ARITMÉTICOS

Os operadores aritméticos são responsáveis pela representação das operações matemáticas a serem realizadas numa ação de processamento de cálculo. A tabela seguinte sugere uma forma padrão para a representação de cada operador aritmético a ser utilizado.

Operador	Operação	Operaç./Result.		Resultado
/	Divisão	5 / 2	2,5	Obtém resultado do tipo real
div	Divisão	5 div 2	2	Obtém resultado do tipo inteiro
⊗	Multiplicação	3 ⊗ 4	12	Obtém resultado do tipo real ou inteiro
+	Adição	5 + 7	12	Obtém resultado do tipo real ou inteiro
-	Subtração	8 - 3	5	Obtém resultado do tipo real ou inteiro
↑	Exponenciação	2 ↑ 3	8	Obtém resultado do tipo real ou inteiro
←	Atribuição	X ← 2	2	Representa atribuição matemática

Tomando-se por base a fórmula de Báskara utilizada para efetuar o cálculo da equação de segundo grau, tem-se como sua definição a estrutura seguinte.

$$x = \frac{-b \pm \sqrt{\Delta}}{2a}$$

$$\Delta = b^2 - 4ac$$

A fórmula de Báskara (bem como qualquer outra fórmula matemática) deverá ser representada formalmente para efeito de documentação em um diagrama de bloco num formato de expressões aritmética a ser mencionada no símbolo de processamento, utilizando-se para a definição de cada parte da fórmula uma linha de texto.

DELTA ← B ↑ 2 – 4 ⊗ A ⊗ C
X1 ← (–B + DELTA ↑ (1 / 2)) / (2 ⊗ A)
X2 ← (–B – DELTA ↑ (1 / 2)) / (2 ⊗ A)

Note que, na forma de expressão aritmética, o conceito de x' (x linha) está representado pela definição da variável X1 e o conceito de x" (x duas linhas) está representado pela definição da variável X2.

A indicação da extração da raiz quadrada é definida na forma exponencial, considerando-se como expoente o inverso do índice da raiz. E no sentido de representar a operação de exponenciação está sendo aplicado o símbolo "↑".

O sinal de atribuição "←" representa a ação de transferência do valor de uma operação a variável definida à sua esquerda. Este símbolo é usado em substituição ao símbolo "=", uma vez que o símbolo "=" é usado para definir uma relação lógica, como descrito no item *operadores relacionais*.

Quanto à representação gráfica no símbolo de processamento de um diagrama de bloco, as expressões poderão ser desenhadas separadamente (Figura 20a) ou no mesmo símbolo (Figura 20b), respeitando-se a definição de uma linha de texto para cada expressão aritmética.

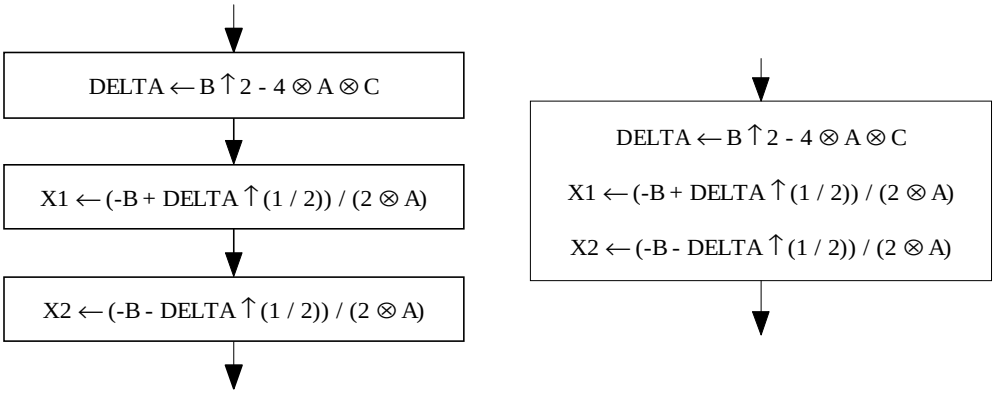


Figura 20 – Forma de representação de uma operação matemática.

OPERADORES RELACIONAIS

Os operadores relacionais são utilizados no estabelecimento das relações lógicas existentes entre dois elementos de uma determinada condição para a efetivação de tomadas de decisão ou da execução de laços de repetição lógicos. A tabela a seguir descreve os operadores existentes:

Símbolo	Significado
=	Igual a
<>	diferente de
>	maior que
<	menor que
>=	maior ou igual que
<=	menor ou igual que

O uso de um operador relacional será sempre definido no símbolo de decisão em um diagrama de blocos, obedecendo à estrutura sugerida na figura 21.

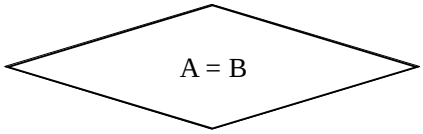


Figura 21 – Representação do uso de operadores relacionais em uma condição.

Os símbolos de representação dos operadores relacionais (=, <>, >, <, >=, <=) devem ser utilizados como descritos anteriormente, independentemente da forma que uma determinada linguagem de programação de computadores faça uso. É importante considerar que a definição de diagramas de blocos deve ocorrer de forma genérica e peculiar. Assim sendo, a codificação de um programa em uma determinada linguagem de programação ocorre em função do modelo da diagramação definida, e não o contrário.

OPERADORES LÓGICOS

Os operadores lógicos de conjunção (operador **.e.**) e de disjunção (operadores **.ou.** e **.oue.** – ou exclusivo) possuem a capacidade de associar para uma avaliação lógica mais de uma condição para uma mesma tomada de decisão. O operador de negação (operador **.não.**) tem por finalidade representar a negação de uma determinada condição. Alguns exemplos de uso são apresentados na figura 22.

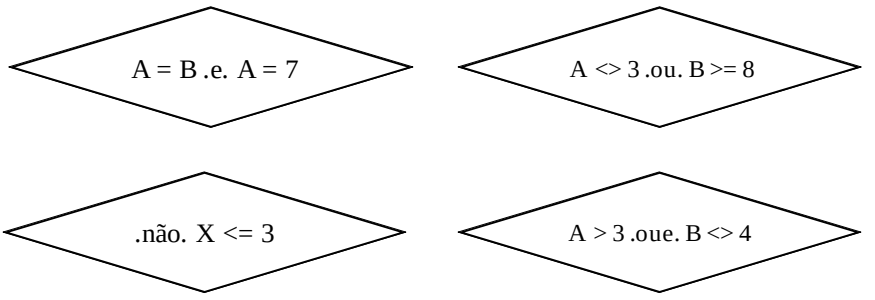


Figura 22 – Exemplos de uso de operadores lógicos.

VALORES LÓGICOS

Operações lógicas resultam em respostas verdadeiras ou falsas dependendo da condição lógica estabelecida. Nesse sentido, os valores lógicos falso e verdadeiro deverão ser expressos, segundo um critério particular:

- valor verdadeiro – representado pelos rótulos **.VERDADEIRO.** ou **.V.**;
- valor falso – representado pelos rótulos **.FALSO.** ou **.F.**

O tipo de valor lógico explícito (**.VERDADEIRO.**, **.V.**, **.FALSO.** ou **.F.**) é normalmente utilizado em operações de processamento para representar a definição de um determinado valor lógico a uma determinada variável. A figura 23 representa uma ação de definição de valor lógico a uma variável.

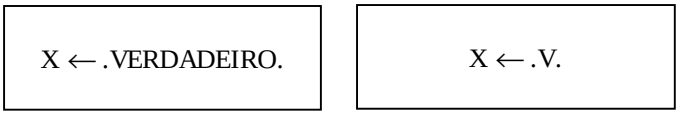


Figura 23 – Definição de valor lógico.

O uso de valores lógicos também poderá ocorrer em situações de verificação de uma condição lógica na execução de uma tomada de decisão ou ação de execução de um laço de repetição. A figura 24 mostra um exemplo de uso de valor lógico em uma condição.

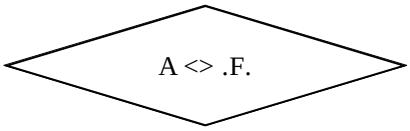


Figura 24 – Definição de valor lógico em uma condição

NOMENCLATURA

Em relação ao uso de diagramas no processo de representação gráfica da linha de raciocínio utilizada por um desenvolvedor, torna-se ideal formalizar uma nomenclatura (já utilizada em parte por alguns poucos profissionais) de identificação dessas estruturas.

A figura 4 representa esquematicamente a estrutura de *diagrama de bloco,* o qual representa o processo seqüencial de ações contínuas existentes entre as definições de início e fim da execução de um programa de computador.

As figuras 5, 6, 7, 8 e 9 representam esquematicamente a estrutura de *diagrama de blocos*, os quais representam a ação da implicação de um desvio condicional (figuras 4, 5, 6 e 7) ou um desvio não condicional (figura 9). Tanto um desvio condicional como um desvio incondicional, efetuam um desvio momentâneo do fluxo central do programa, para que um outro bloco adjacente de ações seja executado. Ao final da execução das ações desse bloco adjacente o programa retorna automaticamente para o fluxo central de funcionamento.

As figuras 11, 12, 13, 14, 15 e 16 representam esquematicamente a estrutura de *diagramas de blocos* (uso mais comum e provável) ou *diagramas de bloco* (uso mais raro, quase impossível). Será considerado uma estrutura de *diagramas de blocos*, quando existir um conjunto de diagramas definidos e relacionados para um mesmo problema computacional, onde esse conjunto envolva a definição de vários *diagrama de blocos*. O conceito *diagramas de bloco* é mais improvável de ser utilizado, pois dificilmente existira um conjunto de diagramas com definições de operações meramente seqüências no estilo apresentado na figura 4.

As formas mais comuns e utilizadas são os *diagrama de bloco* e os *diagramas de blocos*.

BIBLIOGRAFIA

BERG, A. C. , FIGUEIRÓ, J. P. **Lógica de Programação**. Canoas: ULBRA, 1998. 168 p.

PRESSMAN, R. S. **Engenharia de Software**. São Paulo: Makron Books, 1995. 1056 p.

GUIMARÃES, A. de M. , LAGES, N. A. C., **Algoritmos e Estruturas de Dados** 18. ed. Rio de Janeiro: Livros Técnicos e Científicos, 1994, 216 p.

GOLDSTEIN, H. H. & VON NEUMANN, J. **Planning and Coding Problems of an Electronic Computing Instrument**. New York, In A. H. Taub (Eds.), von Neumann, J., 1947, Collected Works, McMillan. pp. 80-151.