Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problen

Programação Diná

subestrutura otima 2. Obter recursão

3. Algoritmo Bottom-UP

Obter uma solução
 Stima de problema

Aula 18 Programação Dinâmica Subsequência Comum mais Longa

Projeto e Análise de Algoritmos

Professor Eurinardo Rodrigues Costa Universidade Federal do Ceará Campus Russas

2021.1

Subsequência Comum mais Longa

Exemplo

Problema

Programação Dinâmica

- 1. Propriedade de subestrutura ótima
- Obter recursão
- 3. Algoritmo Bottom-UP para valor ótimo
- 4. Obter uma solução ótima do problema

Subsequência Comum mais Longa

Exemplo

Programação Dinâmio

- Propriedade de subestrutura ótima
- Obter recursão
 Algoritmo Bottom-UP
- Obter uma solução
 ótima do problema

Definição (Programação Dinâmica)

Seguir os passos:

- (1) Verificar propriedade de subestrutura ótima
 - "Um pedaço da solução ótima é solução ótima de um pedaço do problema?"
- Obter uma recursão para o valor ótimo do
- Obter um algoritmo Bottom-UP para calcular o valor ótimo
- (4) Obter uma solução ótima do problema.

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo

1 100/61

Programação Dinamica
 Propriedade de

2. Obter recursão

 Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

Exemplo

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo

Programação Dinâmica

subestrutura ótima

Algoritmo Bottom-UP para valor ótimo

Obter uma solução

Subsequência Comum mais Longa

Exemplo

Problei

- Programação Dinâmica
 - Obter recursão
 - 3. Algoritmo Bottom-UP
 - Obter uma solução

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA

Subsequência Comum mais Longa

Exemplo

Proble

- Programação Dinâmica
- Obter recursão
- Algoritmo Bottom-UP
 para valor ótimo
- Obter uma solução

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAAY = GTCGTTCGGAATGCCGTTGCTCTGTAAA

Subsequência Comum mais Longa

Exemplo Problema

Proble

- Programação Dinâmica
 - 2. Obter recursão
 - Algoritmo Bottom-UP
 para valor ótimo
 - Obter uma solução
 Atima de problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAAY = GTCGTTCGGAATGCCGTTGCTCTGTAAASCL = GTCGTCGGAAGCCGGCCGAA

Subsequência Comum mais Longa

Exemplo Problems

Proble

- Programação Dinâmica
 - Obter recursão
 - Algoritmo Bottom-UP
 Para valor étimo
 - Obter uma solução
 Stima de problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA

Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA

SCL = GTCGTCGGAAGCCGGCCGAA

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmica

Propriedade de subestrutura ótima

Algoritmo Bottom-UP para valor ótimo

Obter uma solução
 Atima do problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA SCL = GTCGTCGGAAGCCGGCCGAA PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problema

Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Rottom LIB

4. Obter uma solução

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA SCL = GTCGTCGGAAGCCGGCCGAA

Problema

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problema

Programação Dinâmio

- Propriedade de subestrutura ótima
- Obter recursão
 Algoritmo Bottom-UP
 - 4. Obter uma solução

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA

Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA

SCL = GTCGTCGGAAGCCGGCCGAA

Problema

Entrada

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problema

Programação Dinâmio

subestrutura ótima

Algoritmo Bottom-UP
 Para valor étimo

 Obter uma solução ótima do problema

Problema

Programação Dinâmica
 Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-LIP

 Algoritmo Bottom-UP para valor ótimo

Obter uma solução
 ótima do problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA SCL = GTCGTCGGAAGCCGGCCGAA

Problema

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$

Problema

Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-UP

 Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA SCL = GTCGTCGGAAGCCGGCCGAA

Problema

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$ Objetivo X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA

Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA

Problema

Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-UP

Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

Problema

Exemplo

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$

SCL = GTCGTCGGAAGCCGGCCGAA

Objetivo

Obter uma SCL entre X e Y

Subsequência Comum mais Longa

Exemplo Problema

> Programação Dinâmica 1. Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-UP

para valor ótimo

Obter uma soluçã
 ótima do problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAAY = GTCGTTCGGAATGCCGTTGCTCTGTAAA

SCL = GTCGTCGGAAGCCGGCCGAA

Problema

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$

Objetivo

Obter uma SCL entre X e Y

1. Propriedade de subestrutura ótima

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA

Y = GTCGTTCGGAATGCCGTTGCTCTGTAAA

 Propriedade de subestrutura ótima

Algoritmo Bottom-UP
 Para valor ótimo

para valor ótimo 4. Obter uma solução

Problema

Exemplo

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$ Objetivo

SCL = GTCGTCGGAAGCCGGCCGAA

Obter uma SCL entre X e Y

1. Propriedade de subestrutura ótima

"Um pedaço da **solução ótima** é **solução ótima** de um pedaço do problema?"

Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-UP

Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

Exemplo

X = ACCGGTCGAGTGCGCGGAAGCCGGCCGAAY = GTCGTTCGGAATGCCGTTGCTCTGTAAA

SCL = GTCGTCGGAAGCCGGCCGAA

Problema

Entrada

Duas sequências $X[1 \cdots m]$ e $Y[1 \cdots n]$ Objetivo

Obter uma SCL entre X e Y

1. Propriedade de subestrutura ótima

"Um pedaço da **solução ótima** é **solução ótima** de um pedaço do problema?" Sim.

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmic

Propriedade de subsetrutura étima

Obter recursão
 Algoritmo Rottom LIP

Obter uma solução
 śtima da problema

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmio

 Propriedade de subestrutura ótima

Obter recursão
 Algoritmo Bottom-UP
 Para valor étimo

 Obter uma solução ótima do problema

Subsequência Comum mais Longa

Exemplo Problema

- Programação Dinâmica
- Obter recursão
- Algoritmo Bottom-UP para valor ótimo

2. Obter recursão

 $c[i,j] = \text{tamanho da maior SCL entre } X[1 \cdots i] \text{ e } Y[1 \cdots j]$

Subsequência Comum mais Longa

Exemplo Problema

- Programação Dinâmica 1. Propriedade de
- 2. Obter recursão
- para valor ótimo

2. Obter recursão

 $c[i,j] = \text{tamanho da maior SCL entre } X[1\cdots i] \text{ e } Y[1\cdots j]$ Objetivo \rightarrow

Subsequência Comum mais Longa

Exemplo Problema

- Programação Dinâmica

 1. Propriedade de
- 2. Obter recursão
- para valor ótimo 4. Obter uma solução

2. Obter recursão

 $c[i,j] = \text{tamanho da maior SCL entre } X[1\cdots i] \text{ e } Y[1\cdots j]$ Objetivo \rightarrow calcular c[m,n]

2. Obter recursão

 $c[i,j] = \text{tamanho da maior SCL entre } X[1 \cdots i] \text{ e } Y[1 \cdots j]$ Objetivo \rightarrow calcular c[m,n]

$$c[i,j] = \begin{cases} c[i-1,j-1] + 1 &, \text{ se } X[i] = Y[j] \\ \max\{c[i-1,j],c[i,j-1]\} &, \text{ se } X[i] \neq Y[j] \\ 0 &, \text{ se } i = 0 \text{ ou } j = 0 \end{cases}$$

I AA - Aula 10

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problema

- Programação Dinâmica

 1. Propriedade de
- 2. Obter recursão
- Algoritmo Bottom-UP para valor ótimo
- Obter uma solução ótima do problema

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problems

Programação Dinâmica

subestrutura ótima 2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

Obter uma solução
 ótima do problema

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Probler

Programa

Propriedade de

2. Obter recursão

 Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

3. Algoritmo Bottom-UP para valor ótimo

Algoritmo 3: $SCL(X[1 \cdots m], Y[1 \cdots n])$

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmica

2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

Algoritmo 4: $SCL(X[1 \cdots m], Y[1 \cdots n])$

1 criar matriz $c_{m \times n}$

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Probler

Programação Dinâmica

Obter recursão

 Algoritmo Bottom-UP para valor ótimo

Obter uma solução ima do problema

Algoritmo 5: $SCL(X[1 \cdots m], Y[1 \cdots n])$

- 1 criar matriz $c_{m \times n}$
- 2 para $i \leftarrow 0$ até m faça

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Probler

Programação Dinâmio

Obter recursão
 Algoritmo Bottom-UP

para valor ótimo 4. Obter uma solução

Algoritmo 6: $SCL(X[1 \cdots m], Y[1 \cdots n])$

- 1 criar matriz $c_{m \times n}$
- 2 para $i \leftarrow 0$ até m faça $M[i, 0] \leftarrow 0$;

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Probler

Programação Dinâmio

subestrutura ótima

2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

 Obter uma solução tima do problema

Algoritmo 7: $SCL(X[1 \cdots m], Y[1 \cdots n])$

- 1 criar matriz $c_{m \times n}$
- 2 para $i \leftarrow 0$ até m faça $M[i, 0] \leftarrow 0$;
- 3 para $j \leftarrow 0$ até n faça

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Probler

Programação Dinâmio

subestrutura ótima

Algoritmo Bottom-UP para valor ótimo

Obter uma solução
 tima do problema

Algoritmo 8: $SCL(X[1 \cdots m], Y[1 \cdots n])$

- 1 criar matriz $c_{m \times n}$
- 2 para $i \leftarrow 0$ até m faça $M[i, 0] \leftarrow 0$;
- 3 para j ← 0 até n faça M[0,j] ← 0;

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmio

2. Obter recursão

 Algoritmo Bottom-UP para valor ótimo

l. Obter uma solução

```
Algoritmo 9: SCL(X[1 \cdots m], Y[1 \cdots n])
```

- 1 criar matriz $c_{m \times n}$
- 2 para $i \leftarrow 0$ até m faça $M[i, 0] \leftarrow 0$;
- 3 para j ← 0 até n faça M[0,j] ← 0;
- 4 para $i \leftarrow 1$ até m faça

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmica 1. Propriedade de

2. Obter recursão

 Algoritmo Bottom-UP para valor ótimo

4. Obter uma solução ótima do problema

```
Algoritmo 10: SCL(X[1 \cdots m], Y[1 \cdots n])
```

```
1 criar matriz c_{m \times n}

2 para i \leftarrow 0 até m faça M[i,0] \leftarrow 0;

3 para j \leftarrow 0 até n faça M[0,j] \leftarrow 0;

4 para i \leftarrow 1 até m faça

5 para j \leftarrow 1 até n faça
```

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmio

subestrutura ótima

Algoritmo Bottom-UP para valor ótimo

Obter uma solução
 ótima do problema

```
Algoritmo 11: SCL(X[1 \cdots m], Y[1 \cdots n])
```

```
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
               se X[i] = Y[j] então
6
```

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmio

Obter recursão

Algoritmo Bottom-UP para valor ótimo

4. Obter uma solução

Aulas Passadas

Subsequência Comum mais Longa

```
Probler
```

Programação Dinâmi

Obter recursão

Algoritmo Bottom-UP para valor ótimo

```
3. Algoritmo Bottom-UP para valor ótimo
```

```
Algoritmo 12: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
              se X[i] = Y[j] então
6
                    c[i,j] \leftarrow c[i-1,j-1] + 1
```

```
Algoritmo 13: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
              se X[i] = Y[j] então
6
                    c[i,j] \leftarrow c[i-1,j-1] + 1
7
              senão
8
```

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

3. Algoritmo Bottom-UP para valor ótimo

```
Algoritmo 14: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
        para i \leftarrow 1 até n faça
5
              se X[i] = Y[j] então
6
                    c[i,j] \leftarrow c[i-1,j-1] + 1
7
              senão
8
                    se c[i - 1, j] \ge c[i, j - 1] então
```

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmio

2. Obter recursão

 Algoritmo Bottom-UP para valor ótimo

```
Algoritmo 15: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
               se X[i] = Y[j] então
6
                     c[i,j] \leftarrow c[i-1,j-1] + 1
7
               senão
8
                     se c[i - 1, j] \ge c[i, j - 1] então
                          c[i,j] \leftarrow c[i-1,j]
10
```

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problen

Programação Dinâmica

2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

```
Algoritmo 16: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
               se X[i] = Y[j] então
6
                     c[i,j] \leftarrow c[i-1,j-1] + 1
7
               senão
8
                     se c[i - 1, j] \ge c[i, j - 1] então
                          c[i,j] \leftarrow c[i-1,j]
10
                     senão
11
```

Prof. Eurinardo Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmica

Obter recursão
 Algoritmo Bottom-UP

para valor ótimo
4. Obter uma solução
ótima do problema

```
Algoritmo 17: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para i \leftarrow 1 até n faça
5
               se X[i] = Y[j] então
6
                     c[i,j] \leftarrow c[i-1,j-1] + 1
7
               senão
8
                     se c[i - 1, j] \ge c[i, j - 1] então
                          c[i,j] \leftarrow c[i-1,j]
10
                     senão
11
                          c[i,j] \leftarrow c[i,j-1]
12
```

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Problem

Programação Dinâmic

2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

13 retorne c[m, n]

```
Algoritmo 18: SCL(X[1 \cdots m], Y[1 \cdots n])
1 criar matriz c_{m \times n}
2 para i \leftarrow 0 até m faça M[i, 0] \leftarrow 0;
3 para j \leftarrow 0 até n faça M[0, j] \leftarrow 0;
4 para i \leftarrow 1 até m faça
         para j \leftarrow 1 até n faça
5
               se X[i] = Y[j] então
6
                    c[i,j] \leftarrow c[i-1,j-1] + 1
7
               senão
8
                     se c[i - 1, j] \ge c[i, j - 1] então
                          c[i,j] \leftarrow c[i-1,j]
10
                     senão
11
                      c[i,j] \leftarrow c[i,j-1]
12
```

Prof. Eurinardo Aulas Passadas

Subsequência Comum mais Longa

Problen

Programação Dinâmica

Obter recursão
 Algoritmo Bottom-UP

para valor ótimo
4. Obter uma solução ótima do problema

```
Algoritmo 19: SCL(X[1 \cdots m], Y[1 \cdots n])
 1 criar matriz c_{m\times n}, S_{m\times n};
 2 para i \leftarrow 1 até m faça M[i, 0] \leftarrow 0;
 3 para j \leftarrow 1 até n faça M[0, j] \leftarrow 0;
 4 para i \leftarrow 1 até n faça
          para j \leftarrow 1 até m faça
 5
                 se X[i] = Y[j] então
 6
                       c[i,j] \leftarrow c[i-1,j-1] + 1; S[i,j] \leftarrow " ";
 7
                 senão
 8
                       se c[i - 1, j] \ge c[i, j - 1] então
                              c[i,j] \leftarrow c[i-1,j]; S[i,j] \leftarrow "\uparrow";
10
                       senão
11
                         c[i,j] \leftarrow c[i,j-1]; S[i,j] \leftarrow \text{``}\leftarrow\text{''};
12
13 retorne c[m, n], S;
```

1701 Maia 10

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Proble

Programação Dinâmica

Obter recursão
 Algoritmo Bottom-UP

para valor ótimo
4. Obter uma solução

Subsequência Comum mais Longa (SCL)

PAA - Aula 18

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmic

subestrutura ótima 2. Obter recursão

3. Algoritmo Bottom-UP para valor ótimo

Exemplo Problema

Programação Dinâmica

Obter recursão
 Algoritmo Bottom-LIP

 Obter uma solução ótima do problema

Prof. Eurinardo

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problem

Programação Dinâmic

Obter recursão

para valor otimo
4. Obter uma solução ótima do problema

4. Obter uma solução ótima do problema

Algoritmo 22: Imprimir(S, X, i, j)

Programação Dinâmica

2. Obter recursão

 Obter uma solução ótima do problema

4. Obter uma solução ótima do problema

Algoritmo 23: Imprimir(S, X, i, j)

1 se $i \neq 0$ e $j \neq 0$ então

Exemplo Problema

Programação Dinâmica

Obter recursão

 Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

4. Obter uma solução ótima do problema

Algoritmo 24: Imprimir(S, X, i, j)

- se $i \neq 0$ e $j \neq 0$ então
- 2 | caso S[i,j] seja

Programação Dinâmic

2. Obter recursão

Algoritmo Bottom-UP para valor ótimo

 Obter uma solução ótima do problema

```
Algoritmo 25: Imprimir(S, X, i, j)
```

Programação Dinâmica

2. Obter recursão

para valor ótimo
4. Obter uma solução ótima do problema

4. Obter uma solução ótima do problema

```
Algoritmo 26: Imprimir(S, X, i, j)
```

```
1 se i \neq 0 e j \neq 0 então
2 | caso S[i,j] seja
```

3

"\forall ": Imprimir(S, X, i - 1, j);

```
Exemplo
Problema
```

- Programação Dinâmica
- 2. Obter recursão
- para valor ótimo
- Obter uma solução ótima do problema

```
Algoritmo 27: Imprimir(S, X, i, j)
```

```
se i \neq 0 e j \neq 0 então
caso S[i,j] seja

\text{"$\uparrow$} : \text{Imprimir}(S, X, i-1, j);

4
```

```
Exemplo
Problema
```

- Programação Dinâmio
- Obter recursão
 - Algoritmo Bottom-UP
 para valor ótimo
- Obter uma solução ótima do problema

```
4. Obter uma solução ótima do problema
```

```
Algoritmo 28: Imprimir(S, X, i, j)
```

```
1 se i \neq 0 e j \neq 0 então

2 caso S[i,j] seja

3 "↑": Imprimir(S, X, i-1, j);

4 "←": Imprimir(S, X, i, j-1);
```

```
Exemplo
Problema
```

- Programação Dinâmio
- Obter recursão
- Algoritmo Bottom-UP para valor ótimo
- Obter uma solução ótima do problema

```
4. Obter uma solução ótima do problema
```

```
Algoritmo 29: Imprimir(S, X, i, j)

se i \neq 0 e j \neq 0 então

caso S[i, j] seja

"\tau": Imprimir(S, X, i = 1, j);

"\tau": Imprimir(S, X, i, j = 1);

"\tau": ":
```

```
Exemplo
Problema
```

- Programação Dinâmica
- Obter recursão
- Algoritmo Bottom-UP
 para valor ótimo
- Obter uma solução ótima do problema

```
Algoritmo 30: Imprimir(S, X, i, j)

se i \neq 0 e j \neq 0 então

caso S[i,j] seja

"\neq": Imprimir(S, X, i-1, j);

"\neq": Imprimir(S, X, i, j-1);

[Imprimir(S, X, i, j-1);

escreva "X[i]"
```

Aulas Passadas

Subsequência Comum mais Longa

Problema

Programação Dinâmica

Obter recursão

3. Algoritmo Bottom-UP

 Obter uma solução ótima do problema

LEISERSON, C.E., STEIN, C., RIVEST, R.L., CORMEN T.H.

Algoritmos: teoria e prática, 3ed. Editora Campus, ano 2012.

Aulas Passadas

Subsequência Comum mais Longa

Exemplo Problema

Programação Dinâmica

- Propriedade de
- Obter recursão
- Algoritmo Bottom-UP para valor ótimo
- Obter uma solução ótima do problema

Obrigado!