universidade de aveiro deti departamento de electrónica, telecomunicações e informática

Programação I

Folha Exercícios 4

António J. R. Neves
João Rodrigues
Osvaldo Pacheco
Arnaldo Martins

2018/19/20

Folha Exercícios 3

Resumo:

- Estruturas de controlo - repetição
- Operadores aritméticos unários
- Instrução de atribuição com operação
- Instrução repetitiva **while** e **do...while**

Para além da execução condicional de instruções, por vezes existe a necessidade de executar instruções repetidamente. A um conjunto de instruções que são executadas repetidamente designamos por ciclo.

Um ciclo é constituído por uma estrutura de controlo que controla quantas vezes as instruções vão ser repetidas. As estruturas de controlo podem ser do tipo condicional (**while** e **do...while**) ou do tipo contador (**for**).

Normalmente utilizamos as estruturas do tipo condicional quando o número de iterações é desconhecido e as estruturas do tipo contador quando sabemos à partida o número de iterações.

Para escrever os programas dos exercícios seguintes, deve começar pelas especificações completas e os algoritmos finais em pseudocódigo. Os problemas apresentados, exigem para além da sequenciação de instruções, e instruções de decisão, a utilização das instruções de repetição apresentadas.

4.1 Problemas para resolver

Exercício 4.1

Escreva um programa que leia uma série de números inteiros. Quando for introduzido um número negativo, o programa deve escrever quantos números foram introduzidos e terminar.

Exercício 4.2

Escreva um programa que leia uma lista de números reais, terminada por um valor nulo, e calcule o seu produto. Exemplo de utilização:

```
Introduza uma lista de números (terminada com 0):
3  2.0  1.0  2  0
Produto = 12.000
```

Introduza uma lista vazia. Que produto dá? Matematicamente, faz sentido que o produto de uma lista vazia seja 1, tal como faz sentido que a soma de uma lista vazia seja 0.

(Chama-se ***valor sentinela*** a um valor especial que se usa num programa para indicar o fim de uma série de valores. Geralmente não deve ser processado como os restantes valores e por isso escolhe-se tipicamente um valor inválido, ilegal, ou pouco útil para a aplicação em questão.)

Exercício 4.3
Escreva um programa que leia uma lista de números reais, terminada por um valor igual ao primeiro que foi introduzido. No fim, indique o valor máximo, o valor mínimo, a média e o número de elementos da lista, não contando com o valor sentinela.

(Note que neste caso é impossível introduzir uma lista vazia, o que é conveniente porque não faz muito sentido achar o máximo ou o mínimo de um conjunto vazio.)

Exercício 4.4
O jogo *AltoBaixo* consiste em tentar adivinhar um número (inteiro) entre 1 e 100. O programa escolhe um número aleatoriamente. Depois, o utilizador insere uma tentativa e o programa indica se é demasiado alta, ou demasiado baixa. Isto é repetido até o utilizador acertar no número. O jogo acaba indicando quantas tentativas foram feitas.

Nota: Pode gerar o número secreto com o código abaixo.

```
int secret = (int)(100.0*Math.random()) + 1;
```

Exercício 4.5
Escreva um programa que leia do teclado um número inteiro positivo e determine se o número introduzido é um número primo. Um número natural é um número primo quando tem exatamente dois divisores naturais distintos: o número um e ele mesmo. Sugestão: tente dividir o número por 2, por 3, etc. Se for divisível por algum número menor que o próprio, então não é primo. Nota: Este algoritmo é simples, mas não é ideal. A verificação da primalidade é um problema importante e uma área de investigação atual. Repare que deve validar do valor de entrada repetindo a leitura se o valor não for válido (positivo).

Exercício 4.6
Escreva um programa que leia dois números inteiros e determine o máximo divisor comum (MDC) entre eles através do algoritmo de Euclides. Este algoritmo consiste em subtrair sucessivamente o menor número ao maior até que os dois números se tornem iguais. (Pode acelerar o processo se usar o resto da divisão em vez da subtração.)

Exercício 4.7
Escreva um programa que leia do teclado uma quantidade decimal, inteira não negativa, e que a escreva no monitor, pela ordem inversa dos seus algarismos (exemplo: 12345 -> 54321). Tenha em atenção a validação do valor de entrada.

4.2 Exercícios complementares

Exercício 4.8

Escreva um programa que calcule a multiplicação de dois números inteiros segundo um algoritmo utilizado antigamente pelos camponeses Russos (multiplicação russa). Este algoritmo de multiplicação pressupõe apenas o conhecimento da tabuada do dois.

O algoritmo funciona da seguinte forma: para multiplicar *X* por *Y*, divide-se *X* sucessivamente por dois até se obter o quociente 1 e ao mesmo tempo multiplica-se *Y* por 2. Adicionam-se ao resultado os valores de *Y* sempre que *X* é ímpar. Veja o seguinte exemplo da multiplicação de 25 por 15:

```
  X |   Y | soma
----------------
 25 |  15 | sim
 12 |  30 | não
  6 |  60 | não
  3 | 120 | sim
  1 | 240 | sim
----------------
soma: 375
```

O programa deve começar por pedir o valor de *X* e *Y* ao utilizador e depois imprimir o resultado obtido por este processo de multiplicação.

Exercício 4.9

Escreva um programa que leia uma lista de notas (inteiros de 0 a 20), até que apareça um número negativo, calcule a sua soma e a média. Note que a média é real, apesar das notas serem inteiras, e só faz sentido se tiver uma lista não vazia.

Exemplo de utilização:

```
Nota? 12
Nota? 9
Nota? -1
Soma = 21
Média = 10.5
```

Exercício 4.10

Escreva um programa que dada uma lista de números reais, terminada pelo valor zero, contabilize e escreva no monitor as quantidades de:

- números positivos e números negativos;
- números cujo valor se situa no intervalo [100…1000];
- números cujo valor se situa no intervalo [-1000…-100].

Exercício 4.11
Escreva um programa que leia uma lista de números inteiros positivos, até que seja lido um número que é o dobro do número anterior, e determine e escreva no monitor o menor e o maior dos números lidos. O programa deverá também escrever no monitor os números que forçaram a paragem da leitura.

Exercício 4.12
Escreva um programa que dado um número indeterminado de números inteiros positivos introduzidos pelo teclado, até que apareça o número zero como indicador de paragem, verifique se os números lidos constituem uma sequência exclusivamente constituída por números ímpares. Se a sequência for exclusivamente constituída por números ímpares, o programa escreve no monitor a mensagem "A sequência de números é exclusivamente constituída por números ímpares", senão escreve a mensagem "A sequência de números não é exclusivamente constituída por números ímpares".

Exercício 4.13
Altere o jogo *AltoBaixo* (Exercício 4.4) de forma que quando um jogo acaba, o programa pergunta "Novo jogo (s/n)?". O utilizador responde escrevendo uma letra. O programa só termina se a resposta for "n".

Pode ler uma palavra com o código:

```
String resp = sc.next();       // Lê próxima palavra
String resp = sc.nextLine();   // Lê uma linha
```

Deve testar a igualdade de `Strings` com a expressão:

```
resp.equals("n") //testa se Strings resp e "n" são iguais
```