Universidade de São Paulo
Instituto de Matemática e Estatística
IME

Trabalho 1 - Os Conceitos da Linguagem Modula-3

Valores, Tipos e Variáveis
MAC316

Integrantes do grupo:
João Antonio Mazzer Mantovani (nº USP 10812022),
Lucas Henrique Bahr Yau (nº USP 9763832),
Patrícia da Silva Rodrigues (nº USP 11315590),
Sabrina Araújo da Silva (nº USP 12566182),
Samantha Miyahira (nº USP 11797261)

Setembro
2022

Conteúdo

1 Tipos Primitivos e Compostos

1.1 Tipos Primitivos

Os tipos primitivos são valores que não podem ser desmembrados, por isso podem ser chamados de "atômicos". Esses tipos de dados primitivos podem ser numéricos, não-numéricos e enumerados. Em Modula-3, os tipos primitivos estão presentes nas seguintes variáveis:

1.1.1 Numéricos

Os tipos inteiros em Modula-3 são tipos ordinais pré-declarados que podem ser classificados como INTEGER (inteiros) e LONGINT (números inteiros com precisão extendida).

Já os tipos reais os quais não podem ser representados por completo são representados respeitando um grau de precisão imposto pelo hardware. Por isso, os números reais são referidos como pontos-flutuantes, que são: REAL, LONGREAL, e EXTENDED.

1.1.2 Não-numéricos e Enumerados

Em Modula-3, os tipos BOOLEAN e CHAR são classificados como enumerados. BOOLEAN é uma enumeração {FALSE, TRUE} e CHAR é uma enumeração que contém pelo menos 256 elementos.

CARDINAL é um tipo que se comporta como um subrange [0..LAST(INTEGER)].

1.2 Tipos Compostos

Os tipos compostos são valores que podem ser desmembrados em valores mais simples. Em Modula-3 temos os seguintes tipos compostos:

1.2.1 RECORD

O tipo RECORD tem a seguinte forma:

TYPE T = RECORD FieldList END

onde *FieldList* é uma lista de campos de declaração em que cada um possui a seguinte forma:

fieldName: Type := default

onde $fieldName$ é um identificador, *Type* é qualquer tipo não vazio que não seja um tipo de matriz aberta e $default$ é uma expressão constante. Os nomes dos campos devem ser distintos. Um registro é membro de T se tiver campos com os nomes e tipos fornecidos, na ordem especificada, e nenhum outro campo. Registros vazios são permitidos.

1.2.2 PACKED

O tipo PACKED é uma forma de declarar a variável por seu tipo que ocorre em RECORDS, OBJECTS ou ARRAYS os quais ocupam exatamente n bits. Este tipo é declarado da seguinte maneira:

TYPE T = BITS n FOR Base

1.2.3 ARRAY

Há dois tipos de arrays, o fixo e o aberto. O tamanho de um array fixo é determinado no tempo de compilação. Por outro lado, o tamanho de uma array aberto é determinado no tempo de execução, quando é alocado.

1.2.4 SET

O tipo SET é uma coleção de valores primitivos. Este tipo pode ser declarado como:

TYPE T = SET OF Base

onde *Base* é um valor primitivo. Os valores de T são todos os conjuntos cujos elementos têm o tipo *Base*. Desse modo, SET pode ser classificado como conjunto potência.

1.2.5 OBJECTS

Um tipo de objeto determina os tipos de prefixo dos campos do registro de dados, como se "OBJECT"fosse "REF RECORD". Mas no caso de um tipo de objeto, o registro de dados pode conter campos adicionais introduzidos por subtipos do tipo de objeto. Da mesma forma, o tipo de objeto determina um prefixo do conjunto de métodos, mas o conjunto pode conter métodos adicionais introduzidos por subtipos.

Há dois tipos de objetos pré-declarados:

ROOT	The traced object type with no fields or methods
UNTRACED ROOT	The untraced object type with no fields or methods

A declaração de um tipo objeto é dado na forma:

```
TYPE T = ST OBJECT
             Fields
             METHODS
             Methods
             OVERRIDES
             Overrides
             END
```

2 Variáveis quanto ao armazenamento e acesso

Em Modula-3, as variáveis são declaradas usando o prefixo VAR.

Na prática, variáveis simples são "empacotadas"(predefinidas) de certa forma dentro da linguagem. As variáveis compostas surgem da necessidade do usuário da linguagem por informações cujos tipos não estão definidos de forma "empacotada"na linguagem.

2.1 Simples

Variáveis **simples** são variáveis do qual o seu valor como um todo é armazenado/acessado de forma atômica, ou seja, não há possibilidade de realizar o armazenamento/acesso de parte das informações desta variável, por definição da própria linguagem. Exemplos de variáveis deste tipo são números inteiros, de ponto flutuante, caractere, booleanos e apontadores (tipos primitivos). Alguns tipos compostos, como os conjuntos em Pascal, são considerados variáveis simples, pois são tratados como um único elemento e sua manipulação é feita através de operações predefinidas. Tipos recursivos, como listas, também são consideradas variáveis simples em certas linguagens, por também possuírem operações predefinidas.

Variáveis simples em Modula-3: enumerações, sublistas, INTEGER, LONGINT, CARDINAL, BOOLEAN, CHAR, REAL, LONGREAL, EXTENDED, REF.

2.2 Compostas

Variáveis **compostas** são variáveis em que podemos armazenar/acessar em partes específicas dela, de forma seletiva e customizada em relação à linguagem. A informação que compõe a variável é dividida em diversas partes. Exemplos deste tipo de variável são *record* (Pascal), *struct* (C) e *union* (Pascal e C). Usando estas variáveis compostas, o usuário da linguagem pode construir tipos novos, não existentes na linguagem, como, no caso de C, um tipo Data, que armazena dias, meses e anos, ou tipo Cor, que armazena cores. Listas também podem ser variáveis compostas, como no caso de listas ligadas em C, em que são usadas structs e ponteiros.

Variáveis compostas em Modula-3: RECORD, OBJECT, ARRAY, SET, PACKED.

Mapeamentos (arrays) podem ser predefinidos na linguagem (C, C++, Java) ou definidos pelo usuário (Pascal, Modula-3 e Ada), podendo ser variáveis simples ou compostas.

3 Variáveis quanto a sua existência

3.1 Globais

As variáveis globais são aquelas vinculadas as células de memória antes da execução do programa e assim permanecem ate que a execução do programa se encerre. Modula-3 é uma linguagem que consiste em uma série de interfaces e módulos. As variáveis declaradas no nível mais externo em um módulo são chamadas de variáveis globais. Os procedimentos encontrados em um módulo podem operar diretamente nas variáveis globais declaradas no mesmo módulo.

global: INTEGER
PROCEDURE teste(VAR n: INTEGER) =
cur: INTEGER
BEGIN
IF(*global* > *cur*)
THEN RETURN global
ELSE RETURN cur
END;
END;
END teste;

3.2 Locais

Variáveis locais são aquelas que o tempo de vida está relacionado aos blocos de execução nos quais elas estão inseridas. Modula-3 possui blocos de comandos nos quais podemos ter definidas estas variáveis locais, que são acessíveis apenas no tempo de execução dos blocos.

Exemplo:

```
1  PROCEDURE maior(VAR n:INTEGER) =
2      cur : INTEGER;
3    BEGIN
4        IF (n > cur)
5          THEN RETURN n
6          ELSE RETURN cur
7        END;
8    END maior;
9
```

Figura 1: Pseudocódigo de uma variável local

Nesse exemplo temos `cur` como uma variável local (só existe quando a função é executada). Quando a função é chamada, a variável é armazenada em um espaço de memória. Quando a chamada termina, o espaço é liberado.

3.3 Heap

As variáveis *heap* são alocadas e desalocadas em tempo de execução. Esta alocação/desalocação pode ser feita explicitamente com comandos da linguagem ou por coletores de lixo implementados no projeto da linguagem. Modula-3 utiliza-se de um coletor de lixo para associação das variáveis nos endereços de memória.

Variável que ocupa e desocupa espaço de memória TRACED(rastreável) e UNTRACED(inrastreável)
Alocação dinâmica usando new() aloca tanto o tipo rastreável quanto o não rastreável
DISPOSE está disponível para desalocação dinâmica usando para desalocar o tipo não rastreável
Exemplo operação de alocação:
NEW (T, ...)

A operação retorna o endereço de uma variável recém-alocada do tipo referente de T; ou se T for um tipo de objeto, um registro de dados recém-alocado emparelhado com um conjunto de métodos. o referência retornada por NEW é distinta de todas as referências existentes. O tipo alocado da nova referência é T.

Exemplo operação de desalocação:
DISPOSE (v)

Se v não for rastreado, a instrução libera o armazenamento para o referente de v e define v como NULL.

3.4 Persistentes

As variáveis persistentes são mantidas intactas de alguma forma, fora do tempo de execução de um programa. Em geral, a informação está guardada em um arquivo, e este arquivo pode ser acessado utilizando uma variável de tipo arquivo, com comandos próprios para sua manipulação. Em Modula-3, é possível importar a interface STDIO para executar operações de leitura e escrita em arquivos.

4 Forma e tempo de vinculação de tipos às variáveis

A vinculação é feita *a priori* nos tipos predefinidos das linguagens. Para tipos não definidos, é necessário vincular às variáveis a um novo identificador, criado pelo usuário.

A vinculação de variáveis possui duas propriedades: a forma e o tempo de vinculação delas a um determinado espaço de memória. Dentre as formas de vinculação, temos a forma **explícita**, onde o tipo do identificador é escrito junto a ele, ou a forma **implícita**, onde a linguagem infere sobre seu tipo, dependendo do contexto. O tempo de vinculação pode ser **estático**, onde a variável é vinculada antes do tempo de execução (na compilação), ou pode ser **dinâmico**, onde a variável é vinculada durante o tempo de execução. Em Modula-3, a vinculação é estática, ou seja, as variáveis são todas vinculadas em tempo de compilação.

4.1 Primitivas

Os tipos primitivos podem ser vinculados de forma implícita ou explícita. Para realizar uma vinculação implícita é necessário inicializar a variável no momento de sua criação. Já a vinculação explícita é realizada tanto quando uma variável é declarada sem valor inicial quanto ao ser declarada e inicializada ao mesmo tempo. Já sobre o tempo de vinculação, as variáveis de tipos primitivos têm vinculação estática, ou seja, em tempo de compilação. Essa é uma das razões pelas quais não é possível realizar uma vinculação implícita sem inicializar a variável.

Exemplos em Modula-3:
VAR x: INTEGER; *(O tipo é declarado mas x não é inicializado)*
VAR s := "Hello"; *(O tipo text é inferido)*

4.2 Compostas

As variáveis de tipos compostos devem ter seu tipo declarado; portanto, há somente vinculação explícita. Quanto ao tempo de vinculação, quando há um array de um tipo fixo, é feita a vinculação em tempo de compilação. Quando há a declaração de um array de tipo aberto, é feita a vinculação em tempo de execução, quando a variável é alocada.

Exemplos em Modula-3:
VAR a := ARRAY [1..3] OF REAL 1.0, 2.0, 3.0; *(tipo fixo)*
TYPE T = ARRAY OF Element *(array aberto, onde "Element"é qualquer tipo)*

5 Referências

- https://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-113.pdf
- https://www.cs.purdue.edu/homes/hosking/m3/reference/m3.html
- https://en.wikipedia.org/wiki/Modula-3
- http://www1.cs.columbia.edu/graphics/modula3/tutorial/www/m3 toc.html