Relatório

Algoritmos e Estruturas de Dados 1

Aluno/os: João Diogo / Duarte Melo

Professor/es: Marta Susana Lopes Martinho

Licenciatura em Engenharia de Sistemas Informáticos

Barcelos, janeiro, 2020

Índice

1. Introdução

Este relatório é referente ao Trabalho de Grupo I da Unidade Curricular "Algoritmos e Estruturas de Dados I" do 1º semestre do 1º ano, realizado por João Diogo Pinto Machado e Duarte Ribeiro de Melo, da turma de Engenharia de Sistemas Informáticos (Pós-Laboral).

Este trabalho visa cimentar os conhecimentos adquiridos durante as aulas nos seguintes temas:

- Instruções de decisão
- Instruções de repetição
- Funções e procedimentos
- Arrays
- Estruturas

2. Instruções de Decisão

2.1. Questão 1

2.1.1. Descrição e abordagem do problema

Enunciado:

Crie uma aplicação que solicite a quantidade de entregas no final do dia de cada um dos cinco paquetes contratados por uma empresa de entrega de refeições ao domicílio e que calcule a média das quantidades de refeições entregues por paquete e a soma das refeições entregues pelos paquetes com mais de 4 refeições entregues nesse dia.

Abordagem:

Foi lida a quantidade de entregas de cada paquete e foram feitas duas de decisão consoante o que era pedido no enunciado. No fim, foram apresentadas as médias consoante as instruções de decisão.

2.1.2. Fluxograma

https://imgur.com/a/H7B0NRk

2.1.3. Pseudocódigo

```
Algoritmo "Questão 1"
Variaveis
    somatorioMaiorQue4, somatorioMedia, quantidade: inteiro
    media: real
    Inicio
//paquete1
    Ler(quantidade);
    Se (quantidade > 4)
    somatorioMaiorQue4 <- somatorioMaiorQue4 + quantidade;
    Fim-Se
    somatorioMedia <- somatorioMedia + quantidade;
//paquete2
    Ler(quantidade);
    Se (quantidade > 4)
    somatorioMaiorQue4 <- somatorioMaiorQue4 + quantidade;
    Fim-Se
```

```
        somatorioMedia <- somatorioMedia + quantidade;
//paquete3
        Ler(quantidade);
        Se (quantidade > 4)
        somatorioMaiorQue4 <- somatorioMaiorQue4 + quantidade;
        Fim-Se
        somatorioMedia <- somatorioMedia + quantidade;
//paquete4
        Ler(quantidade);
        Se (quantidade > 4)
        somatorioMaiorQue4 <- somatorioMaiorQue4 + quantidade;
        Fim-Se
        somatorioMedia <- somatorioMedia + quantidade;
//paquete5
        Ler(quantidade);
        Se (quantidade > 4)
        somatorioMaiorQue4 <- somatorioMaiorQue4 + quantidade;
        Fim-Se
        somatorioMedia <- somatorioMedia + quantidade;
//calculos finais
        media <- somatorioMedia / 5; //não esquecer que assim é divisão inteira! (na implementação
        em código posterior)
//amostragem de valores
        Escrever(media);
        Escrever(somatorioMaiorQue4);
Fim
```

2.2. Questão 2

2.2.1. Descrição e abordagem do problema

Enunciado:

Um ano é bissexto se é divisível por 4, exceto se, além de ser divisível por 4, for também divisível por 100. Caso não seja divisível por 4 e for divisível por 400 também é bissexto. Crie uma aplicação que leia o valor de um ano e escreva se o ano é ou não bissexto.

Abordagem:

Primeiramente pedimos o ano ao utilizador, fizemos as verificações para ver se era bissexto ou não, e devolvemos ao utilizador se este é bissexto ou não.

2.2.2. Fluxograma

https://imgur.com/a/uwMLri1

2.2.3. Pseudocódigo

Algoritmo "Ano Bissexto"

Variaveis

ano : inteiro

bissexto : booleano

Inicio

```
bissexto <-- 0
ler(ano)
Se(ano % 4 = 0)
        Se(ano % 100 != 0)
                bissexto <-- 1
        Fim-se
Fim-se
Se(ano % 400 = 0)
        bissexto <-- 1
Fim-se
Se(bissexto = 1)
        escrever("o ano e bissexto")
Fim-se
Senao
        escrever("o ano nao e bissexto")
Fim-Senao
```

Fim

2.3. Questão 3

2.3.1. Descrição e abordagem do problema

Enunciado:

Crie uma aplicação que permita ler as notas de um aluno às disciplinas de Matemática, Português, Inglês e Geografia e calcular a sua média. Em função da média deve ser mostrada uma mensagem com o conteúdo "Aprovado" ou "Reprovado". Consideram-se notas positivas as notas iguais ou superiores a 9,5.

Abordagem:

Pedimos as notas ao utilizador, fizemos a média dessas mesmas notas, e mostramos se o aluno passou ou reprovou.

2.3.2. Fluxograma

https://imgur.com/a/mBkItDa

2.3.3. Pseudocódigo

```
Algoritmo "Media"

Variaveis
   media, somatorio, nota: reais
Inicio
   somatorio <- 0;
   escrever("Escreva a nota de matemática")
   ler (nota)
   somatorio <- somatorio + nota;

   escrever("Escreva a nota de portugues")
   ler (nota)
   somatorio <- somatorio + nota;

   escrever("Escreva a nota de ingles")
```

```
ler (nota)
somatorio <- somatorio + nota;

escrever("Escreva a nota de geografia")
ler (nota)
somatorio <- somatorio + nota;

media <- (somatorio) / 4

se (media >= 9.5)
   escrever("aprovado")
fim-se
senao
   escrever("reprovado")
fim-senao
```

Fim

2.4. Questão 4

2.4.1. Descrição e abordagem do problema

Enunciado:

Com o objetivo de promover a vacinação, um veterinário pretende atribuir um desconto nas vacinas para gatos mediante o seu comprimento. Crie uma aplicação que solicite os dados de um gato e indique ao utilizador qual o desconto a atribuir com base no quadro seguinte.

Abordagem:

Inicialmente, desconstruímos os intervalos para fazer as verificações com operadores relacionais.

Posteriormente, obtivemos os dados do animal.

Por fim, com os dados obtidos, efetuar as verificações com instruções de decisão e devolver o valor do desconto.

2.4.2. Fluxograma

https://imgur.com/a/9dMunBU

2.4.3. Pseudocódigo

```
Algoritmo "Desconto vacinas"
Variaveis
   comprimento: real
   genero: caracter
   desconto: inteiro
Inicio
      ler (comprimento)
      ler (genero)
      se (comprimento >= 6 && comprimento < 10)
         se (genero = F)
            desconto <- 10
         fim-se
      fim-se
      se (comprimento >= 10 && comprimento < 15)
         se (genero = M)
            desconto <- 9
         fim-se
      fim-se
      se (comprimento >= 15 && comprimento < 18)
         se (genero = F)
            desconto <- 8
         fim-se
      fim-se
      se (comprimento >= 18 && comprimento < 25)
         desconto <- 7
      fim-se
      senao
         desconto <- 5
      fim-senao
      escrever (desconto)
Fim
```

2.5 Instruções de Repetição

2.5. Questão 5

2.5.1. Descrição e abordagem do problema

Enunciado:

Crie uma aplicação que solicite ao utilizador um número natural e que

mostre ao utilizador o resultado de n ∑ i=1 em que xi representa o i esimo elemento

do conjunto de 1 até n.

Abordagem:

Incialmente, foi feita a obtenção do número.

Depois, com o auxílio de um ciclo for, escrevemos os números de 1 até ao número e a um somatório previamente criado somamos estes valores.

No final, mostramos ao utilizador o somatório.

2.5.2. Fluxograma

https://imgur.com/a/56SKz5z

2.5.3. Pseudocódigo

```
Variaveis
   num, somatorio: inteiro
Inicio
   ler(num)
   para i <- 1 até num passo 1 fazer
      escrever(i);
      somatorio <- somatorio + i

   fim-para-fazer
   escrever(somatorio)
Fim
```

2.1. Questão 6

2.1.1. Descrição e abordagem do problema

Enunciado:

Desenvolva um programa que receba um número indefinido de idades e que mostre na consola a quantidade de pessoas que 15<=idade<48 . Reflita acerca da melhor condição de paragem para este caso.

Abordagem:

Inicialmente, obtivemos o número de idades que iam ser introduzidas pelo utilizador.

Posteriormente, obtivemos as idades e repetimos o procedimento de repetir cada idade consoante o número de idades obtido anteriormente.

Dentro do ciclo de repetição, efetuamos a verificação pedida no enunciado. Caso esta se verificasse, incrementávamos um contador.

No fim, devolvemos o contador ao utilizador.

2.1.2. Fluxograma

https://imgur.com/a/i9t9Nl3

2.1.3. Pseudocódigo

```
Algoritmo "Questao 6"
Variaveis
   idade, contador: inteiros
Inicio
   idade <- 0
   contador <- 0
   enquanto (idade != -1)
      ler(idade)
      se (idade >= 15 && idade < 48)
         contador <- contador + 1
      fim-se
   fim-enquanto
   escrever(contador)
Fim
```

2.2. Questão 7

2.2.1. Descrição e abordagem do problema

Enunciado:

Foi efetuado um questionário a um número indeterminado de estudantes numa universidade. A todos os estudantes foi solicitado o género, a idade e se está a gostar ou não do curso que está a frequentar. Implemente uma aplicação capaz de calcular e informar:

• O número de estudantes entrevistados;

• Percentagem de estudantes por género;

• Quantidade de estudantes de género masculino com menos de 20 anos que não gostam do curso que estão a frequentar.

Abordagem:

Inicialmente, definimos que caso o utilizador introduzisse o caracter '0', terminava a inserção de "géneros" (de utilizadores, no caso).

O utilizador insere "géneros", insere a idade e se gosta do curso até que seja inserido o '0'.

Dentro deste ciclo, são efetuadas verificações consoante o género do utilizador. É também feita a verificação pedida no enunciado (quantidade de estudantes de género masculino com menos de 20 anos que não gostam do curso que estão a frequentar).

Utilizamos contadores para devolver os resultados:

contadorEntrevistados

contador

contadorF

contadorMascMenos20NGosta

No final, mostramos ao utilizador os resultados pedidos no enunciado: contador de entrevistados, percentagem de entrevistados do sexo masculino e feminino e contador de homens com menos de 20 anos que não gostam do curso.

2.2.2. Fluxograma

https://imgur.com/a/PwWKsUJ

2.2.3. Pseudocódigo

Algoritmo "Questao 7"

Variaveis

```
genero: caracter
idade , contadorEntrevistados, contadorM, contadorF, contadorMascMenos20NGosta: inteiro
gostaCurso: boleano
```

Inicio

```
contadorEntrevistados <- 0
contadorM <- 0
contadorF <- 0
contadorMascMenos20NGosta <- 0
ler(genero)
enquanto (genero != '0')
    contadorEntrevistados <- contadorEntrevistados + 1
    se (genero = 'M')
        contadorM <- contadorM + 1
    fim-se
    se (genero = 'F')
        contadorF <- contadorF + 1
    fim-se
    ler(idade)
    ler(gostaCurso)
    se (genero = 'M' && idade < 20 && gostaCurso = falso)
        contadorMascMenos20NGosta <- contadorMascMenos20NGosta + 1
    fim-se
    ler (genero)
fim-enquanto
escrever(contadorEntrevistados)
escrever((contadorM / contadorEntrevistados) * 100)
escrever((contadorF / contadorEntrevistados) * 100)
escrever(contadorMascMenos20NGosta)
```

Fim

2.3. Questão 8

2.3.1. Descrição e abordagem do problema

Enunciado:

Proponha um programa capaz de gerar de forma automática e aleatória 70 números inteiros positivos entre 0 e 100. Apresente na consola a soma e média dos números primos existentes no conjunto criado.

Abordagem:

Criamos uma função que verifica se um número é primo.

Criamos um ciclo for de 0 até 70, que serve para gerar 70 números aleatórios entre 0 e 100.

A partir da função para verificar se os números são primos, contabilizamos os números primos entre os números gerados e adicionamos os números a um somatório (somatorioPrimos).

Antes de apresentar os resultados ao utilizador, calculamos a média dos números primos.

Apresentamos, no fim, a soma e a média dos números primos.

2.3.2. Fluxograma

https://imgur.com/a/KoVhwWF

2.3.3. Pseudocódigo

```
Algoritmo "Questao 8"
funcao VerificaPrimo (numero: inteiro) : inteiro
  Inicio-funcao
    se(numero = 0 || numero = 1)
      retorna 0
    fim-se
    para i <- 2 ate numero passo 1 fazer
      se (numero % i = 0)
        retorna 0
      fim-se
    fim-para-fazer
    retorna 1
  Fim-funcao
```

```
Variaveis
   numero, limiteInferior, limiteSuperior, somatorioPrimos, contadorPrimos: inteiros
   mediaPrimos: real
Inicio
   para i <- 0 até 70 passo 1 fazer
      numero <- rand() % (limiteSuperior - limiteInferior) + limiteInferior
      escrever(numero);
      se(verificaPrimo(numero) = 1)
         somatorioPrimos <- somatorioPrimos + numero
         contadorPrimos <- contadorPrimos + 1
      fim-se
   fim-para-fazer
   mediaPrimos <- somatorioPrimos / contadorPrimos
   escrever(somatorioPrimos)
   escrever(mediaPrimos)
Fim
```

2.4. Questão 9

2.4.1. Descrição e abordagem do problema

Enunciado:

Um número palíndromo é um número que pode ser lido tanto da esquerda para a direita, como da direita para a esquerda. O maior número capicua resultante do produto entre dois números de dois dígitos 9009 = 91x99. Desenvolva uma aplicação que encontre o maio número palíndromo resultante do produto entre dois números de três dígitos.

Abordagem:

Neste exercício, criamos uma função para obter o inverso de um número (566 – 665). Depois, com dois ciclos for e com várias condições para corresponder ao enunciado, conseguimos obter o pedido – o maior número capicua resultante do produto entre dois números de dois dígitos.

No fim, escrevemos no ecrã o maior palíndromo.

2.4.2. Fluxograma

https://imgur.com/a/Phs4xVm

2.4.3. Pseudocódigo

```
Algoritmo "Questao 9"


funcao obterInverso (numero: inteiro) : inteiro
  inicio-funcao
    aux: inteiro
    inverso: inteiro
    aux <- numero
    inverso <- 0
    enquanto (aux != 0)
      inverso <- inverso * 10 + aux % 10
      aux = aux / 10
    fim-enquanto


    retornar(inverso)
  fim-funcao
Variaveis
  original, inverso: inteiros
Inicio
  para a <- 999 ate 100 passo -1 fazer
    para b <- 999 ate 100 passo -1 fazer
      original <- a * b
      inverso <- obterInverso(a*b)
      se (inverso = original)
        escrever(inverso)
        terminarprograma
      fim-se
    fim-para-fazer
  fim-para-fazer
Fim
```

2.10 Restantes entregas: documentação

Relativamente aos restantes exercícios, e dado que consideramos que a documentação dos mesmos está bastante rica, geramos a documentação pelo Doxygen, de maneira a que seja mais simples de ler, e mais fácil de gerar.

3. Conclusão

Como referido na introdução, este trabalho visa cimentar os conhecimentos adquiridos durante as aulas nos seguintes temas:

- Instruções de decisão
- Instruções de repetição
- Funções e procedimentos
- Arrays
- Estruturas

Após alguns meses de trabalho e, consequente aprendizagem, consideramos que todo este trabalho ajudou bastante na aquisição e consolidação de conhecimentos no que toca às bases da algoritmia / estrutura de dados / programação e ainda na autoaprendizagem e trabalho em grupo. O enunciado está bem estruturado e permitiu que aprendêssemos uma coisa de cada vez. Conseguimo-nos aperceber que, todo o conhecimento das entregas iniciais é necessário para as entregas finais, e, por isso, percebemos também o porquê da matéria ser dada com esta ordem.

Em suma, consideramos que com este trabalho consolidamos os conhecimentos adquiridos nas aulas da UC.

Bibliografia

(s.d.). Obtido de stackoverflow: https://stackoverflow.com/