Universidade Federal do Piauí –
Campus Senador Helvídio Nunes de Barros – CSHNB
Curso de Sistemas de Informação
Disciplina: Programação Funcional - Online
Professora: Juliana Oliveira de Carvalho

Análise de Algoritmos com o paradigma funcional

Ericksulino manoel de Araújo Moura

1. Resumo do Projeto

Projeto composto por cinco exercícios, onde devem ser colocados a prova os conhecimentos adquiridos na matéria até esse momento, os mesmos contém equações matemáticas, que necessitam ser resolvidos de forma eficaz, utilizando apenas o que foi ensinado em aula.

2. Introdução

Projeto realizado com o intuito de completar a primeira nota da disciplina de programação funcional, com cinco exercícios com problemas matemáticos complexos, como cálculos de MDC, números perfeitos, entre outros. Para serem resolvidas em forma de código fonte, utilizando o paradigma funcional e a linguagem de programação haskell,

Meios usados para a realização desse projeto foram algoritmos funcionais que continham loops com o método de recursão, com ou sem pendência, por ser o único meio de realizar o mesmo no paradigma funcional e na linguagem de programação haskell.

Todos os algoritmos foram estruturados por uma função principal onde era feito uma interação com o usuário e funções auxiliares para a realização dos cálculos matemáticos, para uma melhor organização dos mesmos.

3. Seções Específicas

a. Informações técnicas

Para o desenvolvimento e testes deste projeto foi utilizado uma máquina equipada com um processador quad-core, dois núcleos e quatro threads, com a velocidade em turbo boost chegando a 3.1Ghz, oito gigas de memoria ram ddr4, sistema operacional com a arquitetura de 64 bits, Linux Ubuntu na versão 20.04Lts. Os códigos foram feitos na ide e editor de texto Visual Studio Code e compilados pelo compilador GHCI versão 8.6.5.

b. Exercício 1

O exercício é um simples cálculo de um produtório com m ≥ 5. Ao estudá-lo notamos que há a necessidade de calcular dois fatoriais. Criando a função “fatorial” com “n” de parâmetro podemos efetuar o cálculo do

produtório. O exercício pede para o cálculo ser feito com e sem pendência. Para fazê-lo com pendência precisamos nomear a função “proCP” e retornar 300, o menor valor do produtório, para n = 5 e chamar “proCP” recursivamente, para n > 5 multiplicado pelo fatorial de n - 1 .

Para fazê-lo sem pendência precisaremos de uma variável auxiliar, passada nos parâmetros da função, chamada “aux” que começa, obrigatoriamente, por 1 e uma variável “prod” que recebe o cálculo do fatorial. Nomeamos a função sem pendência de “proSP” e retornamos o produto de “aux” e “prod” se n = 5, caso contrário (n > 5), chamamos a função recursivamente passando n -1 como primeiro parâmetro e aux*prod como segundo parâmetro. Para facilitar o trabalho do usuário foi criado uma função main em que o mesmo pode escolher se deseja a função com pendência ou sem pendência e logo após é solicitado o número m do produtório.

c. Exercício 2

O objetivo desse exercício é um algoritmo que leia três números inteiros e então faça a fatoração entre eles e mostre os valores que aparecem na fatoração e quantas vezes os mesmos aparecem. Foi feito então uma função auxiliar chamada de “fatora”, e a função principal somente chamada de “main”.

Na função “fatora” é onde é feito o cálculo que faz a fatoração dos números e diz quais elementos e quantas vezes aparecem, ela recebe como parâmetro o número que vai ser fatorado, uma variável “divi” e uma variável chamada de “i” que é o índice, dentro dessa função existem três condicionais, a primeiro verifica se o número é igual a um e retorna somente uma quebra de linha, pois se for igual a um quer dizer então que ele é primo.

Segunda condição verifica se o resto da divisão do número por “div” é igual a zero, se sim é mostrado na tela o “div” e o índice e então é chamada com o resultado da divisão inteira de número por “div” e o “div” e o índice sendo incrementado. Se não entrar nas condições anteriores, entra nessa condição que é o contrário da anterior, se o resto da divisão de número por “div” for diferente de zero, uma variável chamada de “cont” recebe um e a função “fatora” é chamada novamente com os seguintes parâmetros, número,”div” sendo incrementado e a variável “cont”.

A função principal é nada mais que um modo do usuário se comunicar melhor com o código fonte, com mensagens de texto indicando o que o mesmo deve fazer e oque vai aparecer na tela.

d. Exercício 3

Nesse exercício é pedido um código fonte que receba três números e retorne o máximo divisor comum entre eles. Para a resolução desse exercício que foram necessárias duas funções auxiliares, denominadas de “mdc” e “mdc3”, e a função principal chamada somente de main.

A função “mdc”, tem o objetivo de fazer o cálculo do máximo divisor comum dos dois primeiros números, dessa forma ela recebe como parâmetros os dois primeiros números, denominados por “a” e “b” respectivamente, que passam por três condições, na primeira delas, se o primeiro número o “a” for menor que o segundo o “b”, é chamada a função novamente invertendo a ordem dos mesmos, na segunda condição, se o segundo número o “b” for igual a 0 é retornado o primeiro número no caso o “a”, por fim se não entrar nas condições anteriores, é chamada a função novamente, dessa vez com o “b” e o resto da divisão de “a” por “b” como parâmetros.

Na função “mdc3”, é realizado o cálculo do MDC ou Máximo Divisor Comum dos três números, dessa forma ela recebe os três números como parâmetros, nomeados por “a”, “b”, e “c”, e chama a função “mdc“ com o “a” como o primeiro parâmetro e a própria função “mdc” com os parâmetros “b” e “c” como o segundo parâmetro.

Função principal ou simplesmente “main”, é a responsável por tornar o código mais atrativo para o usuário, trazendo mais interações com o mesmo, ela é bem simples, aparece uma mensagem de texto pedindo para o usuário digitar os três números, um de cada vez, guarda os mesmos para depois chamar a função “mdc3”, com esses mesmos três números como parâmetros, o valor retornado por essa função é guardado em uma variável que depois é exibida ao usuário com uma mensagem de texto.

e. Exercício 4

Aqui é pedido um algoritmo capaz de receber um número e determinar se ele é perfeito, deficiente ou abundante, e fazer isso quantas vezes o usuário desejar. Como no exercício anterior, aqui também foram necessárias duas funções auxiliares dessa vez nomeadas de “divis” e “perf” respectivamente e a função principal, nomeada somente por “main”.

Na função auxiliar “divis” recebe três número e retorna o resultado da soma dos seus divisores, da seguinte forma, a mesma recebe três números inteiros como parâmetros, denominados de “i” , “n” e “s” respectivamente, na linha seguinte tem uma condição, se o “i” for igual ao “n” e o resto da divisão do “n” por “i” for igual a zero, é retornado o “s”, que em termos leigos quer dizer exatamente que se a variável “i” for igual ao número e for divisível por ele, é retornado o resultado da soma.

Próxima linha tem outra condição que é a mesma da anterior, mudando só que nessa não tem a condição do “i” ser divisível por “n”. Na linha abaixo contém a penúltima condição da função, que diz o seguinte, se o “i” for menor que o “n” e o resto da divisão do “n” por “i” for igual a zero, então é chamada a própria função “divis” com o “i” sendo incrementado o “n” normal e o “s” sendo incrementado também, como os parâmetros da mesma.

A última linha da função “divis” tem uma condição parecida com a da linha anterior, nesse caso é se o “i” for menor que o “n” então é chamada novamente a função “divis” com os seguintes parâmetros, o “i” sendo incrementado, o “n” e o “s”.

Função “perf” é onde o número é verificado se é perfeito, deficiente ou abundante, ela recebe o número e primeiro compara se ele é maior que zero, se sim a função “divs” é chamada com 1 que onde começa a contar, o “n” que é o número e o 0 que é o valor inicial da variável que vai fazer o somatório como parâmetros, se o valor retornado por essa função for igual ao número, ele é perfeito, se for menor, ele é deficiente e se for maior é abundante, para cada uma das alternativas é exibida para o usuário uma mensagem de texto e logo em seguida é chamada a função “main”.

Se o número for igual a zero ele não entra em nenhuma das condições anteriores e é somente exibida uma mensagem de texto informando que o programa foi finalizado com sucesso.

Por fim, a função “main”, serve para ter uma interação maior com o usuário, o mesmo é instruído a digitar um número ou zero para finalizar o programa , o número digitado é usado como parâmetro para a função “perf” que é chamada dando início ao loop que só vai finalizar quando for digitado 0 pelo usuário.

f. Exercício 5

Neste problema, o algoritmo vai sortear um número para Ana e Beatriz, sendo que estes valores não podem exceder de 100. Ele vai imprimir o número da cadeira onde cada uma delas deve sentar, sendo que o primeiro valor corresponde a Ana, o segundo a Beatriz e o terceiro a Carolina. Para a resolução do mesmo foram feitas três funções auxiliares e uma principal, e ainda para que pudesse sortear os números foi necessário importar uma biblioteca.

função “mesa” é responsável por pegar o número sorteado e contar de acordo com ele o número da cadeira, por meio de uma função recursiva sem pendência, dessa forma o número é comparado a um contador que conta o número de contagens se eles são iguais essa é a condição de parada, e é retornado o “i” que vai ser melhor descrito mais para frente, em seguida tem a condição comparando se o contador é menor que o número e se o “i” que é a variável que vai guardar o valor da cadeira é igual a 2, se sim é chamada a função novamente com o número o “i” menos dois e o contador mais um, como os parâmetros, por fim na última condição é somente comparado se o contador é menor que o número, se sim é chamada a função com o número, o “i” mais um e o contador também mais um.

Na função “bia” o número sorteado é usado para chamar a função “mesa” e comparar se o número retornado por “mesa” é igual ao número de Ana, se sim o número de Beatriz é o próximo na contagem, se não o número de Beatriz não muda.

Na função “carol” é responsável por dizer em qual cadeira Carolina vai se sentar, levando em conta os números de Ana e Beatriz. Os número das cadeiras de Ana e Beatriz são comparados, de forma em que dadas as possibilidades o número de Carolina deve ser sempre diferente do das outras duas.

Função principal responsável por sortear os números de Ana e Beatriz, chamar as funções, determinar os números das três e manter uma boa interação com o usuário, primeiro de tudo são sorteados os números de Ana e Beatriz, a cadeira de Ana é o número retornado da função “mesa” com o número sorteado como parâmetro, a de Beatriz é o retorno da função “bia” com o seu número sorteado e a cadeira de Ana como parâmetros, por fim a cadeira de Carolina vem da função “carol” com as cadeiras de Ana e Beatriz como os parâmetros. Todos os números são apresentados para o usuário com breves mensagens de texto.

4. Resultados da Execução do Programa

Dados os códigos feitos para esse projeto, esses são os resultados de alguns testes feitos nos mesmos, será mostrado a entrada e a saída em uma tabela abaixo da descrição de cada código, os números das entradas foram escolhidos de forma aleatória, por mim quando fiz os testes.

Exercício número um é um algoritmo que lê um número e então por meio de uma função calcula o resultado de uma expressão utilizando os métodos recursivos com e sem pendência.

Entrada	Saída
8	**5.71536e13**
5	**300.0**
6	**540000.0**

Exercício dois a entrada contém três números inteiros e a saída os elementos e quantas vezes eles apareceram na fatoração.

Entrada	Saída
13,15,45	**13-1** **3-1** **5-1** **3-1** **3-2** **5-1**
9,5,4	**3-1** **3-2** **5-1** **2-0** **2-1**

12,24,6	**2-1** **3-1** **2-1** **2-2** **3-1** **3-1**

No exercício três é pedido um algoritmo capaz de receber três números inteiros e entregar o resultado, contendo o MDC dos mesmos.

Entrada	Saída
108, 135 , 63	**9**
15, 75, 105	**15**
180, 240, 270	**30**

Exercício quatro é pedido uma solução que após receber um número qualquer diga se ele é perfeito, deficiente ou abundante.

Entrada	Saída
28	**Eh perfeito!**
100	**Eh abundante!**
15	**Eh deficiente!**

Exercício cinco não têm entrada do usuário, mas dois números são sorteados e com eles são exibidos os números para três cadeiras.

Números sorteados	Saída
76,16	**1,2,0**
54,48	**0,1,2**
6,77	**0,2,1**

5. Conclusão

Tendo em vista o exposto, este trabalho foi realizado para complementar a primeira nota da disciplina, como já foi apresentado anteriormente, e também como

um meio de exercitar os ensinamentos aprendidos em aula, colocando os em prática, com esses exercícios. O objetivo foi alcançado completamente, ficou bem claro o modo de utilizar os paradigmas e métodos, pelos quais a professora nos ensinou.

Para a resolução dos mesmos fomos obrigados a utilizar o método de recursão com forma de loop já que na linguagem haskell não existe os métodos para e nem o enquanto, presentes em várias linguagens de programação, o que é bom, pois, saímos da nossa zona de conforto, e ainda aprendemos um jeito novo para fazer essa tarefa.