

Nome: Luis Eduardo da S. Moraes

Nome: Bianca Lucas da S. Caçula

EXERCÍCIOS E PROBLEMAS DE MATEMÁTICA PARA A COMPUTAÇÃO AUTOR: FABRÍCIO GALENDE MARQUES DE CARVALHO

1. INTRODUÇÃO

2. DEFINIÇÕES MATEMÁTICAS

2.1. Efetue uma definição, através de enumeração, de todos os membros da sua família, considerando apenas parentesco direto de pais, irmãos e filhos.

R:

Família Moraes $\stackrel{\text{def}}{=}$ Monica Luana da Silva Moraes, Everthon Moraes Silva Fernandes, Luis Eduardo da Silva Moraes

Família Silva $\stackrel{\text{def}}{=}$ Claudiney Aparecido da Silva Caçula, Edávila Lucas da Silva, Bianca Lucas da Silva Caçula, Vinícius Lucas da Silva Caçula

2.2. Efetue uma definição, através de enumeração, de cursos existentes na instituição de ensino em que você está estudando.

R: Cursos Fatec Jacaré $\stackrel{\text{def}}{=}$ Meio Ambiente e Recursos Hídricos, Geoprocessamento, Desenvolvimento de Software Multiplataforma

2.3. Efetue uma definição, através de enumeração, de números primos. Um número é dito primo se só possui como divisor ele mesmo e o número um.

R: Números Primos $\stackrel{\text{def}}{=}$ 2, 3, 5, 7, 11, ...

2.4. Efetue uma definição ostensiva, de um colega de classe.



R: Luis Eduardo da Silva Moraes $\stackrel{\text{def}}{=}$



Bianca Lucas da Silva Caçula ^{def}

2.5. Efetue uma definição recursiva de uma sequência numérica cujos elementos são obtidos multiplicando-se, a partir do segundo elemento, o elemento anterior por 3. Considere que o primeiro elemento vale 2.

R: Sequência Exponencial de Multiplicação por 3 com base 2 ^{def} 2, 6, 18, 54, 162, ...

2.6. Utilizando linguagem de programação defina:

a) Uma enumeração para os meses do ano.

R: Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_6/exercicio2_6_a.ts

```
import promptSync from "prompt-sync";
const prompt = promptSync();

enum mesesAno {
  jan = "Janeiro",
  fev = "Fevereiro",
  mar = "Março",
  abr = "Abril",
  mai = "Maio",
  jun = "Junho",
  jul = "Julho",
  ago = "Agosto",
  set = "Setembro",
  out = "Outubro",
  nov = "Novembro",
  dez = "Dezembro"
};

function obterNomeCompleto(mes: mesesAno): string {
  return mes;
}

const abreviacao = prompt("Digite a abreviação do mês: ").toLowerCase();
const mesTeste: mesesAno = mesesAno[abreviacao as keyof typeof mesesAno];
const nomeCompleto = obterNomeCompleto(mesTeste);

if (mesTeste) {
  console.log(`Abreviação: ${abreviacao}`);
  console.log(`Nome completo: ${nomeCompleto}`);
} else {
  console.log("Abreviação inválida. Certifique-se de digitar uma abreviação válida.");
}
```

```

PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 26a

> definicoesmatematicas@1.0.0 26a
> npx ts-node ./src/exercicio2_6/exercicio2_6_a

Digite a abreviação do mês: fev
Abreviação: fev
Nome completo: Fevereiro

```

b) Uma enumeração para os dias da semana.

R: Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_6/exercicio2_6_b.ts

```

1  import promptSync from "prompt-sync";
2  const prompt = promptSync();
3
4  enum diaSemana {
5      dom = "Domingo",
6      seg = "Segunda",
7      ter = "Terça",
8      qua = "Quarta",
9      qui = "Quinta",
10     sex = "Sexta",
11     sab = "Sábado"
12 };
13
14 function obterNomeCompleto(dia: diaSemana): string {
15     return dia;
16 }
17
18 const abreviacao = prompt("Digite a abreviação do dia da semana: ").toLowerCase();
19 const diaTeste: diaSemana = diaSemana[abreviacao as keyof typeof diaSemana];
20 const nomeCompleto = obterNomeCompleto(diaTeste);
21
22 if (diaTeste) {
23     console.log(`Abreviação: ${abreviacao}`);
24     console.log(`Nome completo: ${nomeCompleto}`);
25 } else {
26     console.log("Abreviação inválida. Certifique-se de digitar uma abreviação válida.");
27 }

```

```

PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 26b

> definicoesmatematicas@1.0.0 26b
> npx ts-node ./src/exercicio2_6/exercicio2_6_b

Digite a abreviação do dia da semana: qua
Abreviação: qua
Nome completo: Quarta

```

c) Uma função recursiva para o cálculo do fatorial de um número.

R: Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_6/exercicio2_6_c.ts

```
src > exercicio2_6 > TS exercicio2_6_c.ts > ...
1  import promptSync from "prompt-sync";
2  const prompt = promptSync();
3
4  function calcularFatorial(numero: number): number {
5      if (numero == 0 || numero == 1) {
6          return 1;
7      } else {
8          return numero * calcularFatorial(numero - 1);
9      }
10 }
11
12 const numero: number = parseInt(prompt("Digite seu número: "));
13 const fatorial = calcularFatorial(numero);
14 console.log(`O fatorial de ${numero} é ${fatorial}`);
```

```
PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 26c
```

```
> definicoesmatematicas@1.0.0 26c
> npx ts-node ./src/exercicio2_6/exercicio2_6_c
```

```
Digite seu número: 7
O fatorial de 7 é 5040
```

d) Uma definição que corresponda a definição do tipo gênero-diferença para um uma pessoa que estude em uma faculdade. Utilize uma linguagem que dê suporte a herança.

R: Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_6/exercicio2_6_d.ts

```
src > exercicio2_6 > TS exercicio2_6_d.ts > ...
22 class fisico extends cursando {
23     boasnotas: boolean;
24     constructor(estado: string, caracteristica: string, boasnotas: boolean) {
25         super(estado, caracteristica);
26         this.boasnotas = boasnotas;
27     }
28     notas() {
29         console.log("O aluno tem boas notas.");
30     }
31 }
32
33 const seraluno = new fisico("Einstein", "Noite", true);
34 console.log(`Aluno ${seraluno.estado}`);
35 seraluno.estudar();
36 console.log(`Horário: ${seraluno.caracteristica}`);
37 seraluno.horario();
38 console.log(`Passou de semestre? ${seraluno.boasnotas}`);
39 seraluno.notas();
```

```
PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 26d
```

```
> definicoesmatematicas@1.0.0 26d
> npx ts-node ./src/exercicio2_6/exercicio2_6_d
```

```
Aluno Einstein
O aluno ainda está cursando a faculdade.
Horário: Noite
O aluno estuda no horário da noite.
Passou de semestre? true
O aluno tem boas notas.
```

2.7. Um veículo possui a capacidade de se mover, expressa pela alteração na sua coordenada de longitude e latitude. Um veículo elétrico é um veículo que possui como fonte de energia primária a eletricidade (armazenada em uma bateria). Um veículo elétrico e voador é um veículo que também possui a capacidade de se mover na vertical, expressa pela alteração de sua altitude em relação ao solo. Represente um veículo elétrico e voador utilizando uma cadeia de herança. Defina o código-fonte representativo do modelo em um arquivo separado daquele que faz uso desse e, adicionalmente exemplifique o acesso e a modificação desses atributos através de chamada de suas operações.

R: Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_6/exercicio2_7_model.ts

```
src > exercicio2_7 > TS exercicio2_7_model.ts > ...
1  class veículo {
2      longitude: number;
3      latitude: number;
4      constructor(longitude: number, latitude: number) {
5          this.longitude = longitude;
6          this.latitude = latitude;
7      }
8
9      moverPara(longitude: number, latitude: number): void {
10         this.longitude = longitude;
11         this.latitude = latitude;
12     }
13 }
14
15 class veiculoEletrico extends veiculo {
16     bateria: number;
17     constructor(longitude: number, latitude: number, bateria: number) {
18         super(longitude, latitude);
19         this.bateria = bateria;
20     }
21     carregar(carga: number): void {
22         this.bateria += carga;
23     }
24 }
25
26 class veiculoEletricoVoador extends veiculoEletrico {
27     altitude: number;
28     constructor(longitude: number, latitude: number, altitude: number, bateria: number) {
29         super(longitude, latitude, bateria);
30         this.altitude = altitude;
31     }
32     moverAltitude(altitude: number){
33         this.altitude = altitude;
34     }
35 }
37 export { veiculo, veiculoEletrico, veiculoEletricoVoador };
```

Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_7/exercicio2_7.ts

```

src > exercicio2_7 > TS exercicio2_7.ts > ...
1  import promptSync from "prompt-sync";
2  import { veiculo, veiculoEletrico, veiculoEletricoVoador } from "../exercicio2_7_model";
3  const prompt = promptSync();
4
5  console.log("Criando um veiculo...");
6  const longitudeveiculo = parseFloat(prompt("Digite a longitude: "));
7  const latitudeveiculo = parseFloat(prompt("Digite a latitude: "));
8  const meuveiculo = new veiculo(longitudeveiculo, latitudeveiculo);
9  console.log("Veículo criado:", meuveiculo);
10 console.log("\nMovendo o veículo...");
11 const novalongitude = parseFloat(prompt("Digite a nova longitude: "));
12 const novalatitude = parseFloat(prompt("Digite a nova latitude: "));
13 meuveiculo.moverPara(novalongitude, novalatitude);
14 console.log("Coordenadas do veículo após movimentação:", meuveiculo.longitude, meuveiculo.latitude);
15
16 console.log("Criando um veículo elétrico...");
17 const longitudeEletrico = parseFloat(prompt("Digite a longitude: "));
18 const latitudeEletrico = parseFloat(prompt("Digite a latitude: "));
19 const bateriaEletrico = parseFloat(prompt("Digite a capacidade da bateria: "));
20 const meuveiculoEletrico = new veiculoEletrico(longitudeEletrico, latitudeEletrico, bateriaEletrico);
21 meuveiculoEletrico.carregar(20);
22 console.log("Veículo elétrico criado:", meuveiculoEletrico);
23 console.log("\nMovendo o veículo elétrico...");
24 const novalongitudeEletrico = parseFloat(prompt("Digite a nova longitude: "));
25 const novalatitudeEletrico = parseFloat(prompt("Digite a nova latitude: "));
26 meuveiculoEletrico.moverPara(novalongitudeEletrico, novalatitudeEletrico);
27 console.log("Coordenadas do veículo elétrico após movimentação:", meuveiculoEletrico.longitude, meuveiculoEletrico.latitude);
28 console.log("\nVerificando a carga da bateria do veículo elétrico...");
29 console.log("Bateria do veículo elétrico:", meuveiculoEletrico.bateria);
30 console.log("\nCarregando o veículo elétrico...");
31 const cargaParaCarregar = parseFloat(prompt("Digite a carga para carregar: "));
32 meuveiculoEletrico.carregar(cargaParaCarregar);
33 console.log("Bateria após carregar:", meuveiculoEletrico.bateria);
34
35 console.log("Criando um veículo elétrico voador...");
36 const longitudeVoador = parseFloat(prompt("Digite a longitude: "));
37 const latitudeVoador = parseFloat(prompt("Digite a latitude: "));
38 const altitudeVoador = parseFloat(prompt("Digite a altitude: "));
39 const bateriaVoador = parseFloat(prompt("Digite a capacidade da bateria: "));
40 const meuveiculoEletricoVoador = new veiculoEletricoVoador(longitudeVoador, latitudeVoador, bateriaVoador, altitudeVoador);
41 console.log("Veículo elétrico voador criado:", meuveiculoEletricoVoador);
42 console.log("\nMovendo o veículo elétrico voador...");
43 const novalongitudeVoador = parseFloat(prompt("Digite a nova longitude: "));
44 const novalatitudeVoador = parseFloat(prompt("Digite a nova latitude: "));
45 const novaAltitudeVoador = parseFloat(prompt("Digite a nova altitude: "));
46 meuveiculoEletricoVoador.moverPara(novalongitudeVoador, novalatitudeVoador, novaAltitudeVoador);
47 console.log("\nAlterando altura de vôo...");
48 meuveiculoEletricoVoador.moverAltitude(novaAltitudeVoador);
49 console.log("Coordenadas do veículo elétrico voador após movimentação:", meuveiculoEletricoVoador.longitude, meuveiculoEletricoVoador.latitude, meuveiculoEletricoVoador.altitude);
50 console.log("\nVerificando a carga da bateria do veículo elétrico voador...");
51 console.log("Bateria do veículo elétrico voador:", meuveiculoEletricoVoador.bateria);
52 console.log("\nCarregando o veículo elétrico voador...");
53 const cargaParaCarregarVoador = parseFloat(prompt("Digite a carga para carregar: "));
54 meuveiculoEletricoVoador.carregar(cargaParaCarregarVoador);
55 console.log("Bateria após carregar:", meuveiculoEletricoVoador.bateria);

```

```
PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 27
```

```
> definicoesmatematicas@1.0.0 27
> npx ts-node ./src/exercicio2_7/exercicio2_7
```

```
Criando um veículo...
```

```
Digite a longitude: 0
```

```
Digite a latitude: 0
```

```
Veículo criado: veiculo { longitude: 0, latitude: 0 }
```

```
Movendo o veículo...
```

```
Digite a nova longitude: 10
```

```
Digite a nova latitude: 10
```

```
Coordenadas do veículo após movimentação: 10 10
```

```
Criando um veículo elétrico...
```

```
Digite a longitude: 0
```

```
Digite a latitude: 0
```

```
Digite a capacidade da bateria: 10
```

```
Veículo elétrico criado: veiculoEletrico { longitude: 0, latitude: 0, bateria: 30 }
```

```
Movendo o veículo elétrico...
```

```
Digite a nova longitude: 20
```

```
Digite a nova latitude: 20
```

```
Coordenadas do veículo elétrico após movimentação: 20 20
```

```
Verificando a carga da bateria do veículo elétrico...
```

```
Bateria do veículo elétrico: 30
```

```
Carregando o veículo elétrico...
```

```
Digite a carga para carregar: 30
```

```
Bateria após carregar: 60
```

```
Criando um veículo elétrico voador...
```

```
Digite a longitude: 40
```

```
Digite a latitude: 50
```

```
Digite a altitude: 71
```

```
Digite a capacidade da bateria: 89
```

```
Veículo elétrico voador criado: veiculoEletricoVoador {
```

```
  longitude: 40,
```

```
  latitude: 50,
```

```
  bateria: 71,
```

```
  altitude: 89
```

```
}
```

```
Movendo o veículo elétrico voador...
```

```
Digite a nova longitude: 31
```

```
Digite a nova latitude: 21
```

```
Digite a nova altitude: 11
```

```
Alterando altura de vôo...
```

```
Coordenadas do veículo elétrico voador após movimentação: 31 21 11
```

```
Verificando a carga da bateria do veículo elétrico voador...
```

```
Bateria do veículo elétrico voador: 71
```

```
Carregando o veículo elétrico Voador...
```

```
Digite a carga para carregar: 43
```

```
Bateria após carregar: 114
```

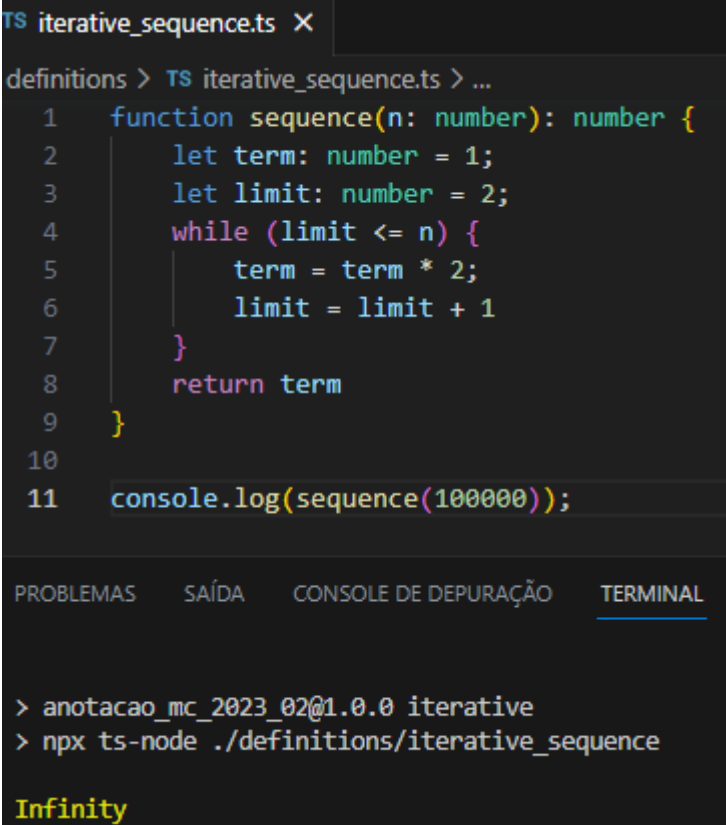
2.8. O que acontece ao se executar uma chamada a uma função recursiva que chama a si mesma um elevado número de vezes? Dê um exemplo utilizando o código-fonte da progressão aritmética fornecido pelo professor. Faça um comparativo escrevendo um algoritmo e código que sejam equivalentes ao recursivo em termos de entradas e saídas, mas que utilizem iteração ao invés de recursão. Qual sua conclusão?

R: Ela acabando não sendo executada até o final e exibe a mensagem “infinity”.

Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_8/exercicio2_8_a.ts

./definicoesmatematicas/codigos/src/exercicio2_8/exercicio2_8_b.ts



The image shows a screenshot of a code editor with a dark theme. The editor has a tab titled 'TS iterative_sequence.ts' with a close button. The code in the editor is as follows:

```
definitions > TS iterative_sequence.ts > ...
1  function sequence(n: number): number {
2      let term: number = 1;
3      let limit: number = 2;
4      while (limit <= n) {
5          term = term * 2;
6          limit = limit + 1
7      }
8      return term
9  }
10
11  console.log(sequence(100000));
```

Below the code editor, there is a panel with four tabs: 'PROBLEMAS', 'SAÍDA', 'CONSOLE DE DEPURAÇÃO', and 'TERMINAL'. The 'TERMINAL' tab is selected and shows the following commands and output:

```
> anotacao_mc_2023_02@1.0.0 iterative
> npx ts-node ./definitions/iterative_sequence

Infinity
```



```
definitions > TS recursive_sequence.ts > ...
1  function sequence(n: number): number {
2      if (n == 1) {
3          return 1;
4      } else {
5          return 2 * sequence(n - 1);
6      }
7  }
8
9  console.log(sequence(100000));
10
11 export{};
```

PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO **TERMINAL** powershell + v [] [X]

Infinity
PS C:\Users\bicas\OneDrive\Área de Trabalho\Nova pasta> npm run recursive

> anotacao_mc_2023_02@1.0.0 recursive
> npx ts-node ./definitions/recursive_sequence

C:\Users\bicas\OneDrive\Área de Trabalho\Nova pasta\definitions\recursive_sequence.ts:2
if (n == 1) {
^
RangeError: Maximum call stack size exceeded

A conclusão que se chega ao realizar códigos equivalentes iterativo e recursivo é que ambos os casos o código não é executado até o final por existir muitos caracteres para serem executados.

2.9. Uma progressão geométrica é uma sequência numérica onde cada elemento, a partir do segundo, é obtido multiplicando-se o anterior por uma constante. Utilizando uma linguagem de programação que dê suporte a orientação a objetos, defina uma progressão geométrica e dê exemplo de geração de seus primeiros 50 termos.

R:

Disponível em:

[./definicoesmatematicas/codigos/src/exercicio2_9.ts](#)

```
1  class ProgressaoGeometrica {
2      primeiroTermo: number;
3      razao: number;
4
5      constructor(primeiroTermo: number, razao: number) {
6          this.primeiroTermo = primeiroTermo;
7          this.razao = razao;
8      }
9
10     gerarTermos(quantidade: number): number[] {
11         const termos: number[] = [this.primeiroTermo];
12         for (let i = 1; i < quantidade; i++) {
13             const proximoTermo = termos[i - 1] * this.razao;
14             termos.push(proximoTermo);
15         }
16         return termos;
17     }
18 }
19
20 // Criando uma progressão geométrica com primeiro termo 2 e razão 3
21 const progressao = new ProgressaoGeometrica(2, 3);
22
23 // Gerando os primeiros 50 termos da progressão
24 const primeiros50Termos = progressao.gerarTermos(50);
25
26 // Exibindo os primeiros 50 termos
27 primeiros50Termos.forEach((termo, index) => {
28     console.log(`Termo ${index + 1}: ${termo}`);
29 });
30
```

```
PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 29
```

```
> definicoesmatematicas@1.0.0 29  
> npx ts-node ./src/exercicio2_9
```

```
Termo 1: 2  
Termo 2: 6  
Termo 3: 18  
Termo 4: 54  
Termo 5: 162  
Termo 6: 486  
Termo 7: 1458  
Termo 8: 4374  
Termo 9: 13122  
Termo 10: 39366  
Termo 11: 118098  
Termo 12: 354294  
Termo 13: 1062882  
Termo 14: 3188646  
Termo 15: 9565938  
Termo 16: 28697814  
Termo 17: 86093442  
Termo 18: 258280326  
Termo 19: 774840978  
Termo 20: 2324522934  
Termo 21: 6973568802  
Termo 22: 20920706406  
Termo 23: 62762119218  
Termo 24: 188286357654  
Termo 25: 564859072962  
Termo 26: 1694577218886  
Termo 27: 5083731656658  
Termo 28: 15251194969974  
Termo 29: 45753584909922  
Termo 30: 137260754729766  
Termo 31: 411782264189298  
Termo 32: 1235346792567894  
Termo 33: 3706040377703682  
Termo 34: 11118121133111046  
Termo 35: 33354363399333136  
Termo 36: 100063090197999410  
Termo 37: 300189270593998200  
Termo 38: 900567811781994600  
Termo 39: 2701703435345984000  
Termo 40: 8105110306037953000  
Termo 41: 24315330918113858000  
Termo 42: 72945992754341580000  
Termo 43: 218837978263024730000  
Termo 44: 656513934789074200000  
Termo 45: 1.9695418043672227e+21  
Termo 46: 5.908625413101668e+21  
Termo 47: 1.7725876239305005e+22  
Termo 48: 5.317762871791501e+22  
Termo 49: 1.5953288615374503e+23  
Termo 50: 4.7859865846123505e+23
```

2.10. A sequência de Fibonacci é definida da seguinte forma: $\{1, 1, 2, 3, 5, 8, 13, \dots\}$, ou seja, para $k > 2$, $x_k = x_{k-1} + x_{k-2}$. Utilizando uma linguagem de programação com suporte a orientação a objetos, defina uma classe que modele a sequência de Fibonacci e exemplifique o cálculo de alguns de seus termos. Ilustre a chamada recursiva e identifique chamadas repetidas a um mesmo valor.

R:

Disponível em:

./definicoesmatematicas/codigos/src/exercicio2_10.ts

```
1  class Fibonacci {
2      private memo: number[] = [];
3
4      calcularTermo(posicao: number): number {
5          if (posicao <= 0) {
6              throw new Error("A posição deve ser um número positivo.");
7          }
8
9          if (this.memo[posicao] !== undefined) {
10             console.log(`Valor já calculado para posição ${posicao}: ${this.memo[posicao]}`);
11             return this.memo[posicao];
12         }
13
14         if (posicao === 1 || posicao === 2) {
15             return 1;
16         }
17
18         const termo = this.calcularTermo(posicao - 1) + this.calcularTermo(posicao - 2);
19         this.memo[posicao] = termo;
20
21         return termo;
22     }
23 }
24
25 const fibonacci = new Fibonacci();
26
27 console.log("Termo na posição 6:", fibonacci.calcularTermo(6));
28 console.log("Termo na posição 8:", fibonacci.calcularTermo(8));
29 console.log("Termo na posição 10:", fibonacci.calcularTermo(10));
```

PS C:\Users\LAB-48\Downloads\mat_comp_2023_02\Unid2\definicoesmatematicas> npm run 210

> definicoesmatematicas@1.0.0 210

> npx ts-node ./src/exercicio2_10

Valor já calculado para posição 3: 2
Valor já calculado para posição 4: 3
Termo da posição 6: 8
Valor já calculado para posição 6: 8
Valor já calculado para posição 5: 5
Valor já calculado para posição 6: 8
Termo da posição 8: 21
Valor já calculado para posição 8: 21
Valor já calculado para posição 7: 13
Valor já calculado para posição 8: 21
Termo da posição 10: 55