

UNIVERSIDADE CESUMAR - UNICESUMAR

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Curso:** Engenharia de Software | | | | | | | | **Série:** 6S | | | | **Turma:** A | | | | | **Turno:** Noite | | |
| **Professor(a):** Thiago Bussola da Silva | | | | | | | | **Horário:** | | | | | | | | | | | |
| **Acadêmico (a): José Ricardo Miessi Gomes** | | | | | | | | | | | | | | | | **RA: 21105683-2** | | | |
| **Disciplina:** Paradigmas de Programação | | | | | | | | | | | | | | | | **Data: 26/09/2023** | | | |
| **Prova** | | **Prova Prática** | | | **Atividades de estudo**  **programadas (AEP)** | | | | | **Prova integrada** | | | | **Nota final do bimestre** | | | | | |
|  | |  | | |  | | | | |  | | | |  | | | | | |
|  | |  | | |  | | | | |  | | | |  | | | | | |
| **INSTRUÇÕES PARA REALIZAÇÃO DA PROVA:**  ⇒ Os dados do cabeçalho deverão ser preenchidos com letra maiúscula. E as questões deverão ser respondidas com letra legível.  ⇒ É vedado, durante a prova, o porte e/ou o uso de aparelhos sonoros, fonográficos, de comunicação ou de registro eletrônico ou não, tais como: notebooks, celulares, tablets e similares.  ⇒ A prova é individual e sem consulta, deverá ser respondida a caneta azul ou preta. Prova escrita a lápis não dá direito à revisão. Não é permitido o uso de corretivo.  ⇒ É obrigatória a permanência do acadêmico 1 (uma) hora em sala de aula após o início da prova.  ⇒ Não será permitida a entrada na sala de aula após 10 minutos do início da prova.  ⇒ É obrigatória a assinatura da lista de presença impressa na qual constam RA, nome e curso.  ⇒ O valor de cada questão está ao lado da mesma.  ⇒ Todas as respostas devem constar no espaço destinado e autorizado pelo professor, à resposta.  ⇒ Em caso de qualquer irregularidade comunicar ao Professor ou fiscal de sala.  ⇒ Ao término da prova, levante o braço e aguarde o atendimento do professor ou do fiscal. | | | | | | | | | | | | | | | | | | | |
| 1ºbim. |  | | 2ºbim. |  | | 1ªsub. |  | | 2ªsub. | |  | | 1ºsem. | |  | | | 2º sem. |  |

|  |  |
| --- | --- |
| **QUADRO PARA O PROFESSOR - REGISTRO DE NOTAS** | |
| Questão 1 |  |
| Questões 2 |  |
| Questão 3 |  |
| Questão 4 |  |
| Questão 5 |  |
| Questão 6 |  |
| Questões 7 |  |
| Questões 8 |  |
| Questão 9 |  |
| Questão 10 |  |

## Instruções - Leia com atenção!

**Preencha os campos do cabeçalho da prova**

**Regras para a prova.**

Os únicos sites que você pode acessar para consultar suas dúvidas sobre sintaxe são:

<https://elixirschool.com/pt/lessons/basics/documentation>

https://elixir-lang.org/docs.html

O uso de qualquer outro site, chat GPT, Github está **proibido**, caso o aluno acesse outra fonte de pesquisa a prova será zerada.

Compiladores: Será permitido o uso de compiladores online para que você possa validar a implementação das soluções propostas para os exercícios. Você está autorizado a utilizar os seguintes compiladores:

<https://www.tutorialspoint.com/execute_elixir_online.php>

<https://onecompiler.com/elixir>

O uso do **Replit** **não está autorizado** e caso o aluno acesse essa ferramenta a prova será zerada.

Você pode criar arquivos .exs para a resolução da prova e fazer o zip para enviar eles. Ou você pode copiar o código de resposta e colar abaixo da pergunta correspondente no arquivo .docx

Você pode converter sua prova para pdf ao enviar, lembre-se de enviar os arquivos .exs ou de colocar as respostas na prova para a entrega.

Caso você não entregue o arquivo .docx / pdf e os .exs (caso tenha seja de sua preferência) a prova será zerada.

**Questão 1 - [1 ponto] -** Explique a diferença entre funções puras e funções de ordem superior em programação funcional. Dê exemplos de cada uma.

Funções puras são funções que executam apenas um comando, enquanto as funções de ordem superior são funções que chamam outras funções para executar um determinado comando, ou seja, são funções que usam da recursividade para serem executadas.

Um exemplo de função de ordem superior é: tenho uma função que requisita a entrada de um número e outra função que calcula o dobro de um número. Estarei aqui usando uma função para a entrada de um número e chamando outra função para realizar a operação matemática.

Enquanto uma função pura, pode ser simplesmentes uma função que calcula a + b e te devolve o resultado.

**Questão 2 - [1 ponto] -** Discorra sobre as vantagens da linguagem de programação elixir e em que tipo de projeto ou cenário devemos optar pelo uso dessa tecnologia.

Uma linguagem como o Elixir que é uma linguagem funcional, leva algumas vantagens em relação a uma linguagem orientada a objetos a depender dos objetivos do projeto em questão, tais vantagens podem ser citadas como:

* Imutabilidade, os dados no elixir não são mutáveis, eles são transformados em novos dados, mas sem perder a sua originalidade.
* O Elixir tem foco nas funções, os problemas são quebrados em várias funções mais simples.
* É também fácil de se realizar testes, com a natureza do elixir sendo quebrar um problema grande em funções pequenas isso facilita a realização dos testes.

A linguagem Elixir vai bem em projetos complexos, onde os problemas podem ser “quebrados” em problemas menores, criando funções para resolver esses problemas menores, realizando testes com mais facilidade e evitando a ocorrência de bugs.

**Questão 3 - [0,5 pontos] -** Escreva uma função que verifique se um número é par.

defmodule Par do

def par(a) do

rem(a, 2) == 0

end

end

IO.puts Par

**Questão 4 - [0,5 pontos] -** Implemente uma função que calcule o dobro de cada elemento em uma lista.

lista = [1,2,3,4,5,6]

dobro = Enum.map(lista, fn item -> item \* 2 end)

**Questão 5 - [0,5 ponto] -** Crie uma função que retorne o último elemento de uma lista.

**Questão 6 - [1 ponto] -** Implemente uma função que calcule o fatorial de um número usando recursão.

**Questão 7 - [1 ponto] -**  Escreva uma função que aplique uma função passada como argumento a cada elemento de uma lista.

**Questão 8 - [1 ponto] -** Escreva uma função que filtre os elementos de uma lista com base em uma função de filtro passada como argumento.

lista = [1,2,3,4,5,6]

EntreN = Enum.filter(lista, fn x -> x > 2 && x < 6 end)

**Questão 9 - [1 ponto] -** Crie uma função que gere os primeiros "n" números da sequência de Fibonacci.  
  
defmodule fibonacci do

def fib(0), do: 0

def fib(1), do: 1

def fibonacci do:

fib(n-1)+fib(n-2)

end

IO.puts fibonacci(5)

**Questão 10 - [2,5 ponto] -** Crie uma função que calcule a média de uma turma.  
O exemplo abaixo demonstra como são passadas as notas dos alunos pertencentes a uma turma.  
Você deve utilizar o método reduce para calcular a média.

notas\_da\_turma = [

{"Alice", [9.5, 8.0, 7.5]},

{"João", [8.0, 7.0, 6.5]},

{"Pedro", [9, 9.5, 9.0]},

{"Lucas", []},

]

Dicas: Utilize map ou flatmap para extrair todas as notas dos alunos em uma única lista