utad UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO

TRABALHO PRÁTICO 1

PROGRAMAÇÃO FUNCIONAL

a80049 Lara Esteves
a79908 Matilde Coelho
a79858 Rui Soares

Índice

Introdução....2
Tarefa 1....3
Tarefa 2....5
Tarefa 3....7
Tarefa4 8
Tarefa 5....9

Introdução

O trabalho em Haskell consiste em desenvolver um programa que gerencie horários e salas para exames em uma instituição de ensino superior. Na primeira fase, o programa deve ser capaz de ler os ficheiros de texto fornecidos e realizar tarefas como mostrar no terminal todos os alunos inscritos em cada unidade curricular, mostrar todas as unidades curriculares em que cada aluno está inscrito, filtrar por aluno e por UC as inscrições e fazer um menu que permita acessar aos restantes programas.

Tarefa 1

```haskell
module Tarefa1 where
import System.IO ()

main1 :: IO()
main1 = do ucs <- readFile "ucs.txt"
           inscricoes <- readFile "inscricoes.txt"
           procuraUCs1 (lines ucs) inscricoes
           putStrLn "====================\n"

procuraUCs1 :: [String] -> String -> IO ()
procuraUCs1 [] _ = putStrLn ""
procuraUCs1 (linha:linhas) inscricoes = do putStrLn ("\n====================\n"++unwords(tail(tail(words linha)))++"\n-----
                                           procuraInscricoes1 (head(words linha)) (lines inscricoes)
                                           procuraUCs1 linhas inscricoes

procuraInscricoes1 :: String -> [String] -> IO()
procuraInscricoes1 _ [] = putStrLn ""
procuraInscricoes1 uc (linha:linhas) = if uc == last(words linha) then do alunos <- readFile "listaalunos.txt"
                                                                          procuraAlunos1 (head(words linha)) (lines alunos)
                                                                          procuraInscricoes1 uc linhas
                                       else procuraInscricoes1 uc linhas


procuraAlunos1 :: String -> [String] -> IO()
procuraAlunos1 _ [] = putStrLn ""
procuraAlunos1 al (linha:linhas) = if al == head(words linha) then do putStrLn (unwords(tail(tail(words linha))))
                                   else procuraAlunos1 al linhas
```

main1- é a função principal que será executada quando o programa for iniciado. Realiza as seguintes ações:

- Lê o conteúdo do arquivo "1" e armazena na variável *“ucs”*.

- Lê o conteúdo do arquivo *"inscricoes.txt"* e armazena na variável *“inscrições”.*

- Chama a função *“procuraUCs1”,* passando as linhas do ficheiro "*ucs.txt*" como argumento, juntamente com o conteúdo do ficheiro*"inscricoes.txt".*

- Imprime uma linha separadora no terminal (Razões Estéticas).

procuraUCs1- recebe uma lista de strings representando as linhas do ficheiro*"ucs.txt"* e uma string representando o conteúdo do ficheiro*"inscricoes.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Caso contrário, imprime o nome da UC (obtido a partir da segunda palavra na linha) no formato de cabeçalho.

- Chama a função “*procuraInscricoes1”,* passando o nome da UC atual e as linhas do arquivo *"inscricoes.txt"* como argumentos.

- Chama recursivamente a função “*procuraUCs1”,* passando o restante das linhas e o conteúdo do ficheiro "inscricoes.txt".

procuraInscricoes1- recebe uma string representando o nome da UC e uma lista de strings representando as linhas do ficheiro "*inscricoes.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Caso contrário, verifica se o nome da UC corresponde ao último elemento (palavra) da linha atual.

- Se houver correspondência, lê o conteúdo do ficheiro *"listaalunos.txt"* e chama a função *“procuraAlunos1”*, passando o nome da UC atual e as linhas do ficheiro "*listaalunos.txt"* como argumentos.

- Chama recursivamente a função *“procuraInscricoes1”,* passando o mesmo nome da UC e o restante das linhas.

procuraAlunos1- recebe uma string representando o nome da UC e uma lista de strings representando as linhas do ficheiro *"listaalunos.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Caso contrário, verifica se o nome da UC corresponde ao primeiro elemento (palavra) da linha atual.

- Se houver correspondência, imprime o restante das palavras na linha (representando os alunos inscritos na UC).

- Chama recursivamente a função *“procuraAlunos1”*, passando o mesmo nome da UC e o restante das linhas.

Tarefa 2

```haskell
module Tarefa2 where
import System.IO ()

main2 :: IO()
main2 = do alunos <- readFile "listaalunos.txt"
           inscricoes <- readFile "inscricoes.txt"
           procuraAlunos2 (lines alunos) inscricoes
           putStrLn "====================\n"

procuraAlunos2 :: [String] -> String -> IO()
procuraAlunos2 [] _ = putStrLn ""
procuraAlunos2 (linha:linhas) inscricoes = do
    if null (words linha) then putStrLn ""
    else do putStrLn ("\n====================\n"++unwords(tail(tail(words linha)))++"\n--------------------")
            procuraInscricoes2 (head(words linha)) (lines inscricoes)
            procuraAlunos2 linhas inscricoes

procuraInscricoes2 :: String -> [String] -> IO()
procuraInscricoes2 _ [] = putStrLn ""
procuraInscricoes2 al (linha:linhas) = do
    if null (words linha) then putStrLn ""
    else do if al == head(words linha) then do
               ucs <- readFile "ucs.txt"
               procuraUCs2 (last(words linha)) (lines ucs)
               procuraInscricoes2 al linhas
            else procuraInscricoes2 al linhas

procuraUCs2 :: String -> [String] -> IO()
procuraUCs2 _ [] = putStrLn ""
procuraUCs2 uc (linha:linhas) = do
    if null (words linha) then putStrLn ""
    else do
        if uc == head(words linha) then do putStrLn (unwords(tail(tail(words linha))))
        else procuraUCs2 uc linhas
```

-main2- Função principal do programa, na qual:

- Lê o conteúdo do ficheiro *"listaalunos.txt"* e armazena na variável *“alunos”*.

- Lê o conteúdo do ficheiro *"inscricoes.txt"* e armazena na variável *“inscricoes”.*

- Chama a função *“procuraAlunos2”,* passando as linhas do ficheiro "*listaalunos.txt*" como argumento, juntamente com o conteúdo do ficheiro "*inscricoes.txt".*

- Imprime uma linha separadora no terminal. (Razões estéticas)

procuraAlunos2 – recebe uma lista de strings correspondentes às linhas do ficheiro *"listaalunos.txt"* e uma string que corresponde ao conteúdo do ficheiro *"inscricoes.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Se a linha atual for vazia, imprime uma linha em branco.

- Caso contrário, imprime o nome do aluno (obtido a partir da segunda palavra na linha) no formato de cabeçalho.

- Chama a função *“procuraInscricoes2”,* passando o nome do aluno atual e as linhas do ficheiro *"inscricoes.txt"* como argumentos.

- Chama recursivamente a função *“procuraAlunos2”*, passando o restante das linhas e o conteúdo do ficheiro "*inscricoes.txt".*

procuraInscricoes2 - recebe uma string representando o nome do aluno e uma lista de strings representando as linhas do ficheiro *"inscricoes.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Se a linha atual for vazia, imprime uma linha em branco.

- Caso contrário, verifica se o nome do aluno corresponde ao primeiro elemento (palavra) da linha atual.

- Se houver correspondência, lê o conteúdo do ficheiro *"ucs.txt"* e chama a função *"procuraUCs"'*, passando o último elemento (palavra) da linha atual e as linhas do ficheiro *"ucs.txt"* como argumentos.

- Chama recursivamente a função "*procuraInscricoes2"*, passando o mesmo nome do aluno e o restante das linhas.

procuraUCs2 - recebe uma string representando uma unidade curricular (UC) e uma lista de strings representando as linhas do ficheiro *"ucs.txt".* Realiza as seguintes ações:

- Se a lista de linhas estiver vazia, imprime uma linha em branco.

- Se a linha atual for vazia, imprime uma linha em branco.

- Caso contrário, verifica se a unidade curricular (UC) corresponde ao primeiro elemento (palavra) da linha atual.

- Se houver correspondência, imprime o restante das palavras

Tarefa 3

```haskell
module Tarefa3 where
import System.IO ()
import qualified Tarefa1

main3 :: IO()
main3 = do putStr "\n================================================\nQUAL DAS UC'S PRETENDE VISUALIZAR AS INSCRIÇÕES?\r
           uc <- getLine
           putStrLn "--------------------"
           inscricoes <- readFile "inscricoes.txt"
           Tarefa1.procuraInscricoes1 uc (lines inscricoes)
           putStrLn "--------------------"
```

main3- Função principal. Realiza as seguintes ações:

- Imprime uma mensagem solicitando ao usuário que selecione uma UC para visualizar as inscrições.

- Lê a entrada do usuário utilizando a função *"getLine"* e armazena o valor na variável *"uc"*.

- Imprime uma linha separadora no terminal. (razões estéticas)

- Lê o conteúdo do ficheiro *"inscricoes.txt"* e armazena na variável *"inscrições"*.

- Chama a função *"procuraInscricoes1"* que é importada do módulo *"Tarefa1"*, passando a UC selecionada pelo usuário (UC) e as linhas do ficheiro *"inscricoes.txt"* como argumentos.

- Imprime uma linha separadora no terminal. (razões estéticas)

Tarefa 4

```haskell
module Tarefa4 where
import System.IO ()
import Tarefa2

main4 :: IO()
main4 = do inscricoes <- readFile "inscricoes.txt"
           putStr "\n================================================\nQUAL DOS ALUNOS PRETENDE VISUALIZAR AS INSCRIÇÕES:
           aluno <- getLine
           putStrLn "--------------------"
           if aluno == "1" then do
                let al = "al001"
                Tarefa2.procuraInscricoes2 al (lines inscricoes)
           else if aluno == "2" then do
                let al = "al002"
                Tarefa2.procuraInscricoes2 al (lines inscricoes)
           else if aluno == "3" then do
                let al = "al003"
                Tarefa2.procuraInscricoes2 al (lines inscricoes)
           else if aluno == "4" then do
                let al = "al004"
                Tarefa2.procuraInscricoes2 al (lines inscricoes)
           else putStrLn "Aluno Inválido"
           putStrLn "--------------------"
```

main4- função principal. Realiza as seguintes ações:

- Lê o conteúdo do ficheiro*"inscricoes.txt*" e armazena na variável *"inscrições"*.

- Imprime uma mensagem solicitando ao usuário que selecione um aluno para visualizar as inscrições.

- Lê a entrada do usuário usando a função *"getLine"* e armazena o valor na variável *"aluno"*.

- Imprime uma linha separadora no terminal. (Razoes estéticas)

- Usa estruturas condicionais "if" e "else" para determinar qual aluno foi selecionado pelo usuário e, em seguida, chama a função *"procuraInscricoes2"* importada do módulo *"Tarefa2"*, passando o código do aluno correspondente (alXXXX) e as linhas do ficheiro *"inscricoes.txt"* como argumentos.

- Se o valor de aluno não corresponder a nenhum dos alunos válidos, imprime *"Aluno Inválido*".

- Imprime uma linha separadora no terminal. (razoes estéticas)

Tarefa 5

```haskell
module Tarefa5 where
import System.IO
import Tarefa1
import Tarefa2
import Tarefa3
import Tarefa4

--Função principal, usada para mostar ao utilizador uma interface do menu
menu :: IO ()
menu = do
    putStr "\n================MENU================\n|1| - ALUNOS INSCRITOS POR UC\n|2| - UCs INSCRITAS POR ALUNO\n|3| - PRC
    opcao <- getLine
    escolheOpcao opcao

--Função do menu
escolheOpcao :: String -> IO ()
escolheOpcao opcao
            | opcao == "1" = do
                             Tarefa1.main1
                             putStr "PRESSIONE QUALQUER TECLA PARA CONTINUAR..."
                             _ <- getLine
                             menu
            | opcao == "2" = do
                             Tarefa2.main2
                             putStr "PRESSIONE QUALQUER TECLA PARA CONTINUAR..."
                             _ <- getLine
                             menu
            | opcao == "3" = do
                             Tarefa3.main3
                             putStr "PRESSIONE QUALQUER TECLA PARA CONTINUAR..."
                             _ <- getLine
                             menu
            | opcao == "4" = do
                             Tarefa4.main4
                             putStr "PRESSIONE QUALQUER TECLA PARA CONTINUAR..."
                             _ <- getLine
                             menu
            | opcao == "0" = return()
            | otherwise = do
                             putStrLn "OPÇÃO INVÁLIDA\n\nPRESSIONE QUALQUER TECLA PARA CONTINUAR..."
                             _ <- getLine
                             menu
```

menu- função principal. Ela apresenta um menu para o usuário com várias opções e realiza as seguintes ações:

- Imprime o menu no terminal.

- Lê a entrada do usuário usando a função *"getLine"* e armazena o valor na variável *"opcao".*

- Chama a função *"escolheOpcao"*, passando a opção selecionada pelo usuário como argumento.

escolheOpcao- recebe uma string representando a opção selecionada pelo usuário e realiza ações com base nessa opção. Realiza as seguintes ações:

- Se a opção for "1", chama a função Tarefa1.main1 para mostrar os alunos inscritos por UC.
- Se a opção for "2", chama a função Tarefa2.main2 para mostrar as UCs inscritas por aluno.
- Se a opção for "3", chama a função Tarefa3.main3 para realizar uma busca por UC.
- Se a opção for "4", chama a função Tarefa4.main4 para realizar uma busca por aluno.
- Se a opção for "0", retorna da função, encerrando o programa.
- Se a opção for diferente de todas as opções válidas, imprime "OPÇÃO INVÁLIDA".

- Aguarda a entrada do usuário com" getLine" e chama recursivamente a função menu para voltar ao menu principal.