Introdução à Computação

Rosane Minghim e Guilherme P. Telles

9 de Agosto de 2012

Capítulo 6

Registros e Arquivos

Neste capítulo apresentamos registros e arquivos. Registros são estruturas que permitem criar tipos de dados heterogêneos, isto é, que são formados pela composição de outros tipos. Arquivos são estruturas para manipular dados armazenados em dispositivos de memória secundária, como discos magnéticos e ópticos, fitas e cartões de memória.

6.1 Registros

Um **registro** é uma estrutura que permite a definição de variáveis e tipos de dados heterogêneos, isto é, que são formados pela composição de outros. Por exemplo, usando um registro podemos construir uma variável para armazenar vários dados relativos a um aluno e tratá-los como um conjunto.

```
aluno registro
  cadeia nome
  caractere sexo
  cadeia endereço
fim registro
```

Uma variável do tipo `aluno` será um conjunto composto por

- uma variável do tipo cadeia de caracteres chamada `nome`,
- uma variável do tipo caractere chamada `sexo` e
- uma variável do tipo cadeia de caracteres chamada `endereço`

Cada um dos elementos deste conjunto é chamado de **campo** e pode ser referenciado individualmente. O operador de acesso aos campos de um registro é o ponto ´.´.

Usando pseudo-código uma variável que é um registro é declarada da seguinte forma:

```
nome registro
  tipo nome, nome, ..., nome
  tipo nome, nome, ..., nome
  ...
  tipo nome, nome, ..., nome
fim registro
```

Um tipo que é um registro é definido da seguinte forma:

```
tipo nome = registro
  tipo nome, nome, ..., nome
  tipo nome, nome, ..., nome
  ...
  tipo nome, nome, ..., nome
fim registro
```

No exemplo abaixo ilustramos a criação da variável `aluno`, a atribuição e o acesso aos campos do registro.

Exemplo 6.1

```
variável
  aluno: registro
    nome: cadeia[30]
    matrícula: cadeia[6]
    disciplina: cadeia[7]
    nota1: real
    nota2: real
  fim registro

início
  aluno.nome ← "Petrus"
  aluno.matrícula ← "001827"
  aluno.nota1 ← 5.7
  aluno.nota2 ← 1.25 * aluno.nota1
  escreva("A nota do aluno", aluno.nome, "é", aluno.nota2)
fim.
```

O conjunto de campos de cada variável do tipo registro em um algoritmo é independente. No exemplo abaixo, os campos das variáveis `aluno1` só é afetado pelos comandos de atribuição que se referem a ele. O mesmo vale para os campos de `aluno2`.

Exemplo 6.2

```
tipo
  aluno = registro
    nome: cadeia[30]
    nusp: cadeia[6]
    nota: real
  fim registro

variável
  aluno: aluno1, aluno2

início
  aluno1.nome ← "Petrus"
  aluno1.nusp ← "001827"
  aluno1.nota ← 5.7

  aluno2.nome ← "Claudius"
  aluno1.nusp ← "038758"
  aluno1.nota ← 7.2

  escreva("A média das notas dos alunos ", aluno1.nome, " e ",
           aluno1.nome, " é ", (aluno1.nota * aluno2.nota)/2)
fim.
```

Este algoritmo gera como saída:

```
A média das notas dos alunos Petrus e Claudius é 6.45
```

Os registros podem ser usados como qualquer outro tipo de variável: podemos construir vetores de registros e passar registros como parâmetros para funções.

6.2 Arquivos

A leitura desta seção deveria ser precedida da leitura da Seção 1.4.

Um arquivo é uma abstração, isto é, uma forma simplificada de pensar em um conjunto de dados armazenado em dispositivos de memória secundária, como discos, fitas e cartões de memória. Usando arquivos, podemos esquecer dos detalhes relacionados a cada tipo de dispositivo e manipular dados armazenados em memória secundária da mesma forma, independentemente do fato dos dados estarem em disquete, HD, CD-ROM, memória não-volátil (flash) ou outro meio qualquer.

Em um algoritmo, um arquivo tem um tipo que está relacionado ao tipo dos dados que são gravados no arquivo[1].

Em um algoritmo, um arquivo é uma coleção seqüencial e homogênea de dados. Um arquivo pode ser visto como um vetor que pode crescer à direita. Um arquivo está associado com algum dispositivo de memória secundária ou de entrada e saída. Por exemplo, um arquivo pode estar associado com uma região de um disco ou com uma conexão remota através de uma placa de rede. A forma como os dados são escritos e lidos no algoritmo não depende do dispositivo a que o arquivo está associado.

Em um algoritmo, um arquivo está associado a uma variável do tipo caractere ou registro. Por exemplo:

```
tipo
  pessoa = registro
    nome: cadeia[30]
    idade: inteiro
  fim registro

variável
  arquivo de pessoa: pessoas
  arquivo de caractere: relatorio
```

Uma variável do tipo arquivo tem algumas informações associadas, que são modificadas ou recuperadas através de funções:

1. Uma referência para um arquivo em memória secundária.

2. Um índice de posição no arquivo.

Antes de ler ou escrever em um arquivo ele deve ser aberto. Quando um arquivo é aberto, uma variável do tipo arquivo fica associada a um arquivo em memória secundária e o índice de posição do arquivo fica setado para o início do arquivo recém-aberto. A função `abra` têm a seguinte forma:

[1] A forma de definir o tipo de um arquivo em linguagens de programação é muito variada. Ela tem relação com a maneira como as funções de manipulação de arquivos oferecidas pela linguagem gravam registros e seus campos em arquivos e do sistema operacional.

```
abra(variável, nome, modo)
```

onde o primeiro parâmetro é uma variável do tipo arquivo, `nome` é uma cadeia de caracteres com o nome do arquivo no dispositivo de memória secundária e `modo` é uma cadeia de caracteres que pode ter os valores `leitura` ou `escrita`.

Um arquivo pode ser aberto para escrita ou para leitura. Quando o arquivo é aberto para escrita e não existe, ele é criado com tamanho igual a zero bytes. Se ele já existe, seu conteúdo é descartado e ele fica com tamanho igual a zero bytes. Quando um arquivo é aberto para leitura, seu conteúdo é preservado. Um arquivo que não existe não pode ser aberto para leitura. Pode-se escrever em um arquivo aberto para leitura, levando-se em conta que o conteúdo do arquivo será substituído se o apontador de arquivo se referenciar a uma posição que tenha algum conteúdo.

Depois que o arquivo está aberto, as funções `escreva` e `leia` podem ser usadas para acrescentar ou modificar o conteúdo de um arquivo. A operação `escreva` escreve um ou mais dados em um arquivo e a operação `leia` lê dados do arquivo.

```
escreva(variável do tipo arquivo, constante ou variável, ...,
        constante ou variável)
leia(variável do tipo arquivo, constante ou variável, ...,
     constante ou variável)
```

No exemplo abaixo, um arquivo chamado `c:/home/alunos` é aberto, os dados de cinco alunos são lidos do teclado e gravados no arquivo.

Exemplo 6.3 *tipo aluno = registro nome: cadeia[30] nota: real fim registro*
arquivo_alunos = arquivo de aluno
variável aluno: aux arquivo_alunos: cadastro
inicio abra(cadastro,"c:/home/alunos","escrita")
para i de 1 até 5 faça leia(aux.nome) leia(aux.nusp) escreva(cadastro,aux) fim para
feche(cadastro) fim

Quando uma operações para escrever no arquivo é executada, o dado é escrito na posição indicada pelo índice do arquivo. Ao término da operação de escrita, o índice passa a referenciar a posição do arquivo imediatamente após o dado que foi escrito.

A operação `posicione` move o apontador do arquivo para uma posição determinada. A posição pode ser absoluta `INÍCIO` ou `FIM` do arquivo, ou relativa à posição atual do apontador, para frente ou para trás. A operação

`feche` fecha o arquivo. Depois de fechado, nenhuma outra operação pode ser feita, a menos que o arquivo seja aberto novamente.

A forma das funções em pseudocódigo é:

```
posicione(nome-logico, inteiro, ATUAL)
posicione(nome-logico, inteiro, FIM)
posicione(nome-logico, inteiro, INICIO)
feche(nome-logico)
```

No exemplo abaixo, um arquivo contendo registros de alunos é aberto e depois o segundo e o quarto registro são lidos.

Exemplo 6.4 *tipo aluno = registro nome: cadeia[30] nota: real fim registro*
arquivo_alunos = arquivo de aluno
variável aluno: aux arquivo_alunos: cadastro
inicio abra(cadastro,"c:/home/alunos","leitura")
posicione(cadastro,2,INICIO); leia(cadastro,aux) imprima(aux.nome);
posicione(cadastro,2,ATUAL); leia(cadastro,aux) imprima(aux.nome);
feche(cadastro) fim