EXERCÍCIOS E PROBLEMAS DE ESTRUTURAS DE DADOS

AUTOR: FABRÍCIO GALENDE MARQUES DE CARVALHO

INTRODUÇÃO

PRÁTICA DE ALGORITMOS E PROGRAMAÇÃO

P.1.1. Crie um programa em typescript e defina:
a) Uma variável que tenha anotação explícita de tipo numérico.
b) Uma variável que tenha anotação explícita de tipo string.
c) Uma variável que não tenha anotação explícita de tipo mas que receba um valor lógico.
d) Uma variável que seja um JSON contendo propriedades de tipo numérico e de tipo string.
e) Um array numérico com anotação explícita de tipo.

Execute cada um desses programas e mostre na saída os valores e os tipos utilizando a função **`typeof()`**

P.1.2. Para os itens do exercício **P.1.1**, qual a diferença com relação à exibição dos tipos quando você passa o cursor sobre a variável durante a sua declaração?

P.1.3 Escreva um programa, em TypeScript, que solicite que o usuário digite dois números e imprima o maior deles.

P.1.4. Escreva um programa, em TypeScript, que solicite que o usuário digite duas letras e diga qual delas vem antes e qual vem depois no alfabeto.

P.1.5. Escreva um programa, em TypeScript, que solicite que o usuário digite duas palavras e diga qual delas aparece antes da outra no dicionário. O programa não deve solicitar nenhuma

informação adicional por parte do usuário e supõe que as palavras são escritas somente com caracteres de ‘a’ a ‘z’. Para esse caso, especifique, também, o algoritmo em pseudocódigo, conforme notação explicada em sala de aula. Utilize o método de string charCodeAt().

P.1.6. Repita o exercício P.1.5 mas utilizado operadores relacionais ao invés do método charCodeAt().

P.1.7. Escreva um programa que exiba um menu contendo 3 alternativas: “1. Dúvidas”, “2.Reclamações”, “3.Sair”. O usuário deve digitar a palavra correspondente à opção do menu e, dependendo da opção, deve ser fornecida uma orientação ao usuário. Exemplo: Caso o usuário digite “Dúvidas” exiba: “Suas dúvidas devem ser encaminhadas para o email duvidas@email.com”. Esse programa deve ser escrito em TypeScript e deve fazer uso do bloco switch.

P.1.8. Escreva um programa, utilizando o bloco for, que imprima todos os múltiplos de 3 contidos entre 0 e 100, inclusive. Os valores devem ser impressos em ordem crescente.

P.1.9. Repita o exercício P.1.8 mas utilizando um bloco while, imprimindo os números em ordem decrescente.

P.1.10. Repita o exercício P.1.9 utilizando um bloco do while.

P1.11. Escreva dois programas equivalentes, em TypeScript, que solicitem que o usuário digite dois números, n1 e n2. O programa deve dizer se o n1 é “menor ou igual a n2” ou se é “maior do que n2”. Um programa deve utilizar o bloco switch e o outro deve utilizar if-else. Qual dificuldade surgiria caso se desejasse que o programa discriminasse 3 condições (>, < ou =)?

P.1.12. Repita o exercício P.1.11 utilizando uma chamada à função e uma estrutura if de comparação interna entre os números. A entrada para a função deve ser os números e a saída deve ser uma string contendo uma mensagem informativa da relação entre esses números (>, < ou =). Assegure-se de que a interface da função não é quebrada e de que o código é devidamente modularizado. A função deve ser definida em um arquivo e invocada em outro (use export e import).

P.1.13. Modele, utilizando orientação a objetos, um usuário de um sistema que tenha preenchido as seguintes informações em uma interface de cadastro: nome, ano de nascimento, cpf e gênero. Esse usuário deve possuir um método chamado *equals*, que compara uma instância da classe com outra passada como argumento para o método *equals* e outro método chamado *speak_name* que retorna a string representativa do nome do usuário. Demonstre a execução de um programa que faça uso dessa classe, exibindo resultados no console.

P.1.14. Demonstre como uma variável declarada com escopo de bloco pode levar a uma obtenção errada de resultado quando existe uma função ou método que faz uso de um valor armazenado em uma variável global. Ilustre isso com uma função simples e diga como esse tipo de construção de programa viola um bom projeto de função. Como esse tipo de problema poderia ser resolvido?

P.1.15. Crie um programa que contenha o seguinte: Num arquivo que contenha modelo, defina uma variável do tipo cadeia de caracteres, uma variável do tipo numérico, uma variável do tipo array de cadeia de caracteres, uma variável do tipo array de números, um objeto que modele uma pessoa (que tenha nome e idade). Nesse mesmo arquivo, inicialize valores específicos para cada uma dessas variáveis e exporte essa variáveis para outro arquivo de mesmo nome mas com sufixo _view. Crie uma função que receba cada uma dessas variáveis como argumentos (parâmetros formais da função) e, internamente, altere o valor de cada um dos parâmetros passados. Imprima os valores das variáveis, no arquivo _view, antes e depois de chamar a função. Qual sua conclusão a respeito? OBS: não viole as interfaces de entrada e saída da função, conforme instruído em sala de aula.

TERMINOLOGIA E CONCEITOS

TC.1.1. Complete os seguintes parágrafos com termos utilizados na área de algoritmos e estruturas de dados.

Um algoritmo corresponde a um conjunto de _____(1)_____ a serem efetuados para a resolução de um problema. Os _____(2)_____ ou _____(3)_____ definidos em um algoritmo devem ser _____(4)_____, ou seja, o algoritmo deve possuir início e fim bem definidos. Algoritmos podem ser vistos também como _____(5)_____ funcionais que transformam um conjunto de _____(6)_____ em um conjunto de _____(7)_____.

Os blocos fundamentais que podem constituir um algoritmo incluem tipicamente _____(8)_____, _____(9)_____, _____(10)_____, _____(11)_____, _____(12)_____ e _____(13)_____.

Sob o ponto de vista de entrada e saída, pode-se dizer que algoritmos _____(14)_____ são aqueles que obtém uma mesma correspondência _____(15)_____considerando as mesmas _____(16)_____.

ALGORITMOS RECURSIVOS

PRÁTICA DE ALGORITMOS E PROGRAMAÇÃO

P.2.1. Desenvolva um programa recursivo para calcular o menor elemento presente em um array não ordenado.
P.2.2. Desenvolva um programa recursivo que calcule o elemento como maior valor absoluto presente em um array não ordenado.
P.2.3. Desenvolva um programa recursivo que calcule o fatorial do n-ésimo número. Faça a análise de desempenho do programa através da criação do cache para chamadas repetidas à função utilizando o mesmo argumento. Considere que a função é um método de uma classe denominada de Factorial. Crie um gráfico que mostre a evolução do tempo de execução para repetidas chamadas à mesma função, comparando a versão com e sem cache, aumentando-se o número de vezes que a função e invocada, fixando-se n (preferencialmente um valor alto que não gere estouro de pilha).

Número de vezes que a função é chamada, para n fixo	Recursiva (tempo ms)	Recursiva com cache (tempo ms)
1		
10		
1000		
.....		

P.2.4. Desenvolva um programa que calcule os elementos da sequência de Fibonacci e que exiba na tela a árvore de chamadas “aproximada”.
P.2.5. Para o item 2.4. Faça uma análise de desempenho para diferentes valores de n (termo da sequência) utilizando orientação a objetos e criação de um cache de valores. Crie um gráfico comparativo tal como descrito no exercício **P.2.3.**
P.2.6. Defina uma classe denominada MyArray que contenha um método recursivo que retorne a soma de todos os elementos presentes no array.
P.2.7. Defina uma classe MyArray que contenha um método que imprima de modo recursivo todos os elementos do array e que contenha outro método, também recursivo, que retorne os elementos do array em ordem reversa.
P.2.8. Defina uma classe que modele uma caixa d’água. A caixa deve conter uma determinada quantidade de líquido (em litros). Defina um método, que retorna o valor total em litros, obtido de modo recursivo extraindo litro a litro a capacidade da caixa d´água até esgotar sua capacidade (esvaziar).
P.2.10. Utilizando um código recursivo desenvolvido durante as aulas ou em qualquer um dos exercícios anteriores, envolvendo passagem de array à função recursiva, crie um programa que imprima uma pilha de execução. Considere que cada dado presente na pilha de execução, para sucessivas chamadas usando array, é representado utilizando um asterisco “*”. Compare o resultado considerando uma função que faz chamada recursiva simples (ex. fatorial ou progressão aritmética) com uma função que faz chamada recursiva dupla (ex. sequência de Fibonacci). A Figura abaixo mostra um exemplo de saída de execução esperado para a obtenção da pilha de execução do programa recursivo que obtém o máximo elemento em um array de 5 elementos.

```
Array original:
[ 1, 4, 10, 20, -1 ]
*****|
*********|
**********|
***********|
************|
**********|
**********|
******|
*|
O maior número do array é:  20
```

TERMINOLOGIA E CONCEITOS

TC.2.1. Complete o seguinte parágrafo com termos utilizados na área de estruturas de dados referente ao desenvolvimento de software utilizando recursão.

É importante, para qualquer profissional da área de computação que trabalhe com desenvolvimento, que os conceitos de ____(1)_____ e _____(2)_____ sejam bem diferenciados. Um ____(3)_____ corresponde a um _____(4)_____ geral e demanda uma ____(5)_____. Já uma ____(6)_____ corresponde a um _____(7)_____ do problema.

Um _____(8)_____ é resolvido desmembrando-se o problema em um problema menor até que se chegue a menor instância possível, chamada de _____(9)_____. A solução para o problema é então obtida combinando-se sucessivamente o _____(10)_____ a _____(11)_____ até que se chegue a instância original.

A ____(12)_____ diz respeito a quantidade de ____(13)_____ utilizada por um programa. A _____(14)_____ diz respeito às _____(15)_____ envolvidas no programa, já a _____(16)_____ diz respeito aos dados temporários que são necessários para que o programa inicie e termine sua execução. Alterar a maneira como um algoritmo é codificado para um programa pode _____(17)_____ ou _____(18)_____ sua complexidade _____(19)_____ e sua complexidade _____(20)_____.

TC.2.2. Dê um exemplo de algoritmo recursivo que apresente complexidade espacial superior a outro, também recursivo, porém distinto. Ilustre esboçando os dados de programa e entrada na pilha de execução. **OBS:** Seu exemplo não precisa conter código de programação mas sim explicar o porquê da complexidade computacional superior utilizando pseudocódigo para representar os dados analisados.

TC.2.3. Dê um exemplo de algoritmo recursivo que apresente complexidade espacial superior a outro em decorrência principalmente a dados internos da função e não como decorrência da entrada. **OBS:** Seu exemplo não precisa conter código de programação mas sim explicar o porquê da complexidade computacional superior utilizando pseudocódigo para representar os dados analisados.