```
algoritmo "vet_strings"
var
   vet_strings: vetor [1..10, 1..31] de caractere
   i, j: inteiro
   aux: caractere
   funcao caractere_valido (aux: caractere): logico
   inicio
      retorne (...)
   fimfuncao
Inicio
   para i de 1 ate 10 faca
      escreva ("Obs.: Digite um a um os caracteres que compõem a" )
      escreva (" string, pressionando após")
      escreval ("cada caractere a tecla enter.")
      escreva ("A leitura de uma string é finalizada pelo fornecimento")
      escreva (" de 30 caracteres ou")
      escreval ("pelo fornecimento do caractere espaço ' '.")
      escreval ("Entre com os caracteres da ",i,"ª string:")
      para j de 1 ate 30 faca
        repita
           leia (aux)
        ate (caractere_valido(aux))
        se (aux=" ") entao
           interrompa
        senao
           vet_strings[i,j]<-aux
        fimse
      fimpara
```

```
      vet_strings[i,j]<-"\0"
      se (j=31) entao
        escreval ("Atenção! Tamanho máximo da ",i,"ª string atingido.")
        se (i<>10) entao
          escreval ("Os próximos caracteres a serem digitados pertencerão
       a ",i+1,"ª string")
        fimse
      fimse
    fimpara
    escreval ("Conteúdo do vetor.")
    para i de 1 ate 10 faca
      escreva (i,"ª string: ")
      para j de 1 ate 30 faca
        se (vet_strings[i,j]<>"\0") entao
          escreva(vet_strings[i,j])
        senao
          interrompa
        fimse
      fimpara
      escreval("")
    fimpara
fimalgoritmo
//Obs.: nesta resposta não foi explorado adequadamente o conceito de
//modularização, pois a intenção foi deixar o algoritmo compatível com
//o VisuAlg. Possibilitando que os alunos o utilizem para interpretar
//a solução.
```

Modularização

Exercício 42-b:

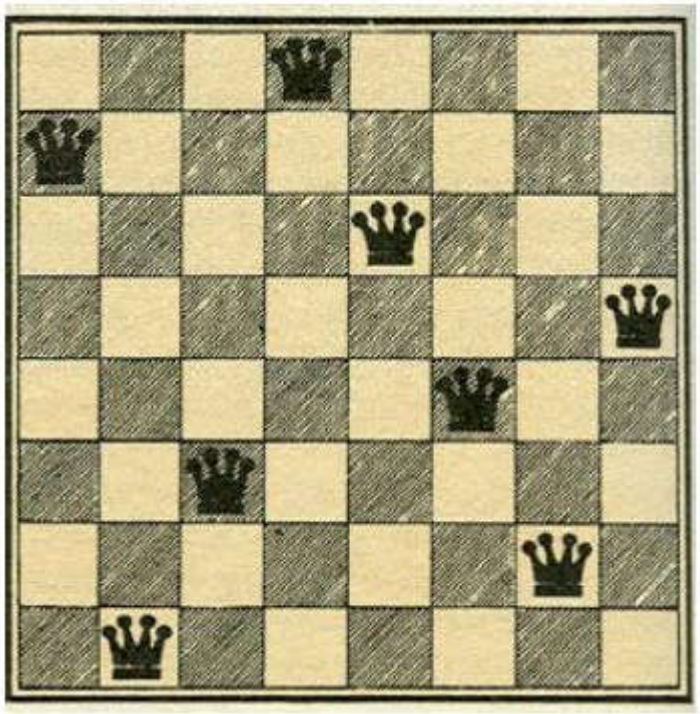
Considerando a movimentação de uma rainha no tabuleiro de xadrez, desenvolveremos um módulo capaz de identificar se um determinado tabuleiro ocupado por apenas 8 rainhas encontra-se em uma configuração válida para o jogo de oito rainhas.

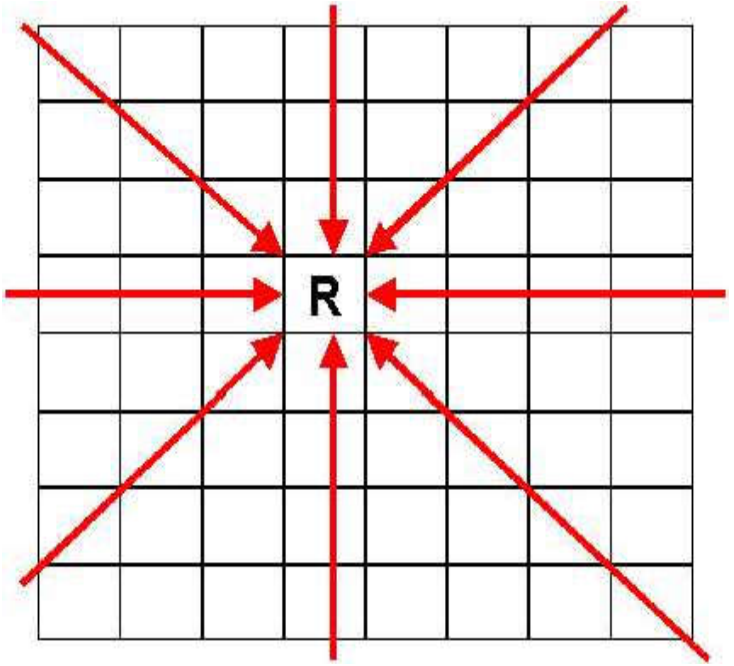
Uma configuração válida para o tabuleiro no jogo de oito rainhas, consiste em um tabuleiro onde cada uma das oito rainhas está posicionada em uma casa onde nenhuma das demais pode atingi-la com apenas um movimento.

Com base no que foi exposto, construa um módulo que receba uma matriz 8x8 de inteiros e retorne um lógico correspondendo ao fato da matriz recebida representar ou não um tabuleiro com uma configuração válida para o jogo de oito rainhas.

Observação: considere que casas desocupadas no tabuleiro conterão o valor 0 (zero) e casas ocupadas por rainhas terão o valor 1 (um).

Modularização





Estruturas de dados heterogêneas

Registros

Vimos inúmeras aplicações onde são necessários conjuntos de elementos do mesmo tipo, e para tal utilizamos os vetores.

No entanto, em alguns problemas há necessidade de definirmos conjuntos onde os elementos não sejam do mesmo tipo.

Um típico exemplo de nosso cotidiano é a utilização do conjunto de informações que caracterizam um aluno: Nome(caractere), CPF(inteiro), RG(inteiro), data de nascimento(caractere), coeficiente de rendimento(real), etc..

Estruturas de dados heterogêneas

Em uma análise superficial um estudante poderia pensar que uma solução para a questão apresentada poderia ser obtida declarando-se cinco variáveis:

algoritmo “exemplo”

var Nome: caractere

CPF: inteiro

RG: inteiro

data_de_nascimento: caractere

coeficiente_de_rendimento: real

…

Para uma melhor visualização da utilidade dos registros basta imaginarmos que ao invés de manipular as informações de um aluno exista a necessidade de gerenciamento de uma turma com cinquenta alunos.

Estruturas de dados heterogêneas

Um estudante desatento imaginaria ser necessário a declaração de 250 variáveis. Porém, um estudante com uma visão mais coerente dos conceitos estudados sugeriria a utilização de cinco vetores:

algoritmo "exemplo"
var Nomes: vetor [1..50] de caractere
CPFs: vetor [1..50] de inteiro
RGs: vetor [1..50] de inteiro
datas_de_nascimento: vetor [1..50] de caractere
coeficientes_de_rendimento: vetor [1..50] de real
...

Porém, manipular de forma adequada os vetores, mantendo seus dados consistentes, se torna trabalhoso. Com a utilização de um registro podemos resolver este problema apenas com um vetor de cinquenta registros.

Estruturas de dados heterogêneas

Exemplo:

algoritmo "exemplo"
var alunos: vetor [1..50] de registro
    inicio
        nome: caractere
        CPF: inteiro
        RG: inteiro
        datas_de_nascimento: caractere
        coeficientes_de_rendimento: real
    fimregistro
...

A cada um dos elementos que constituem um registro é dado o nome de campo. No exemplo acima, temos os campos: nome, CPF, RG, datas_de_nascimento e coeficientes_de_rendimento.

Estruturas de dados heterogêneas

Com base no exemplo anterior podemos deduzir a estrutura geral para a declaração de um registro:

<nome_da_variavel>: registro
inicio
<nome_campo_1>: <tipo_campo_1>
<nome_campo_2>: <tipo_campo_2>
...
<nome_campo_n>: <tipo_campo_n>
fimregistro

Estruturas de dados heterogêneas

Abriremos um parêntese em nosso estudo sobre registros para falarmos sobre **definição de tipo de dado**.

Com o objetivo de facilitar a leitura e consequentemente o entendimento dos algoritmos construídos foi criada o conceito de definição de tipo de dado.

Sintaxe:

tipo <nome_do_tipo>: <definicao_do_tipo>

Exemplo:

tipo vetor_de_inteiros: vetor [1..100] de inteiro

As definições de tipos devem ser feitas entre a constante caractere que nomeia o algoritmo e a declaração de variáveis globais ou dos módulos.

Obs.: Um estudante atento já vislumbrou as vantagens da definição de tipo na passagem de vetores e registros como parâmetros em módulos.

Estruturas de dados heterogêneas

Com a utilização dos conceitos vistos podemos resolver o problema aludido da seguinte forma:

```
algoritmo “exemplo”
tipo registro_aluno: registro
inicio
   nome: caractere
   CPF:  inteiro
   RG:  inteiro
   datas_de_nascimento: caractere
   coeficientes_de_rendimento: real
fimregistro
tipo vetor_de_registros: vetor [1..50] de registro_aluno
var alunos: vetor_de_registros
...
```

Estruturas de dados heterogêneas

Agora devemos tratar de como é feita a manipulação de um registro.

Da mesma forma que trabalhamos com um vetor acessando-o elemento a elemento, seja para atribuição ou seja para consulta de um valor, o mesmo ocorre com relação aos registros, devemos acessá-lo campo a campo.

Para acessarmos um determinado campo de um registro devemos utilizar o operador "." da seguinte forma:

No caso do exemplo com o qual temos trabalhado, a leitura do campo nome do décimo segundo aluno da turma é feita através de

leia(alunos[12].nome)

ou a impressão na saída padrão do CPF do terceiro aluno seria feita da seguinte forma

escreva(alunos[3].cpf)

UNIVASF

Exercício 43:

Defina um tipo de dado capaz de armazenar as seguintes informações sobre um determinado cliente de um banco: nome, CPF, RG, número da conta, data de abertura da conta e saldo.

```
tipo data: registro
inicio
   dia:inteiro
   mes: inteiro
   ano: inteiro
fimregistro
tipo registro_conta: registro
inicio
   nome: caractere
   cpf: caractere
   rg: caractere
   numero_conta: inteiro
   data_abertura: data
   saldo: real
fimregistro
```

Exercício 44:

Com base no exercício anterior, construa um algoritmo que manipule um vetor com 15 registros de clientes, onde cada registro é um elemento do tipo de dado definido. A manipulação do vetor é feita através dos seguintes módulos: inicializar vetor, imprimir um determinado registro com base no valor do campo CPF e imprimir um determinado registro com base em sua posição no vetor. O algoritmo deve se utilizar de forma satisfatória dos módulos mencionados e não deve possuir variáveis globais.

UNIVASF