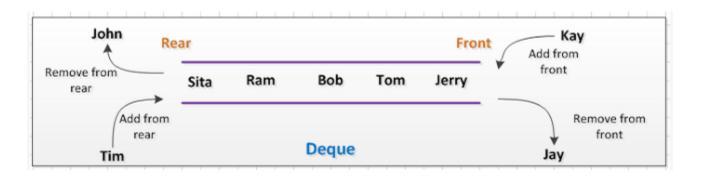
<u>DEQue</u>

DEQue

A Fila Circular com Dupla Terminação ou *DEQue* (*Double* - *Ended Queue*) é uma estrutura de dados em lista, à qual pode-se ter

acesso pelas duas extremidades:





Prof. Calvetti

2/130

DEQue

Existem quatro funções principais que se aplicam em DEQue:

- pushFront: Insere um item à frente;
- pushBack: Insere um item atrás;
- popFront: Retira um item da frente; e
- popBack: Retira um item de trás.

Prof. Calvetti 3/130

DEQue

Entrada

<u>DEQue</u>

Saída

E D C B A

Início > < Fim

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

F E D C B pushBack

Início > A < Fim

Prof. Calvetti

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

F E D C B

< Fim

A

Início >

Autor. Prof Robson Calvetti - Todos os direitos reservados (6

Prof. Calvetti

6/130

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

G F E D C pushBack

B < Fim

A

Início >

Autor. Prof Robson Calvetti - Todos os direitos reservados (3)

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

GFEDC

< Fim

В

Α

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F E D pushFront

< Fim

В

Α

Início > C

utor: Prof. Robson Calvetti - Todos os direitos reservados (

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

HGFED

< Fim

B

A

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F E pushBack

D < Fim

В

1

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados (

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

HGFE

< Fim

)

 \mathbf{B}

4

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F pushFront

< Fim

В

4

 \mathbb{C}^{-}

Início > E

Autor: Prof. Robson Calvetti - Todos os direitos reservados

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F < Fim

D

B

4

 \Box

E

Início >

DEQue

Entrada

H G F

<u>DEQue</u>

< Fim popBack

<u>Saída</u>

D

B

4

E

Início >

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F

< Fim

B

4

 \mathbb{C}

E

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados

DEQue

Entrada <u>DEQue</u>

H G F E D

< Fim

B

4

 \mathbb{C}

Início > <u>popFront</u>

ttor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Saída

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G F E D

< Fim

B

A

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados O.

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

H G F

BED

< Fim popBack

A

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

H G F

BED

< Fim

A

 \mathbf{C}

Início >

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H G pushBack B E D

F < Fim

4

 \mathbf{C}

Início >

utor: Prof. Robson Calvetti - Todos os direitos reservados ©

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

HG BED

< Fim

F

Α

 \mathbf{C}

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados @

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H pushBack B E D

G < Fim

F

4

 \mathbf{C}

Início >

Autor Prof Robson Calvetti - Todos os direitos reservados

DEQue

<u>Entrada</u> <u>DEQue</u> <u>Saída</u>

H B E D

< Fim

G

F

A

 \mathbf{C}

Início >

Autor. Prof Robson Calvetti - Todos os direitos reservados (

Prof. Calvetti

24/130

< Fim

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

<u>pushBack</u>

BED

G

Н

F

1

 \mathbf{C}

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

< Fim

BED

H

G

F

4

 \mathbf{C}

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

< Fim popBack H B E D

G

F

A

 \mathbf{C}

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

HBED

< Fim

G

F

 \mathbf{A}

 \mathbf{C}

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

GHBED

< Fim popBack

F

4

 C

Início >

Autor: Prof Robson Calvetti - Todos os direitos reservados

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

GHBED

< Fim

F

4

 \mathbf{C}

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

CGHBED

< Fim

F

4

Início >

popFront

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

CGHBED

< Fim

F

A

Início >

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti

32/130

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

ACGHBED

< Fim

F

Início >

popFront

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

ACGHBED

< Fim

F

Início >

DEQue

Entrada

<u>DEQue</u>

<u>Saída</u>

FACGHBED

< Fim

Início >

popFront

Prof. Calvetti

DEQue

Entrada

<u>DEQue</u>

Saída

FACGHBED

Início >

< Fim

DEQue

Entrada

<u>DEQue</u>

Saída

HGFEDCBA

FACGHBED

alteração da ordem

para qualquer quantidade

DEQue

Exemplos de utilização de DEQue:

- Execução de processos prioritários em Sistemas Distribuídos;
- Alteração da ordem de informações armazenadas; e
- Retirada de um elemento específico de uma estrutura de

dados.

Prof. Calvetti 38/130

DEQue

Outras funções que se aplicam a *DEQue*:

- *size* : Informa o tamanho;

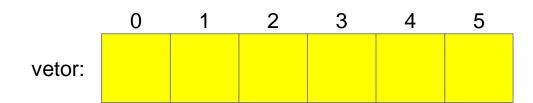
- front: Informa o elemento do início, sem retirá-lo;

- back: Informa o elemento do fim, sem retirá-lo;

- isEmpty: Informa se a DEQue está vazia; e

- isFull: Informa se a DEQue está cheia.

Prof. Calvetti 39/130



int vetor[] = new vetor[6];

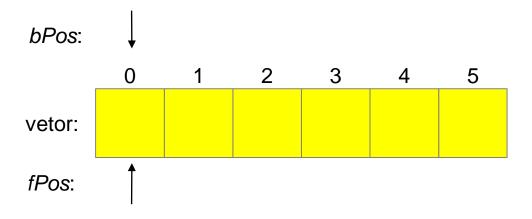
Vetor com o tamanho máximo necessário

ttor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti

40/130

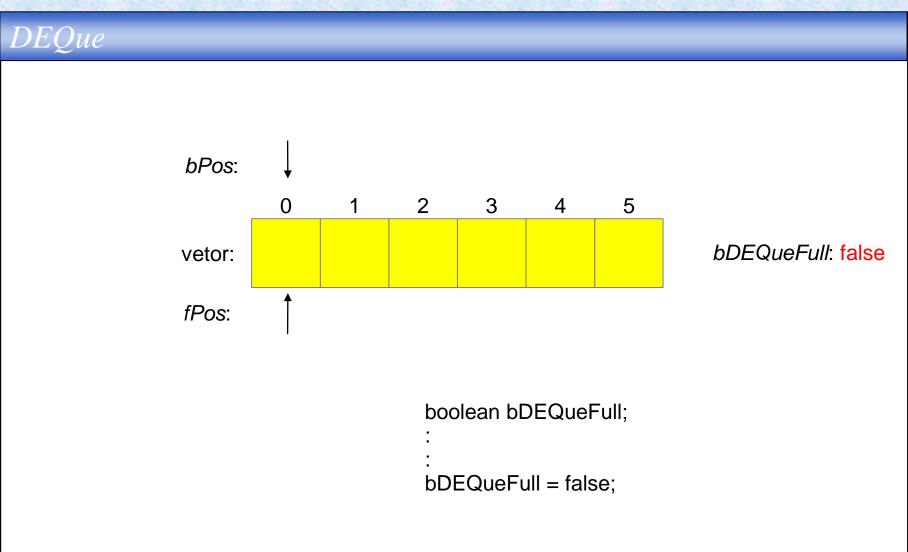
DEQue



```
int bPos, fPos;
:
bPos = fPos = 0;
```

Inicializa indicadores de posição (front e back)

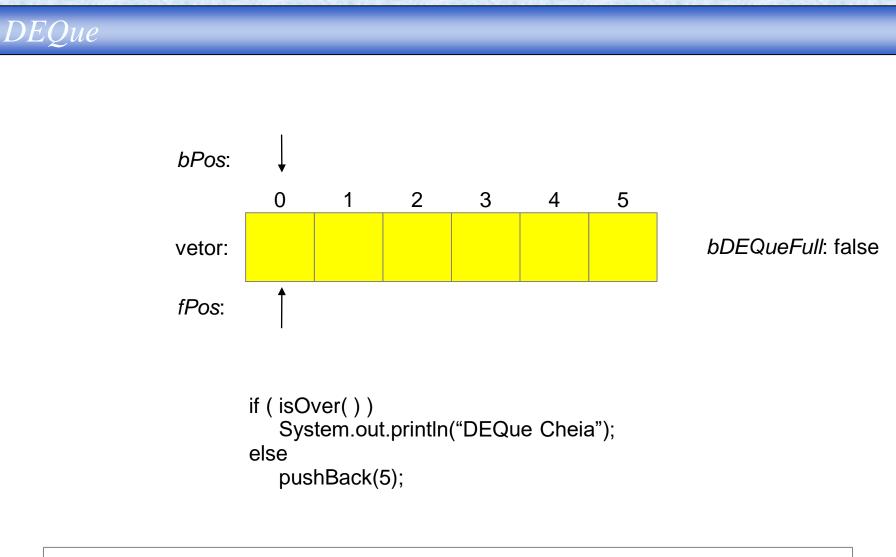
Prof. Calvetti 41/130



Inicializa indicador de *DEQue* cheia

Autor: Prof. Robson Calvetti - Todos os direit

Prof. Calvetti 42/130



Verifica se está cheia

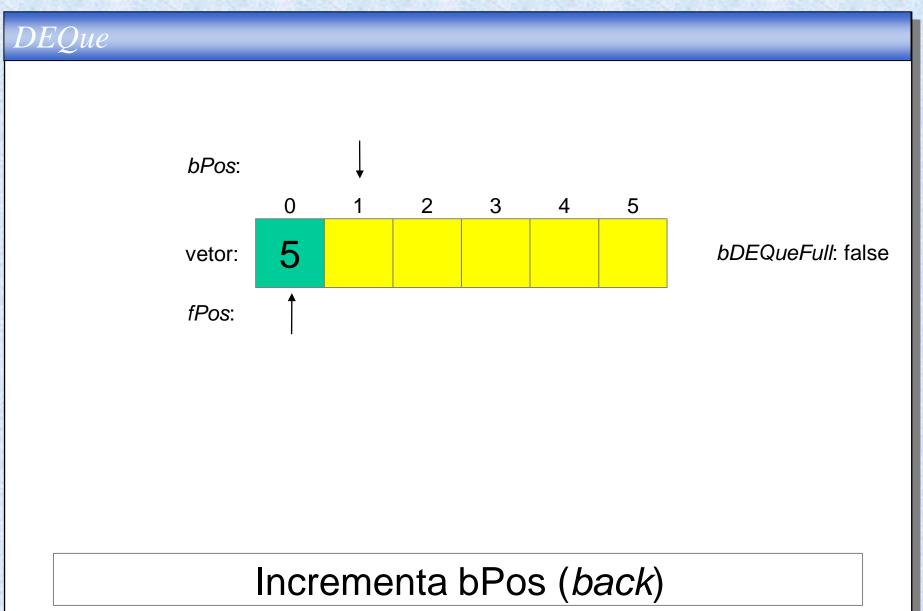
Prof. Calvetti 43/130

DEQue

bPos: 3 5 bDEQueFull: false vetor: fPos: if (isOver()) System.out.println("DEQue Cheia"); else pushBack(5);

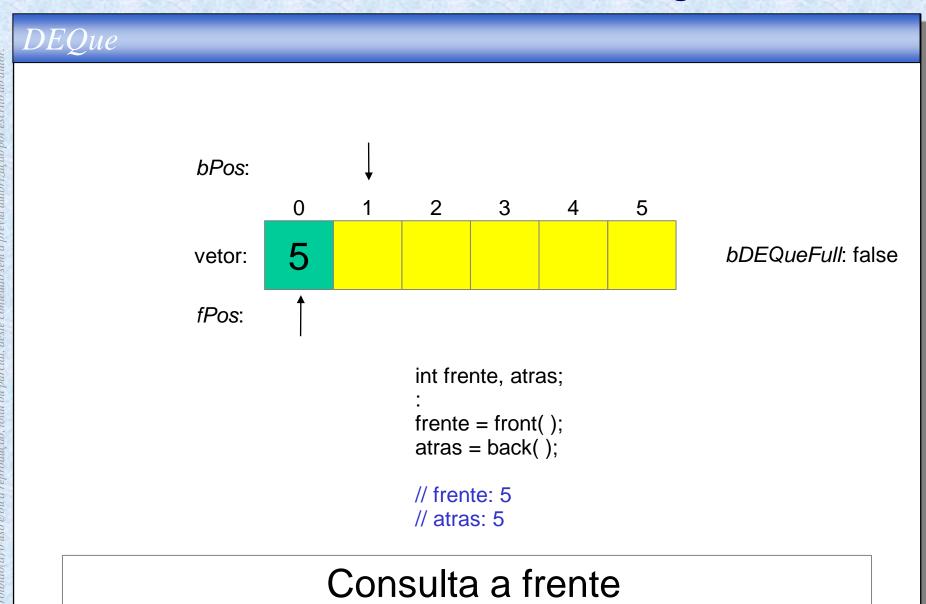
Não está cheia! Coloca o elemento 5

Prof. Calvetti 44/130



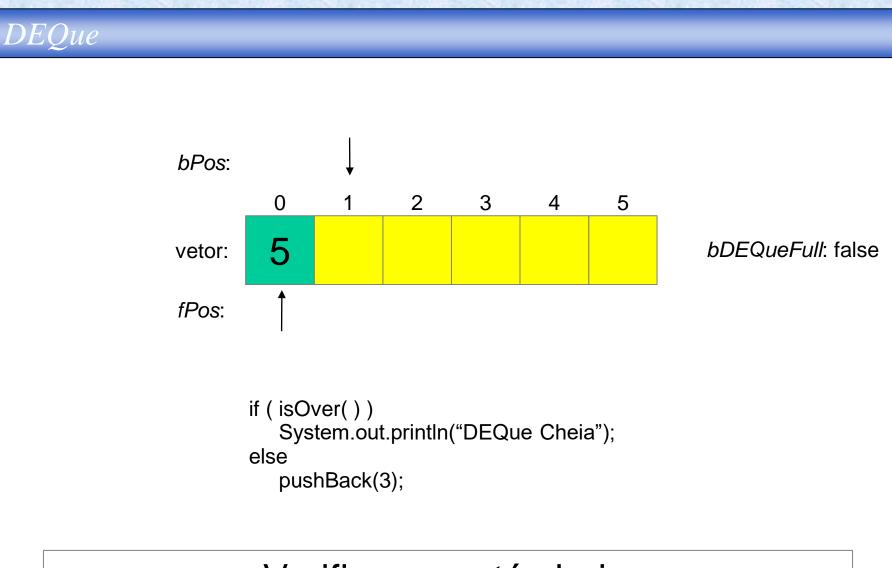
Prof. Calvetti

45/130



Prof. Calvetti

46/130



Verifica se está cheia

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 47/130

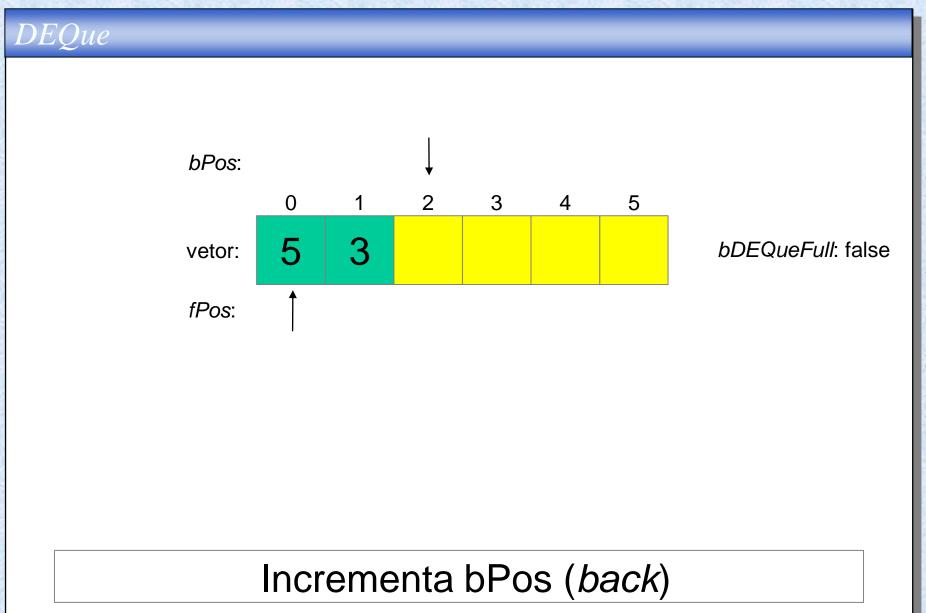
DEQue

bDEQueFull: false

```
if ( isOver( ) )
    System.out.println("DEQue Cheia");
else
    pushBack(3);
```

Não está cheia! Coloca o elemento 3

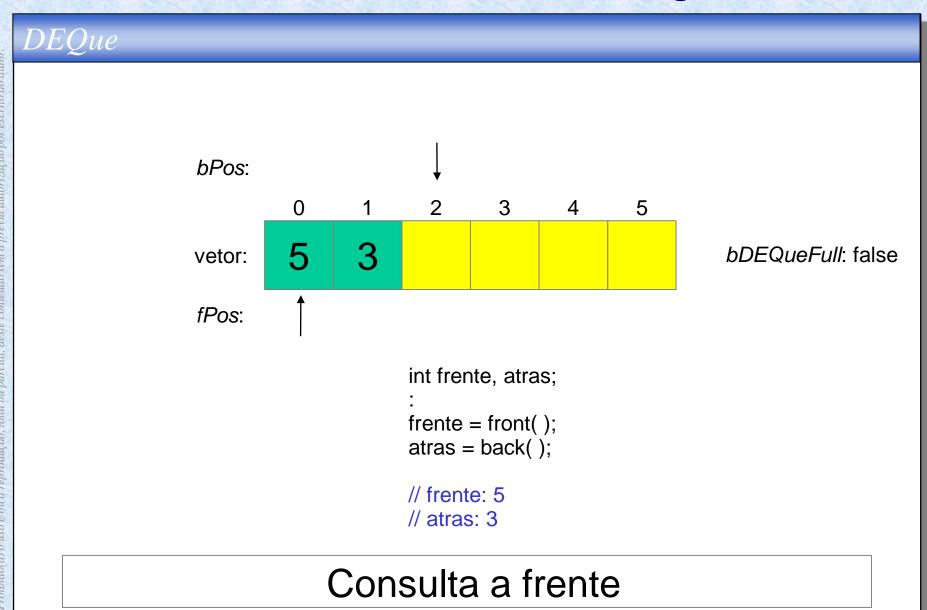
Prof. Calvetti 48/130



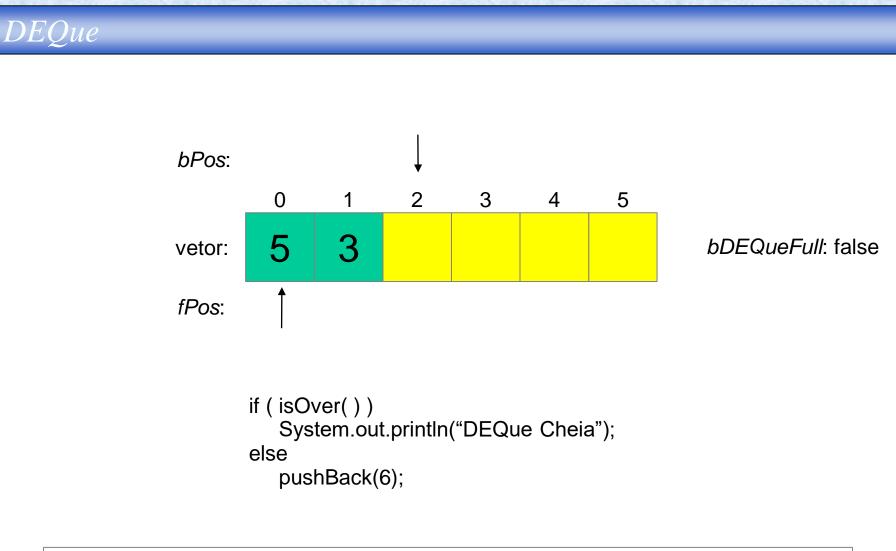
Prof. Calvetti 49/130

Ow Dust Dobern Calmotti Todos os divoitos voscomodos O

Estruturas de Dados e Análise de Algoritmos



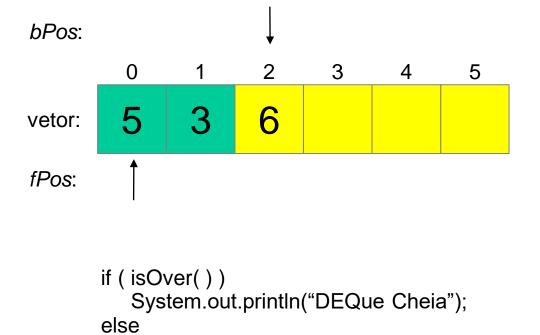
Prof. Calvetti 50/130



Verifica se está cheia

Prof. Calvetti 51/130

DEQue

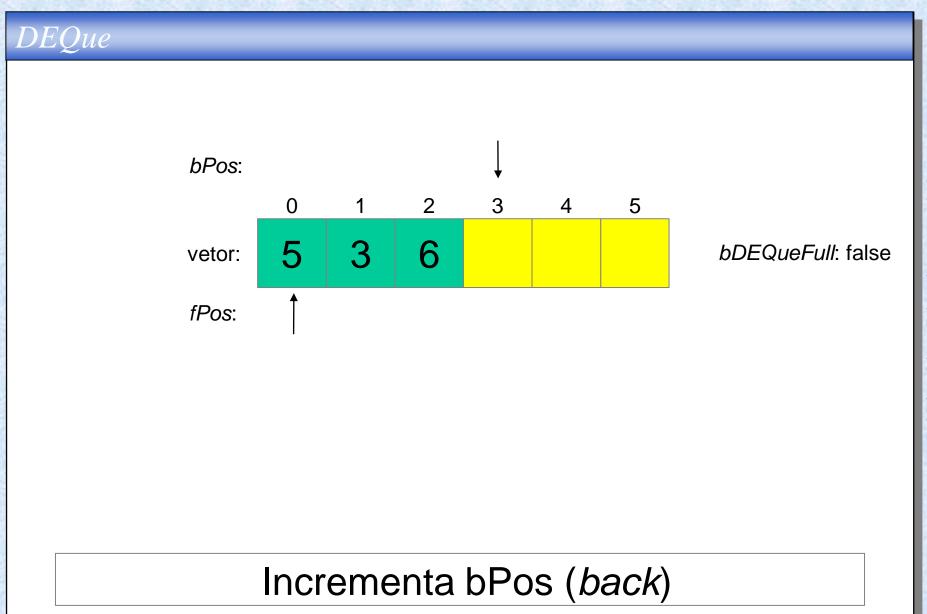


pushBack(6);

Não está cheia! Coloca o elemento 6

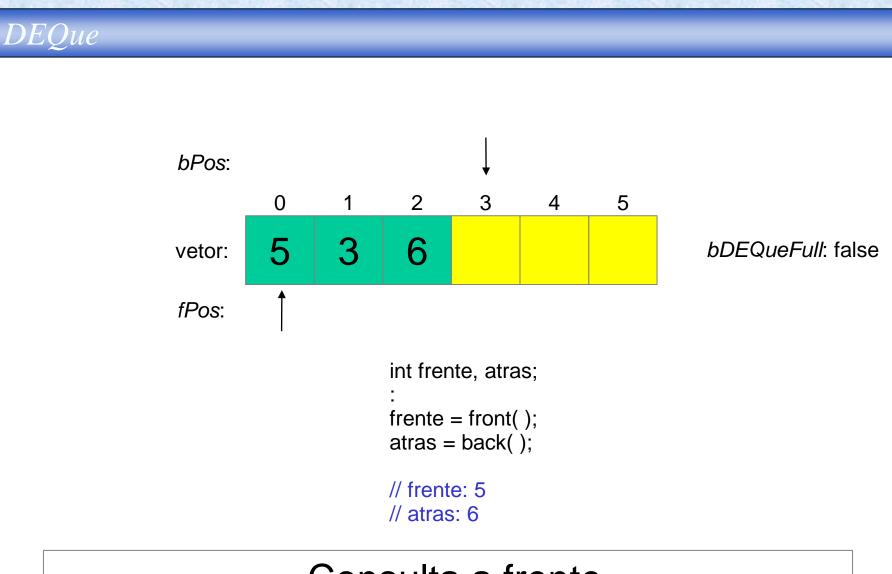
bDEQueFull: false

Prof. Calvetti 52/130



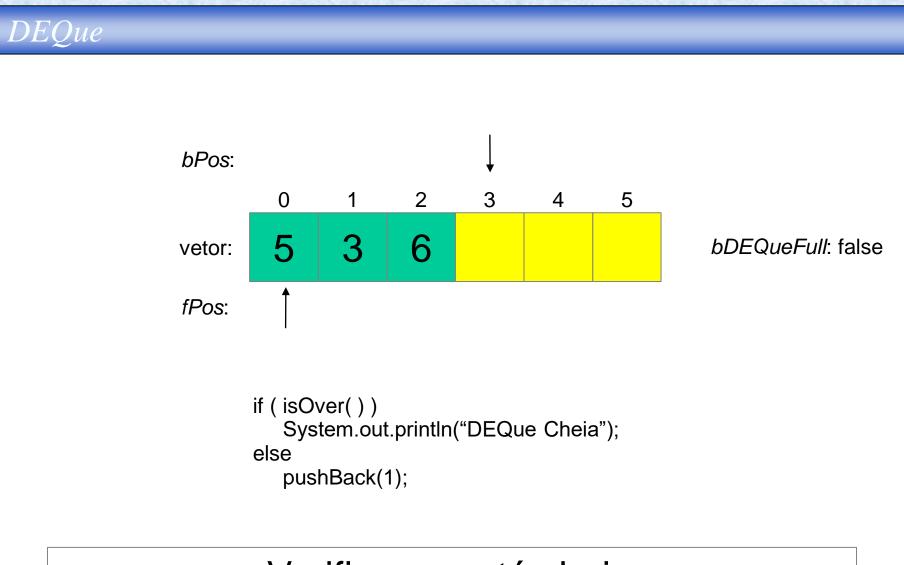
Prof. Calvetti

53/130



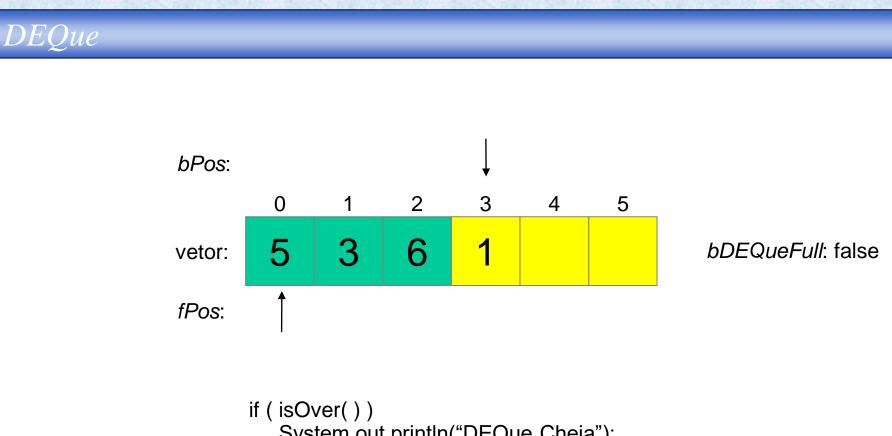
Consulta a frente

Prof. Calvetti 54/130



Verifica se está cheia

Prof. Calvetti 55/130

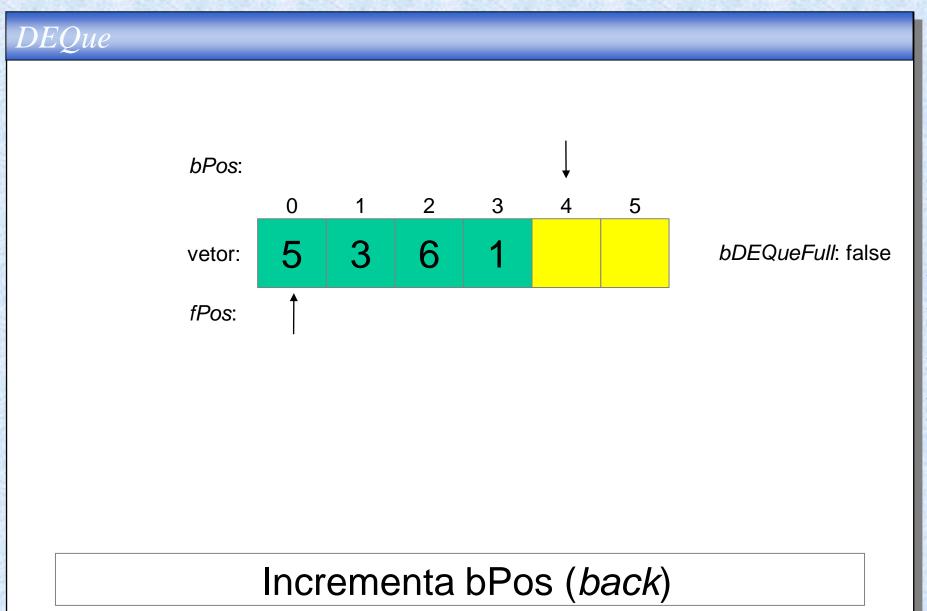


```
if ( isOver( ) )
    System.out.println("DEQue Cheia");
else
    pushBack(1);
```

Não está cheia! Coloca o elemento 1

: Prof. Robson Calvetti - Todos os direitos reservados ©.

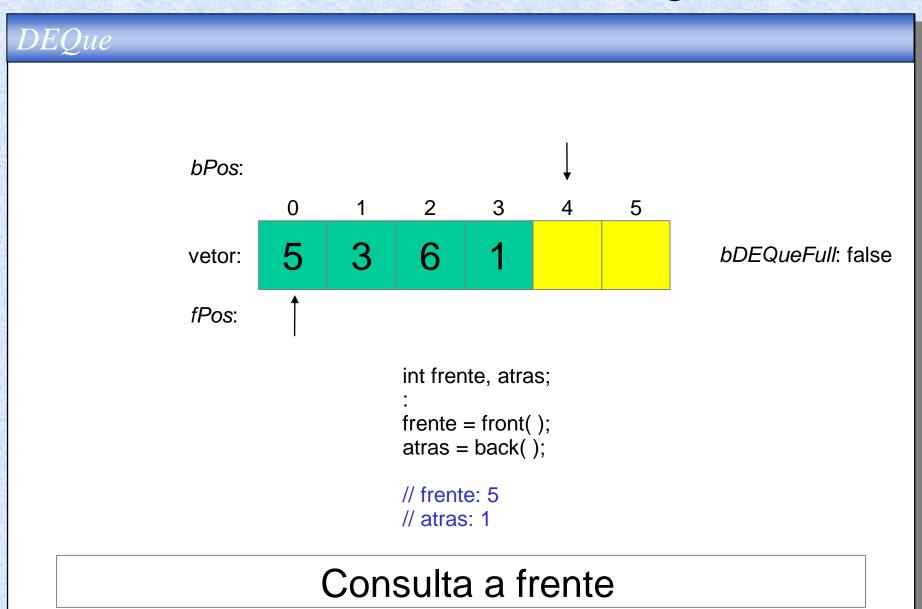
Prof. Calvetti 56/130



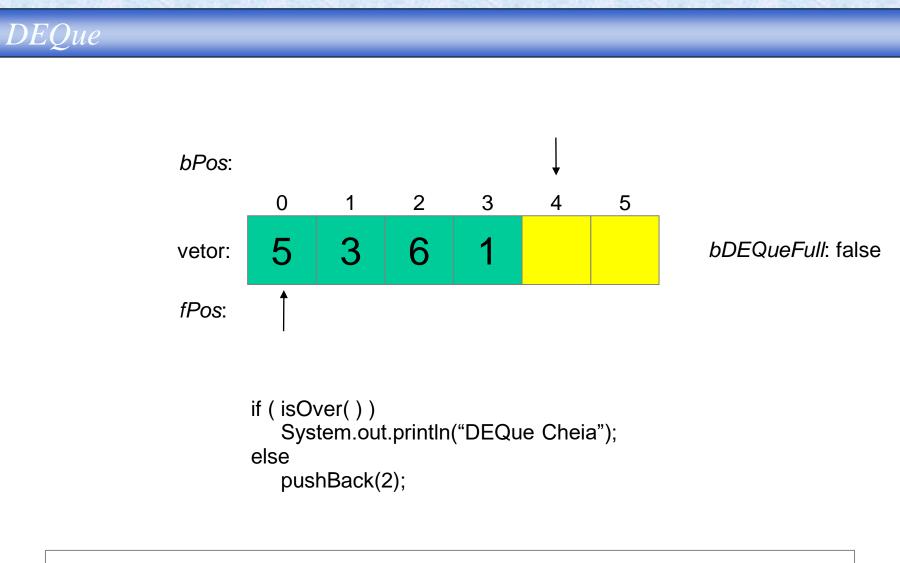
Prof. Calvetti 57/130

Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos



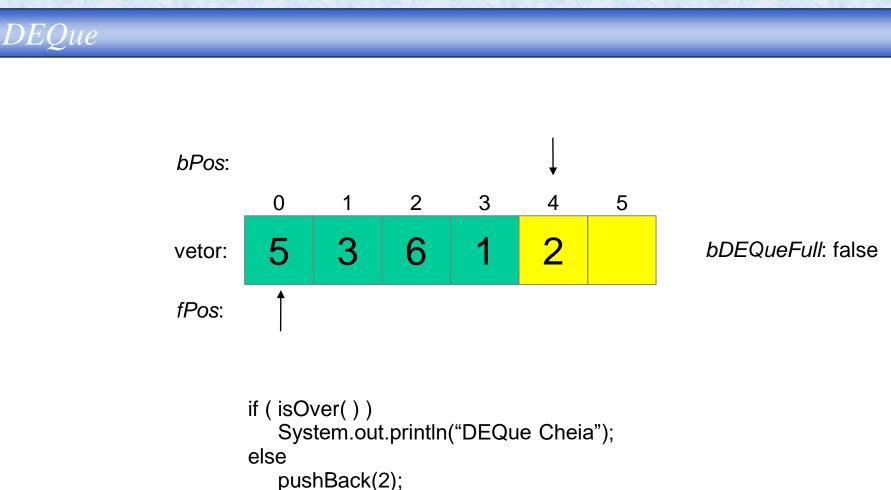
Prof. Calvetti 58/130



Verifica se está cheia

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 59/130

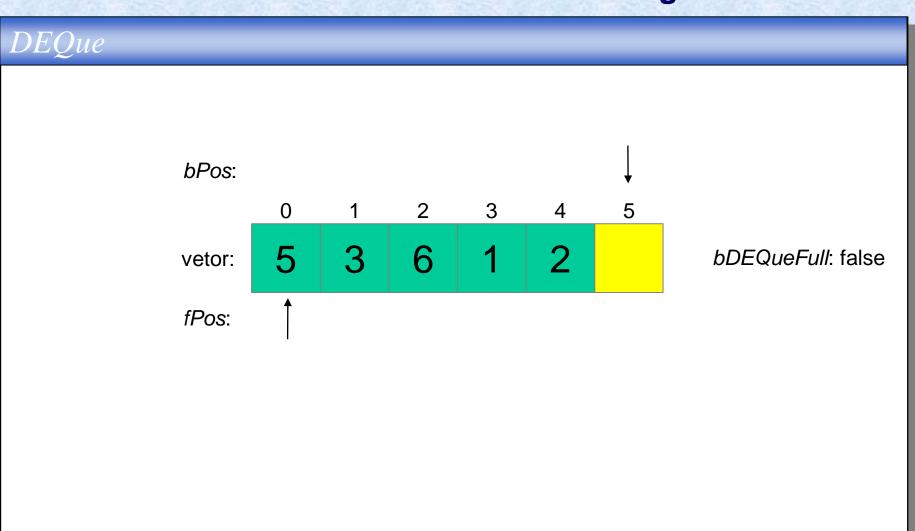


Não está cheia! Coloca o elemento 2

tor: Prof. Robson Calvetti - Todos os direitos reservados ©.

60/130

Prof. Calvetti

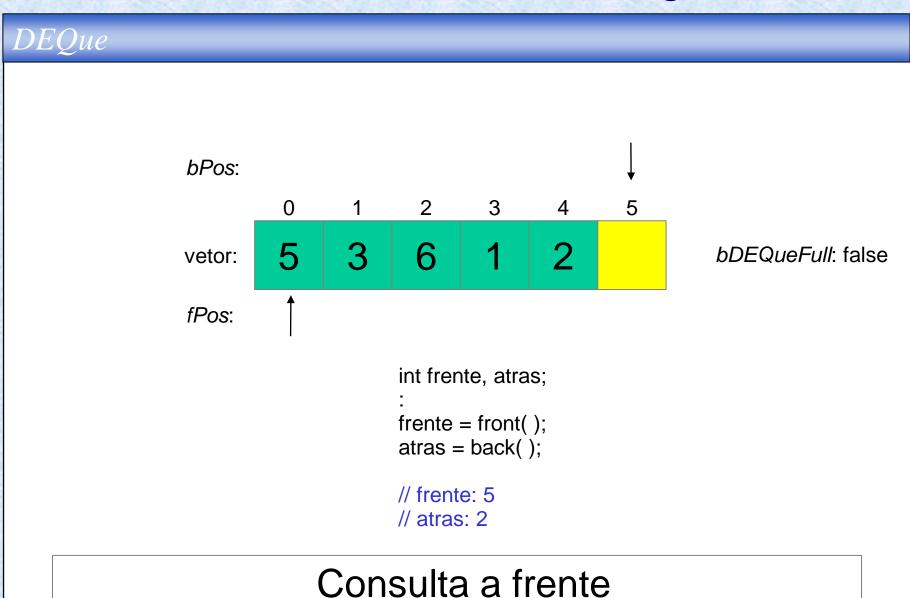


Incrementa bPos (back)

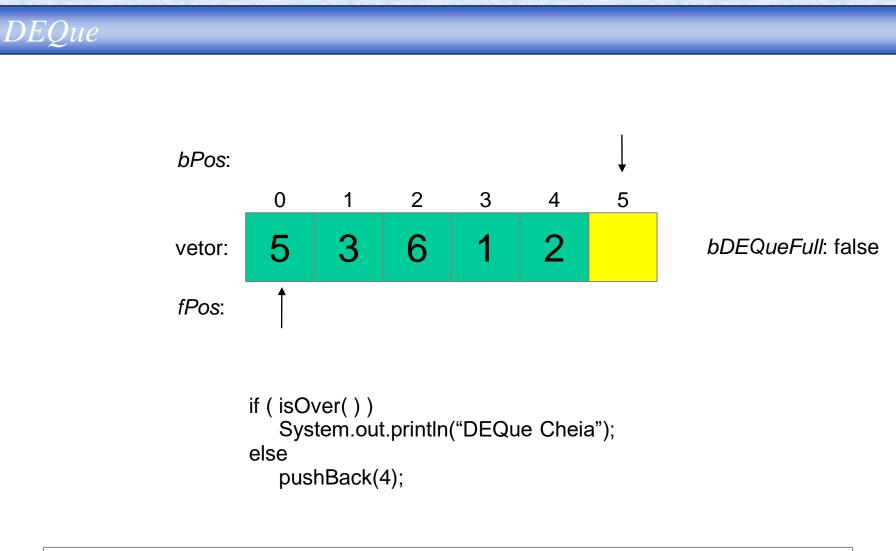
Prof. Calvetti

61/130

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.



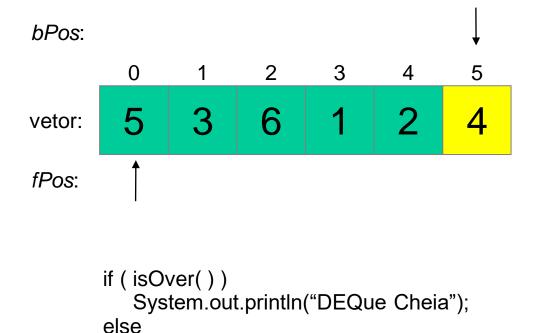
Prof. Calvetti 62/130



Verifica se está cheia Prof. Calvetti 63/130

Latituturas de Dados e Arianse de Algoritmos

DEQue

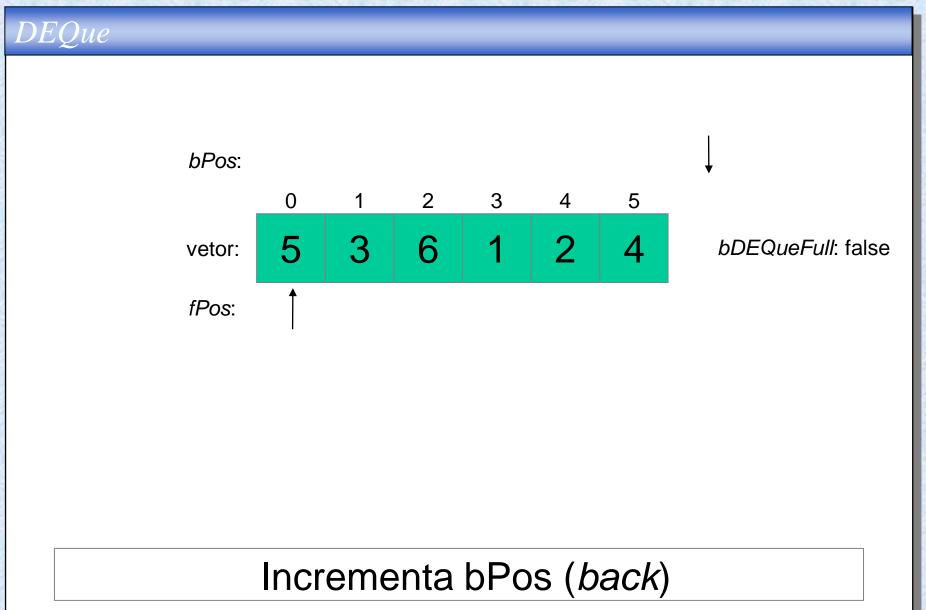


pushBack(4);

bDEQueFull: false

Não está cheia! Coloca o elemento 4

Prof. Calvetti 64/130



Prof. Calvetti 65/130

Todos os direitos reservados (O).

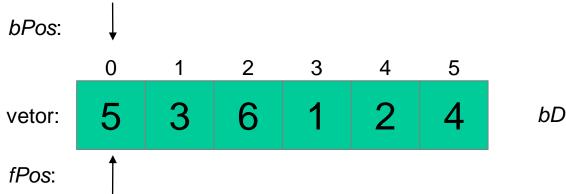
Estruturas de Dados e Análise de Algoritmos

DEQue bPos: 5 3 bDEQueFull: false vetor: fPos: if (bPos >= vetor.length) bPos = 0; else bPos++;

Ajusta bPos (back)

Prof. Calvetti 66/130





bDEQueFull: true

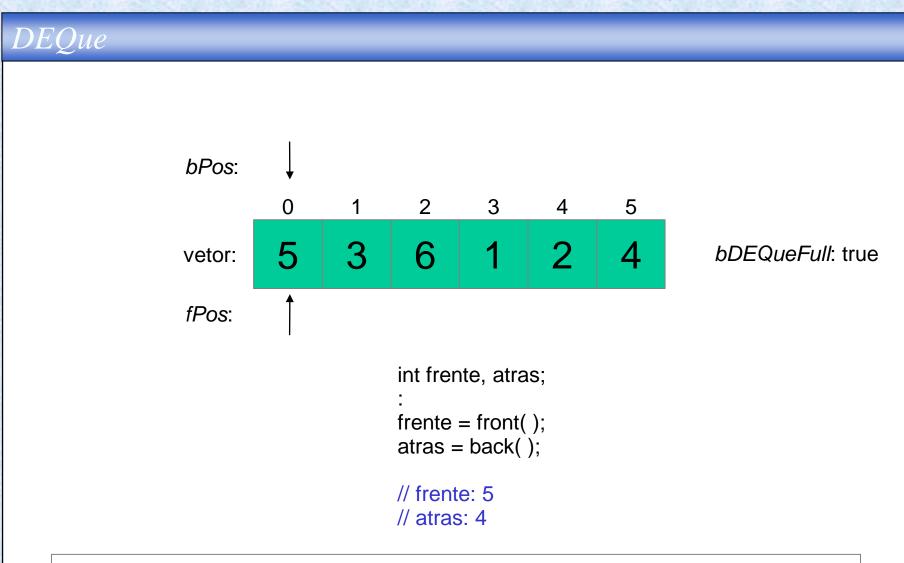
Indica que a DEQue está cheia

eprodução, total ou parcial, deste conteúdo sem a prévia autorização p

Prof. Calvetti 67/130

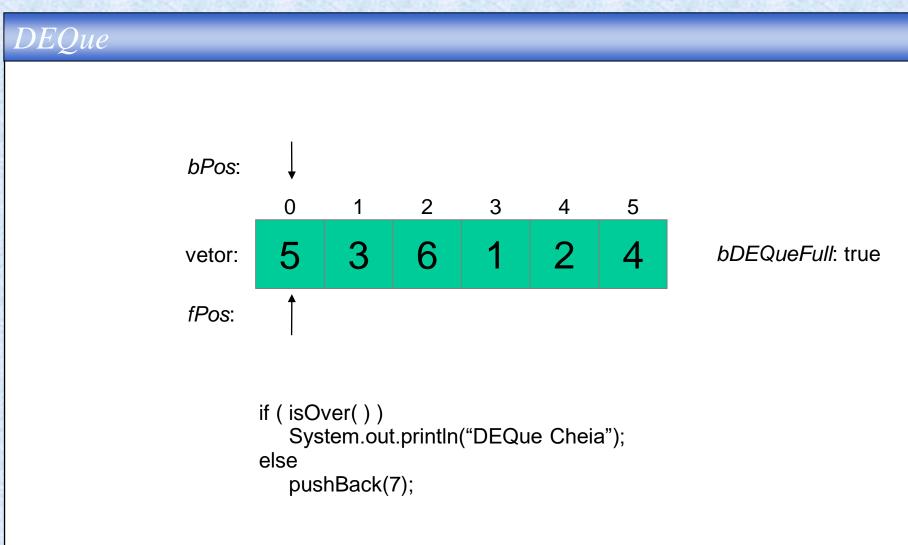
obson Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos



Consulta a frente

Prof. Calvetti 68/130



Verifica se está cheia

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.

69/130

Prof. Calvetti

DEQue bPos: 3 bDEQueFull: true vetor: fPos: if (isOver()) System.out.println("DEQue Cheia");

```
else
   pushBack(7);
```

Está cheia! Não coloca o elemento 7

Prof. Calvetti 70/130

Robson Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos

DEQue bPos: 5 3 2 bDEQueFull: true vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popFront();

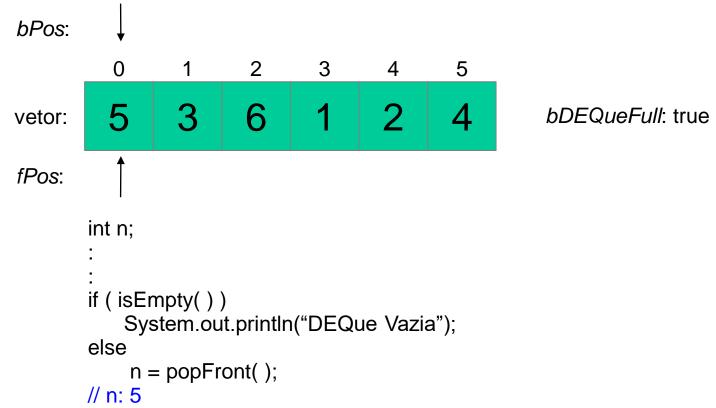
Verifica se está vazia

Prof. Calvetti 71/130

r: Prof. Robson Calvetti - Todos os direitos reservados ©.

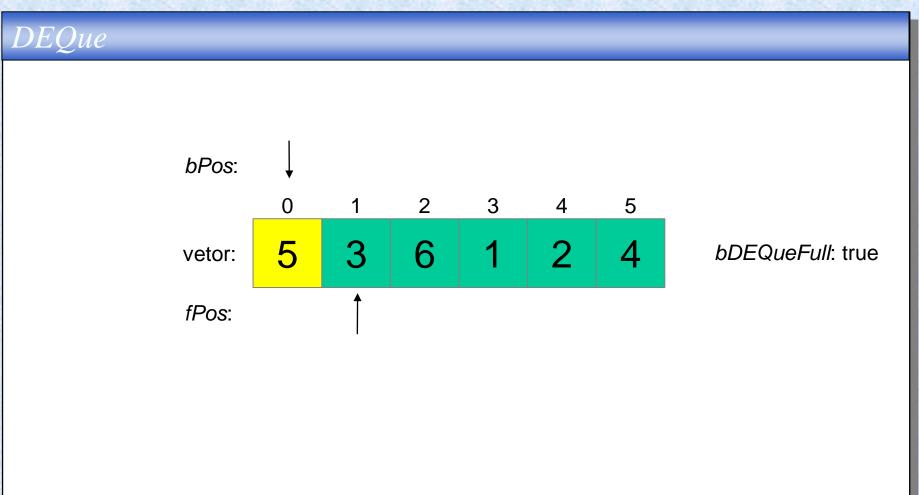
Estruturas de Dados e Análise de Algoritmos

DEQue

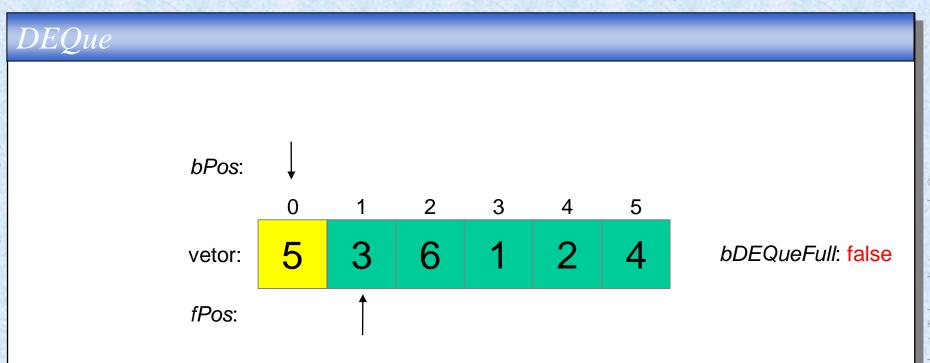


Não está vazia! Retira o elemento da frente

Prof. Calvetti 72/130

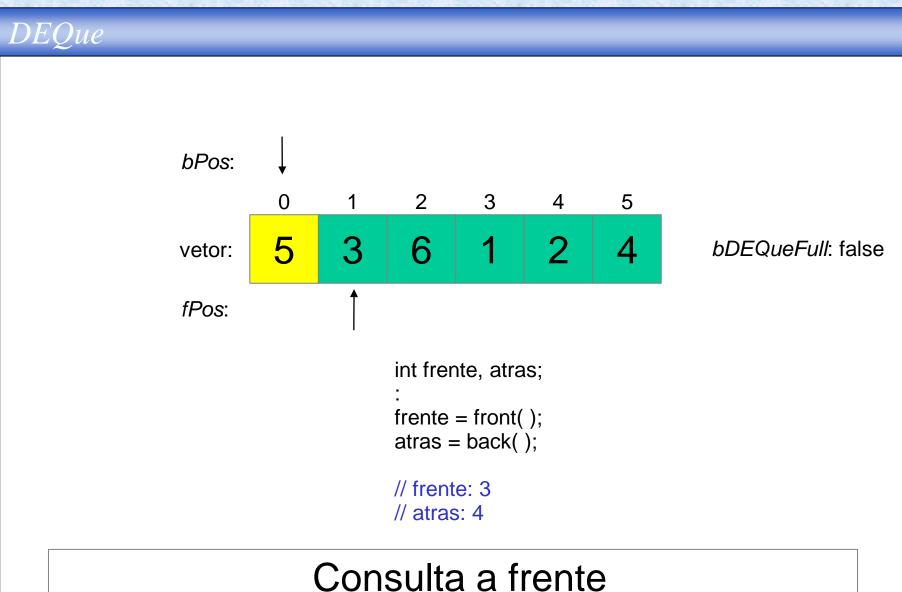


Incrementa fPos (front) Prof. Calvetti 73/130

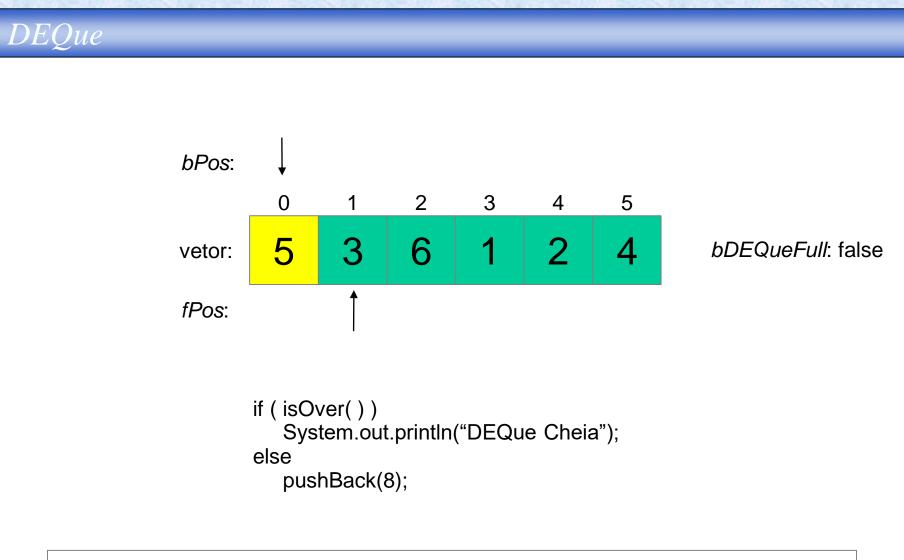


Indica que a DEQue não está cheia

Prof. Calvetti 74/130



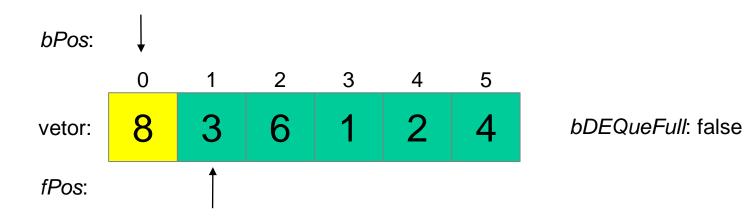
Prof. Calvetti 75/130



Verifica se está cheia

Prof. Calvetti 76/130

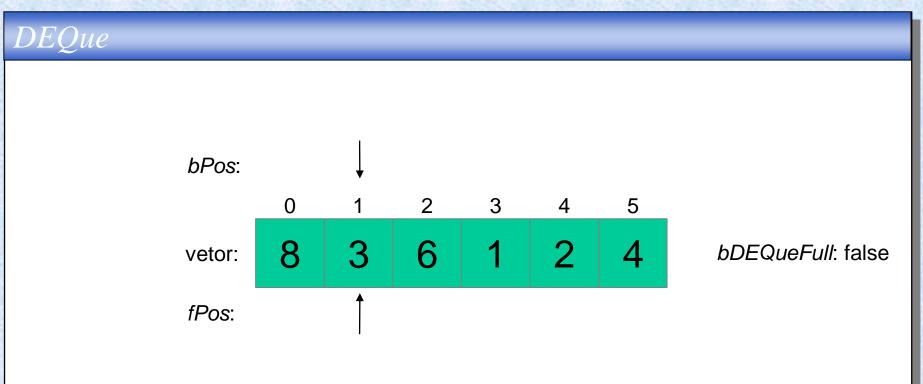
DEQue



```
if (isOver())
   System.out.println("DEQue Cheia");
else
   pushBack(8);
```

Não está cheia! Coloca o elemento 8

Prof. Calvetti 77/130

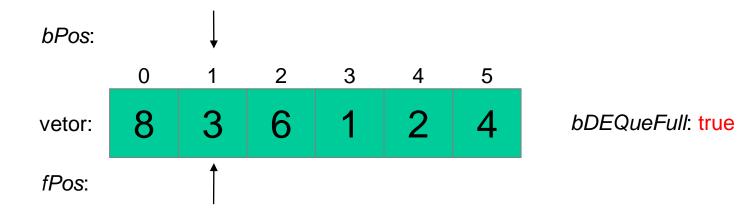


Incrementa bPos (back)

Prof. Calvetti 78/130

.: Prof. Robson Calvetti - Todos os direitos reservados ©.





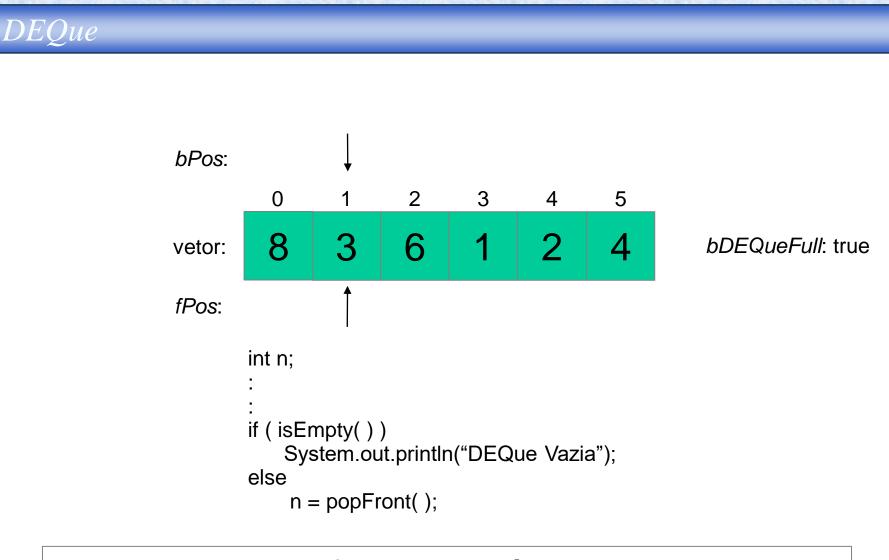
Indica que a DEQue está cheia

Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 79/130

or: Prof. Robson Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos



Verifica se está vazia

Prof. Calvetti 80/130

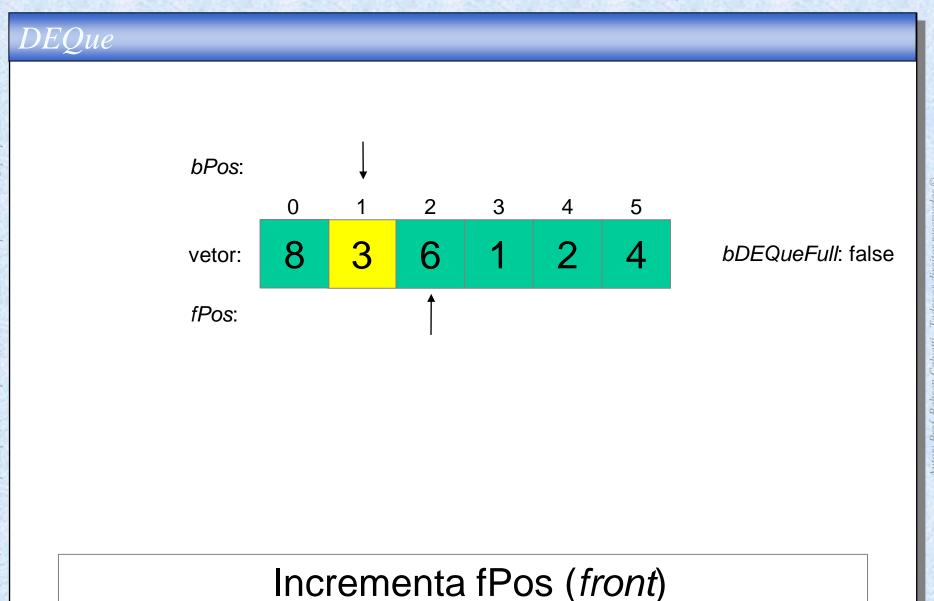
: Prof. Robson Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos

DEQue bPos: 3 3 bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popFront(); // n: 3

Não está vazia! Retira o elemento da frente

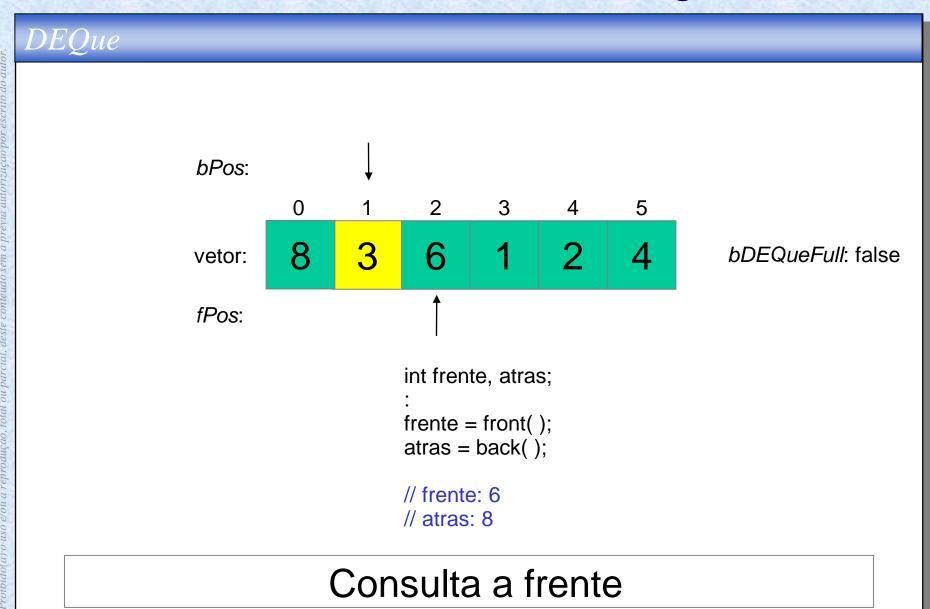
Prof. Calvetti 81/130



Prof. Calvetti 82/130

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos



Prof. Calvetti 83/130

DEQue bPos: 3 bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popFront();

Verifica se está vazia

Autor: Prof. Robson Calvetti - Todos os direitos reservad

Prof. Calvetti 84/130

Robson Calvetti - Todos os direitos reservados ©.

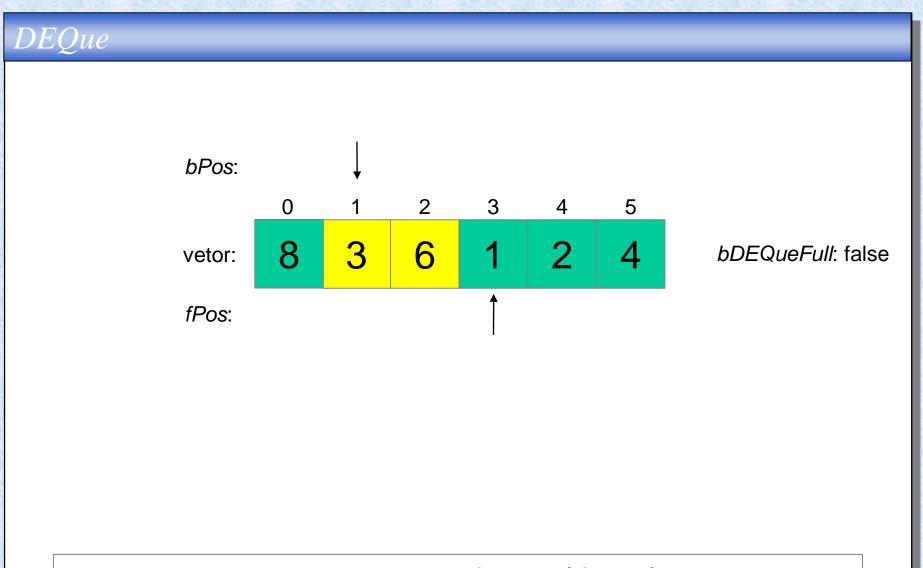
Estruturas de Dados e Análise de Algoritmos

DEQue bPos: 3 bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popFront();

Não está vazia! Retira o elemento da frente

// n: 6

Prof. Calvetti 85/130

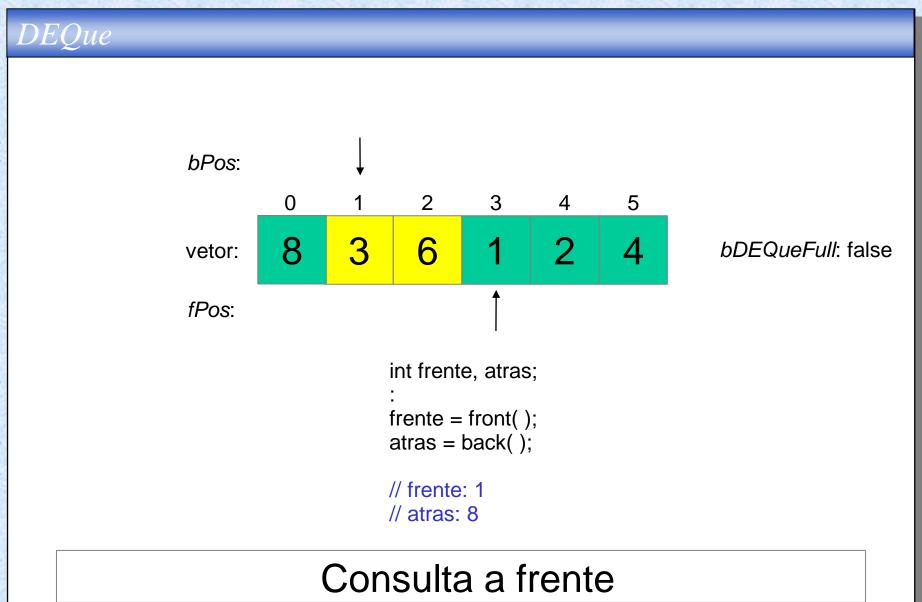


Incrementa fPos (front)

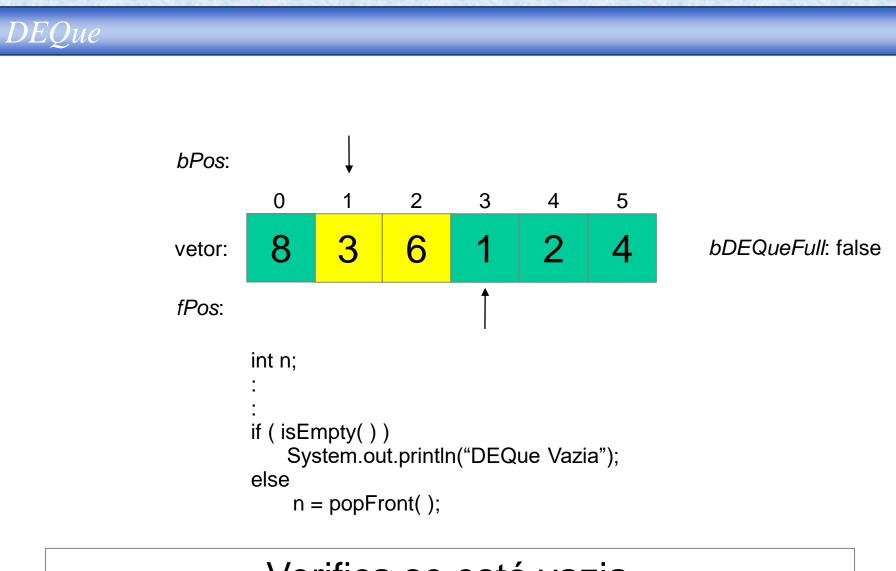
Prof. Calvetti

86/130

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.



Prof. Calvetti 87/130



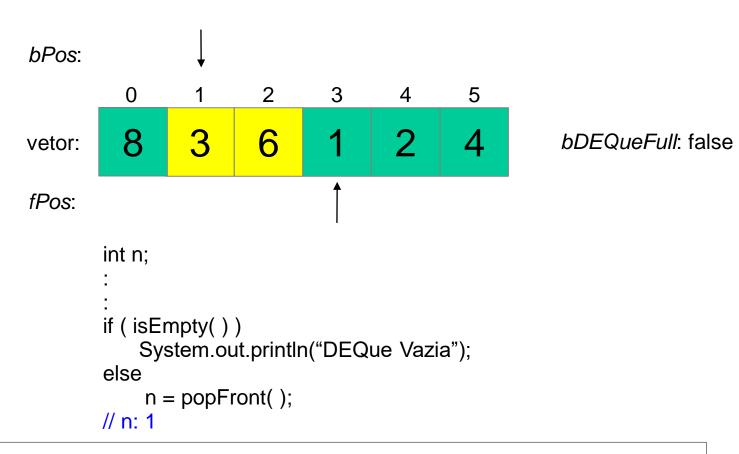
Verifica se está vazia

Prof. Calvetti 88/130

os direitos reservados ©.

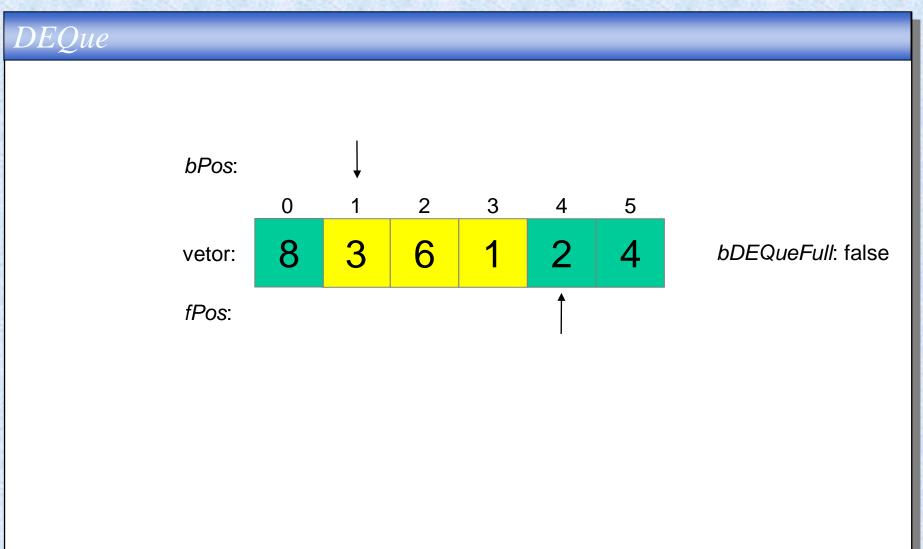
Estruturas de Dados e Análise de Algoritmos

DEQue



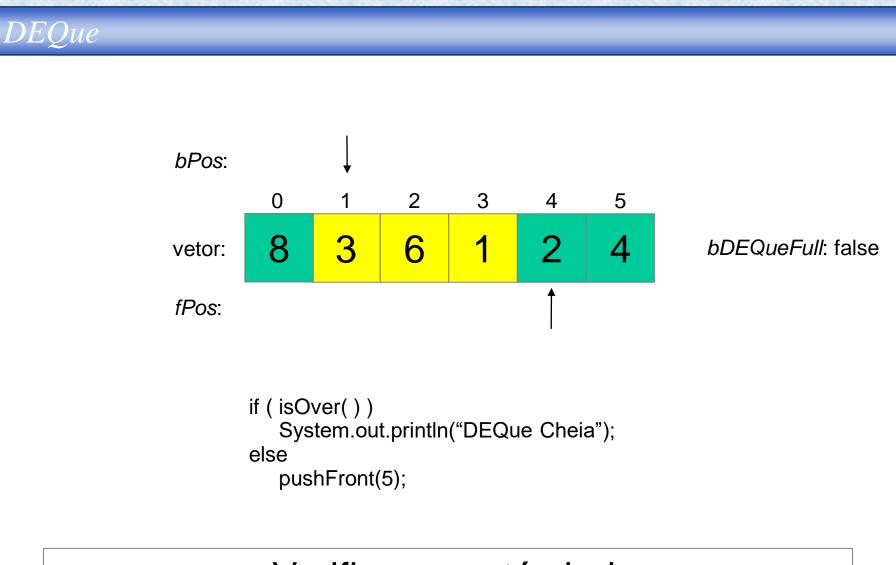
Não está vazia! Retira o elemento da frente

Prof. Calvetti 89/130



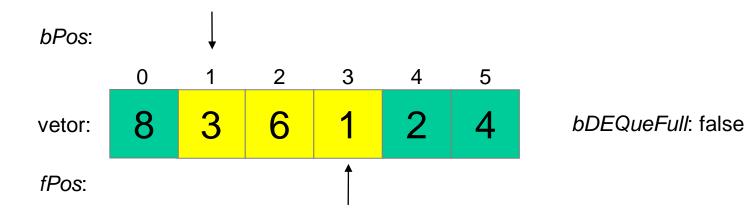
Incrementa fPos (front)

Prof. Calvetti
90/130



Verifica se está cheia

DEQue

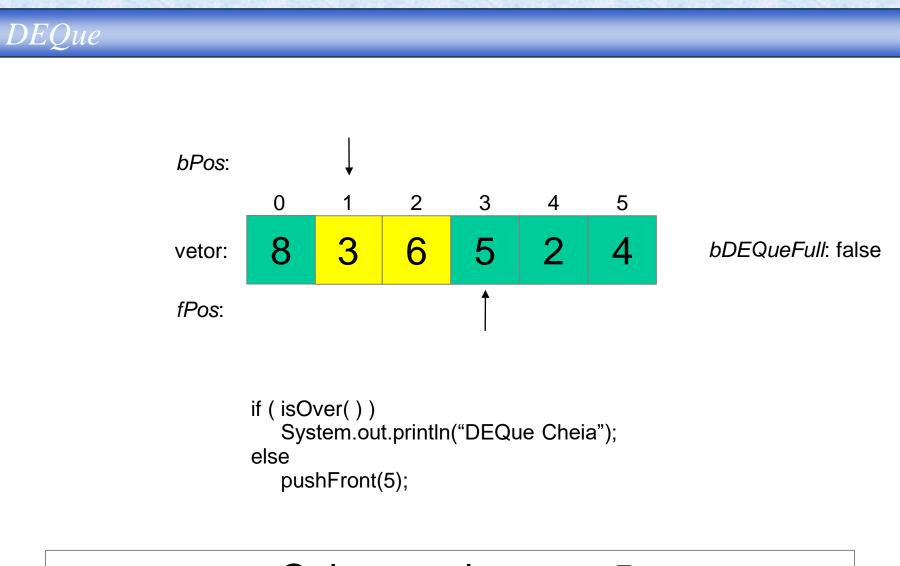


```
if ( isOver( ) )
    System.out.println("DEQue Cheia");
else
    pushFront(5);
```

Não está cheia! Decrementa fPos (front)

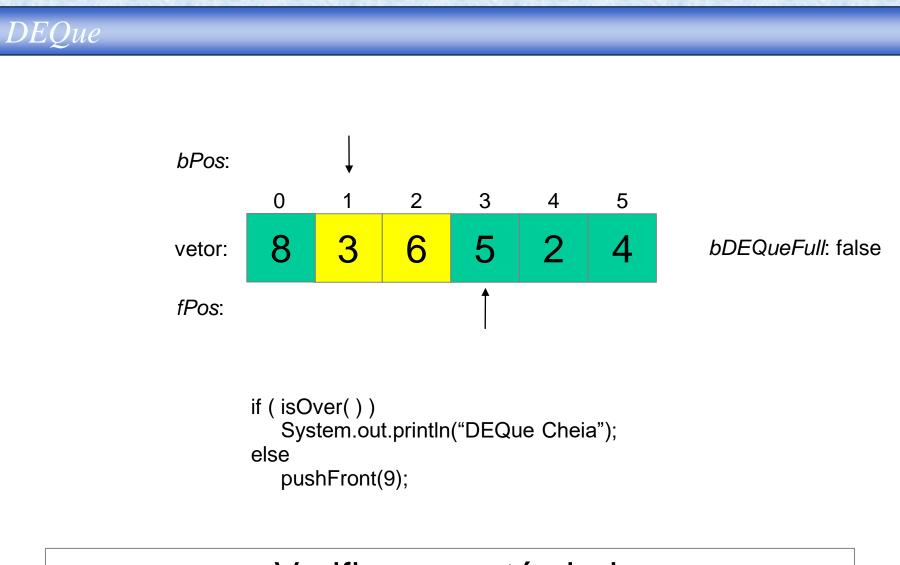
or: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 92/130



Coloca o elemento 5

Prof. Calvetti 93/130



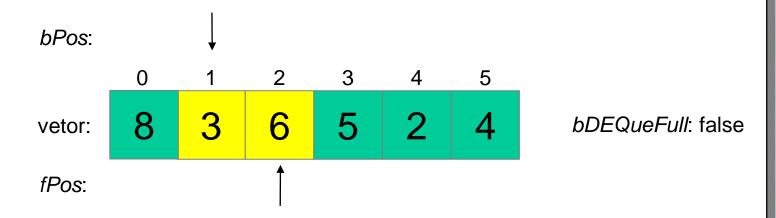
Verifica se está cheia

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 94/130



DEQue

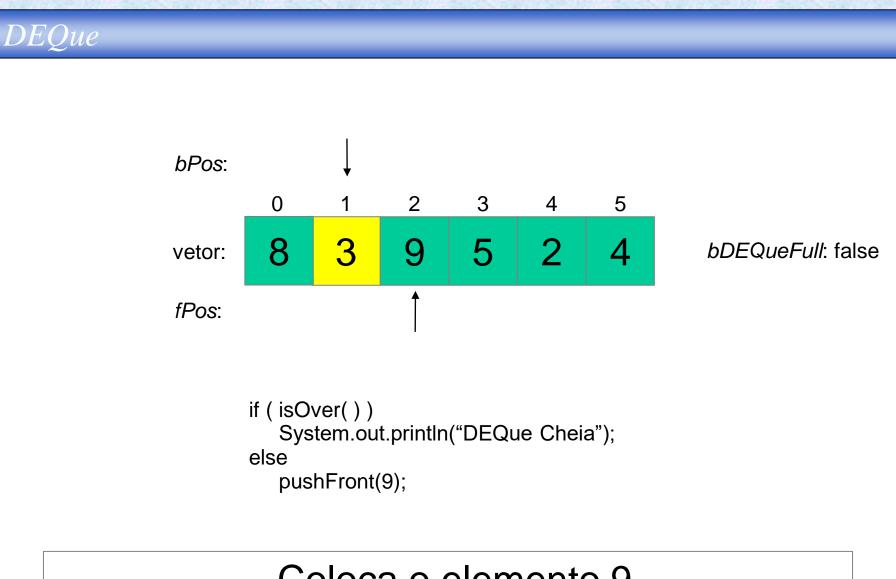


```
if ( isOver( ) )
    System.out.println("DEQue Cheia");
else
    pushFront(9);
```

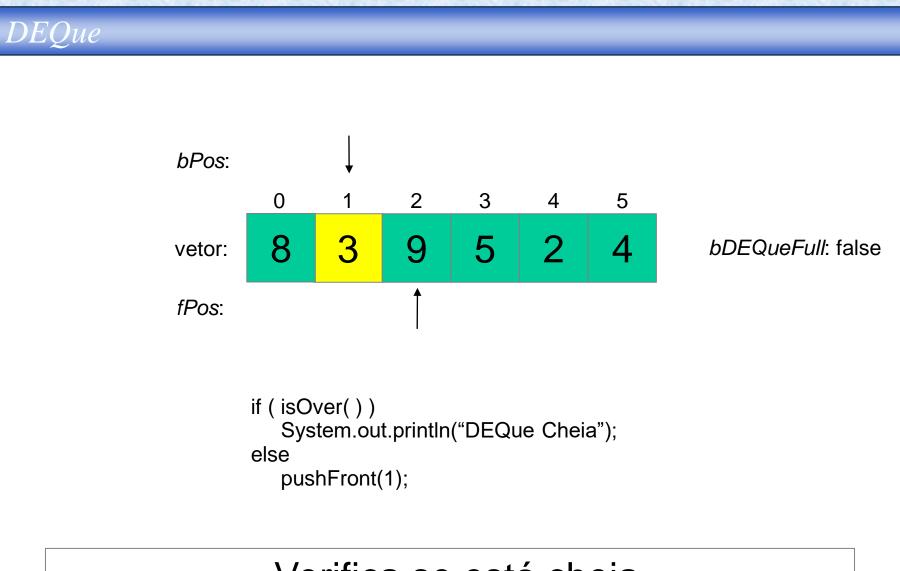
Não está cheia! Decrementa fPos (front)

.: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 95/130



Coloca o elemento 9



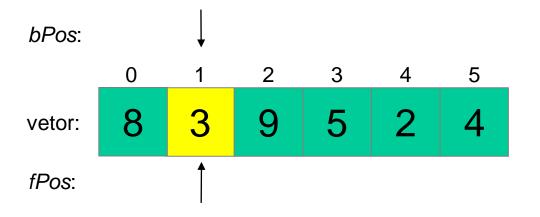
Verifica se está cheia

Prof. Calvetti

97/130



DEQue



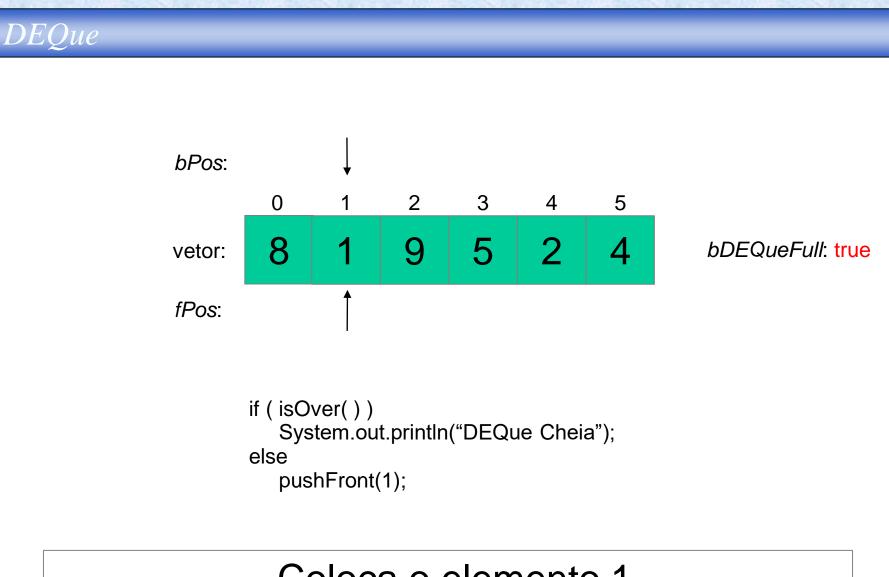
bDEQueFull: false

```
if ( isOver( ) )
    System.out.println("DEQue Cheia");
else
    pushFront(1);
```

Não está cheia! Decrementa fPos (front)

or: Prof. Robson Calvetti - Todos os direitos reservados ©.

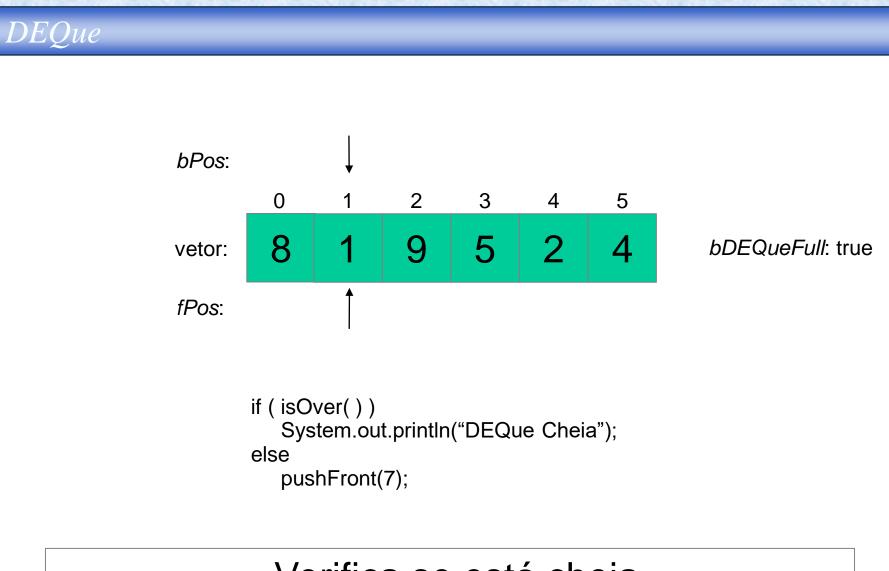
Prof. Calvetti 98/130



Coloca o elemento 1

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.

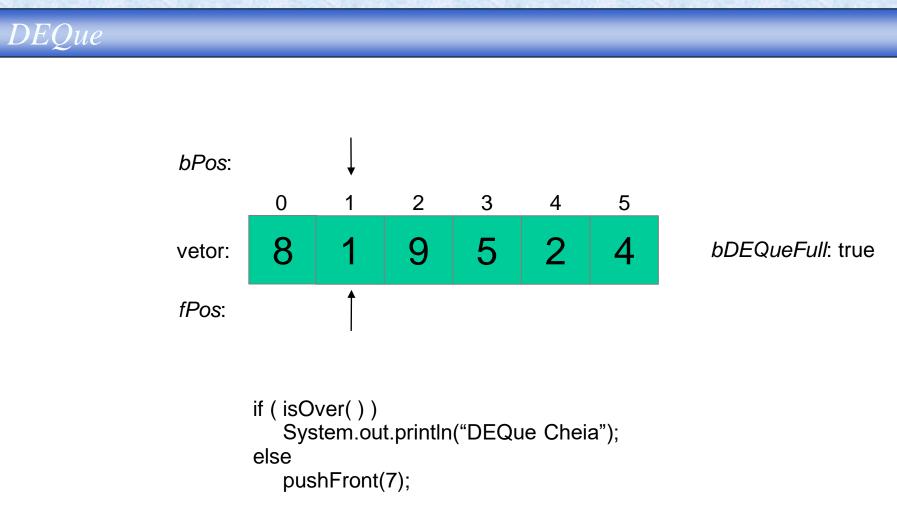
Prof. Calvetti 99/130



Verifica se está cheia

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.

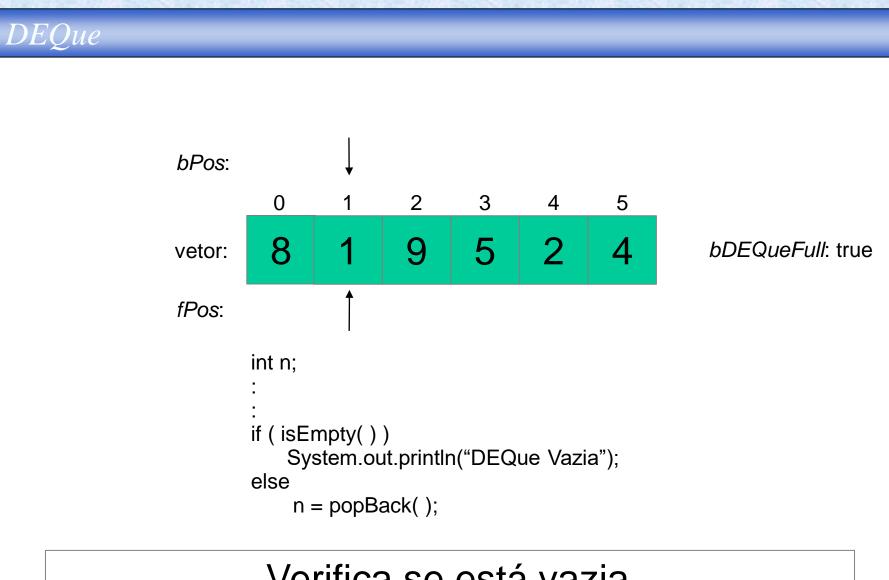
Prof. Calvetti 100/130



Está cheia! Não coloca o elemento 7

or: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 101/130



Verifica se está vazia

Prof. Calvetti 102/130

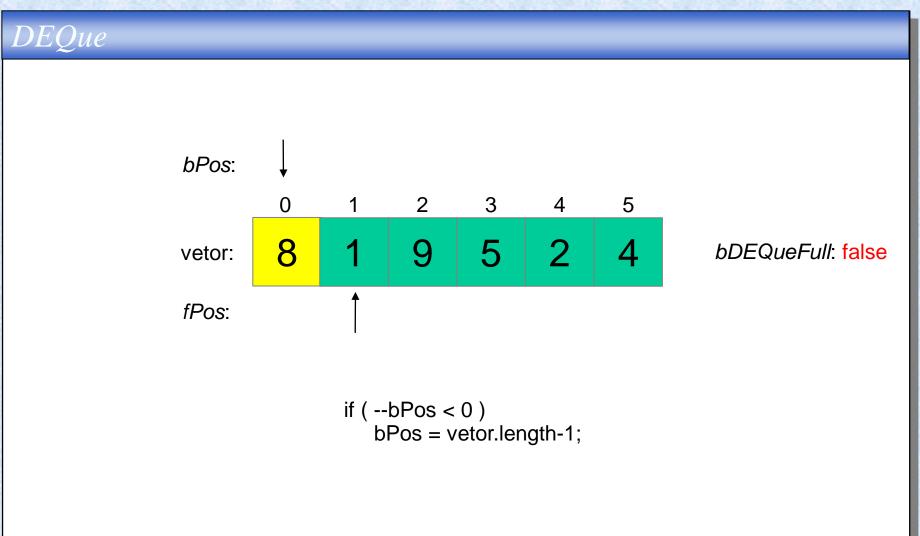
son Calvetti - Todos os direitos reservados ©.

Estruturas de Dados e Análise de Algoritmos

DEQue bPos: bDEQueFull: true vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popBack(); // n: 8

Não está vazia! Retira o elemento de trás

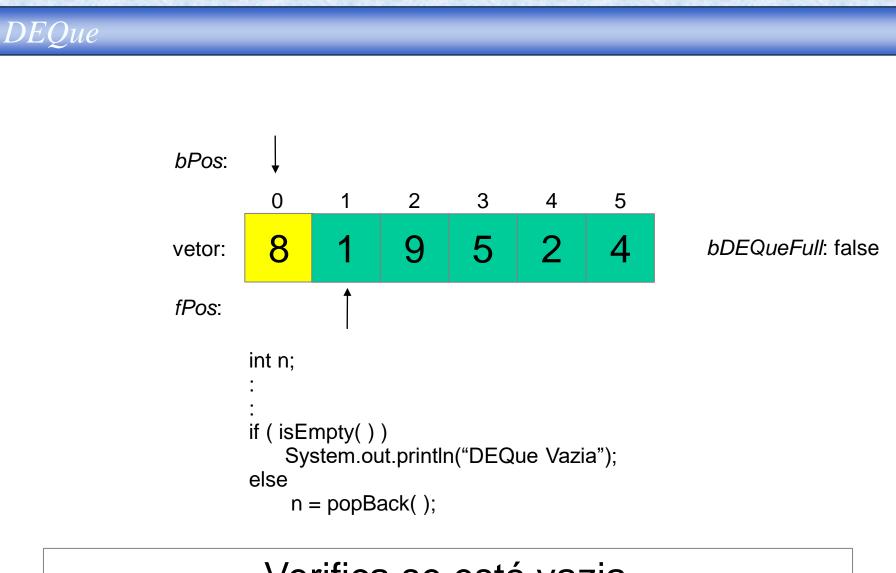
Prof. Calvetti 103/130



Decrementa bPos (back)

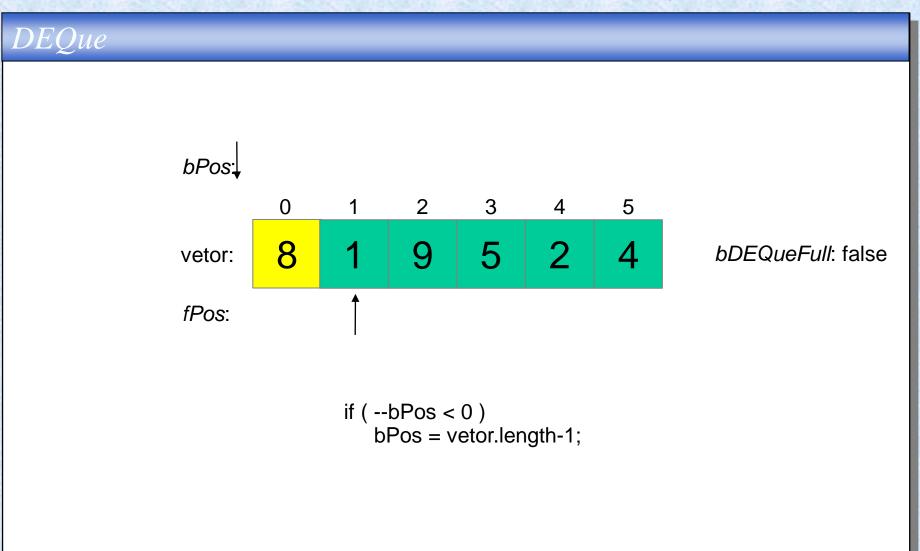
Prof. Calvetti

104/130



Verifica se está vazia

Autor: Prof. Robson Calvetti - Todos os direitos reservado



Decrementa bPos (back)

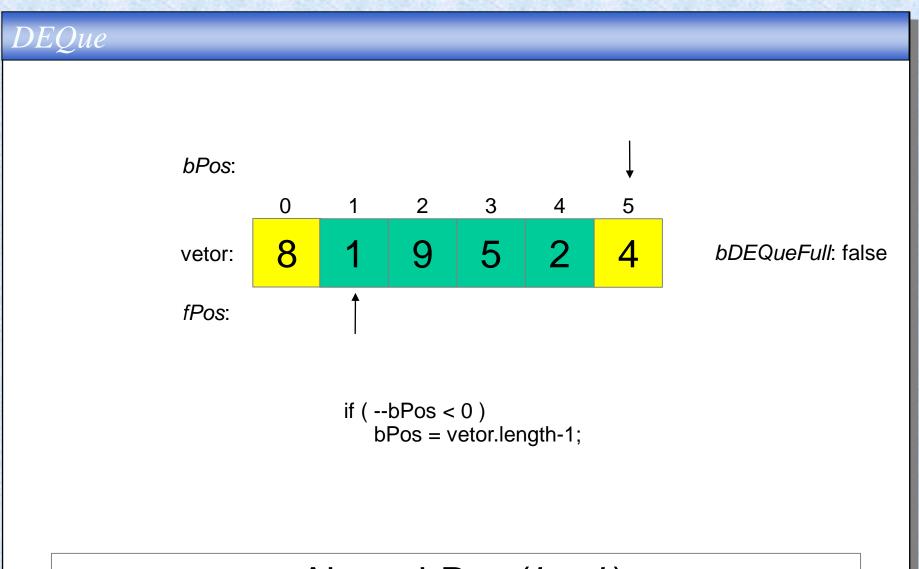
Prof. Calvetti 106/130

DEQue bPos: bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popBack(); // n: 4

Não está vazia! Retira o elemento de trás

Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

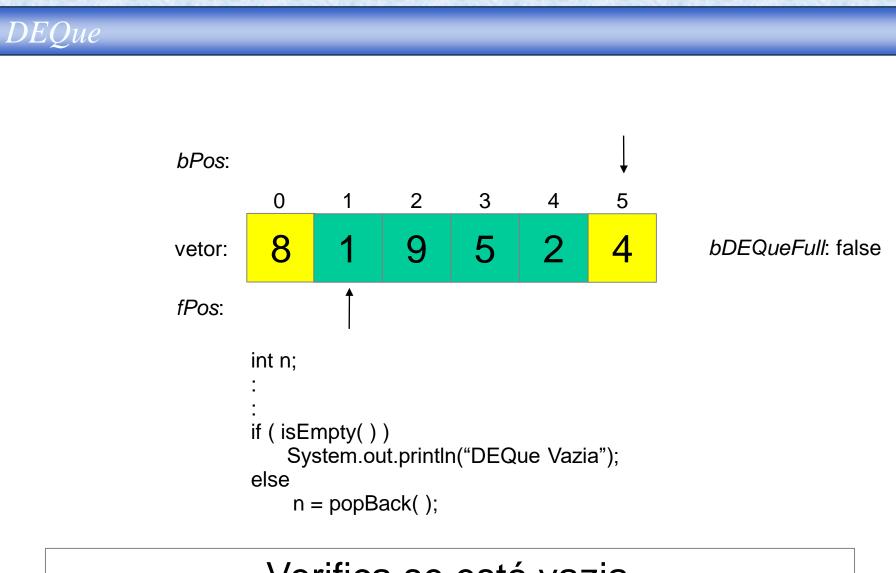
Prof. Calvetti 107/130



Ajusta bPos (back)

Prof. Calvetti

108/130



Verifica se está vazia

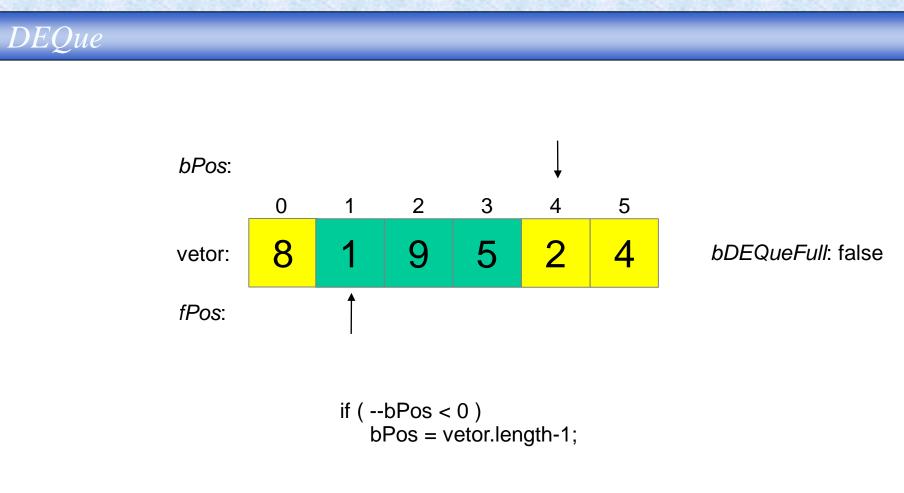
4utor: Prof. Robson Calvetti - Todos os direitos reservados ©

Prof. Calvetti 109/130

DEQue bPos: bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else n = popBack(); // n: 2

Não está vazia! Retira o elemento de trás

Prof. Calvetti 110/130

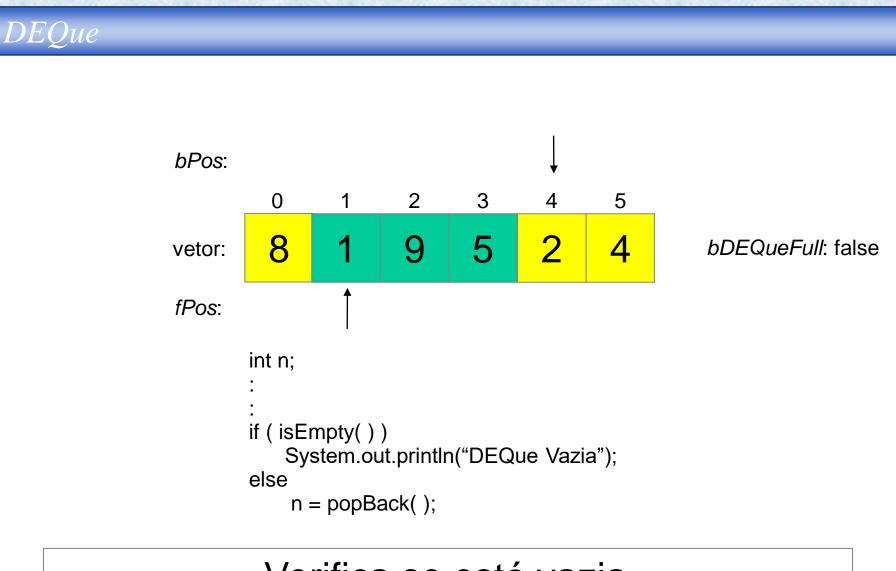


Decrementa bPos (back)

Autor: Prof. Robson Calvetti - Todos os direitos reservad

111/130

Prof. Calvetti



Verifica se está vazia

Prof. Calvetti 112/130

DEQue bPos: 5 bDEQueFull: false vetor: fPos: int n; if (isEmpty()) System.out.println("DEQue Vazia"); else

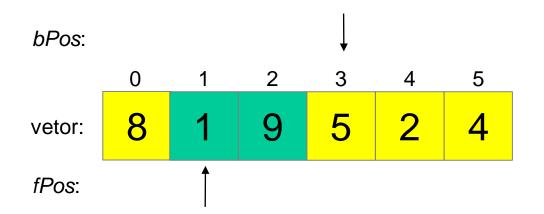
Não está vazia! Retira o elemento de trás

n = popBack();

// n: 5

Prof. Calvetti 113/130

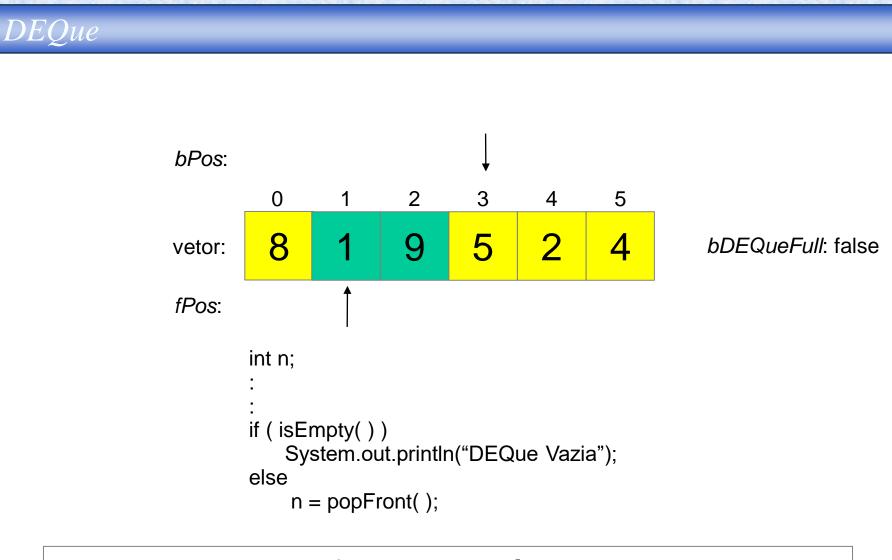
DEQue



bDEQueFull: false

Decrementa bPos (back)

Prof. Calvetti 114/130



Verifica se está vazia

Autor: Prof. Robson Calvetti - Todos os direitos reservad

Prof. Calvetti 115/130

bDEQueFull: false

if (isEmpty())

n = popFront();

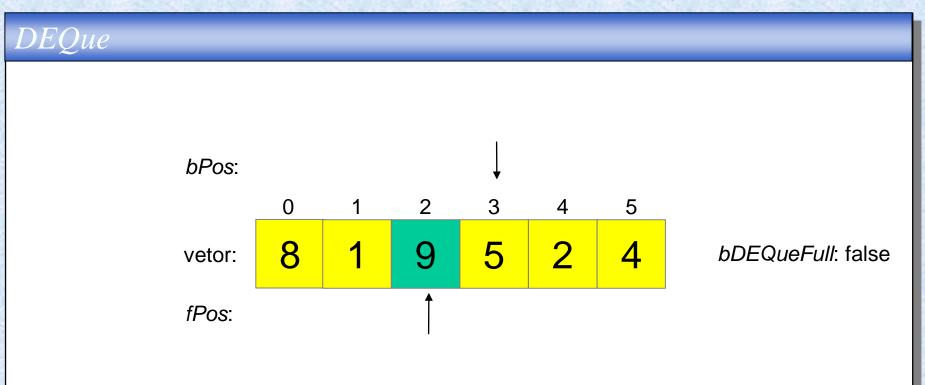
else

// n: 1

Não está vazia! Retira o elemento da frente

System.out.println("DEQue Vazia");

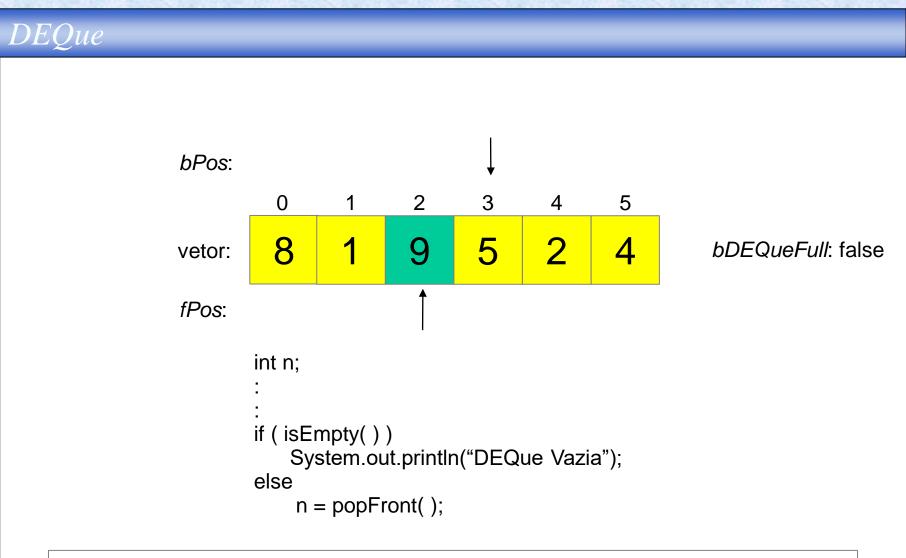
Prof. Calvetti 116/130



Incrementa fPos (front)

(front)

Prof. Calvetti



Verifica se está vazia

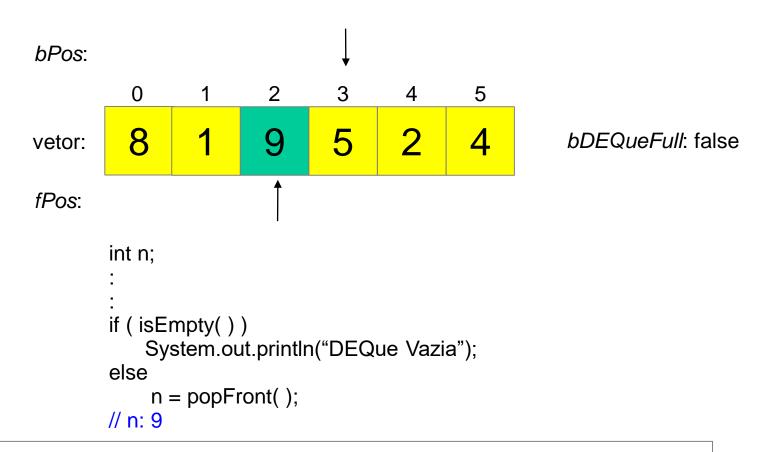
Autor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti 118/130

i - Todos os direitos reservados ©.

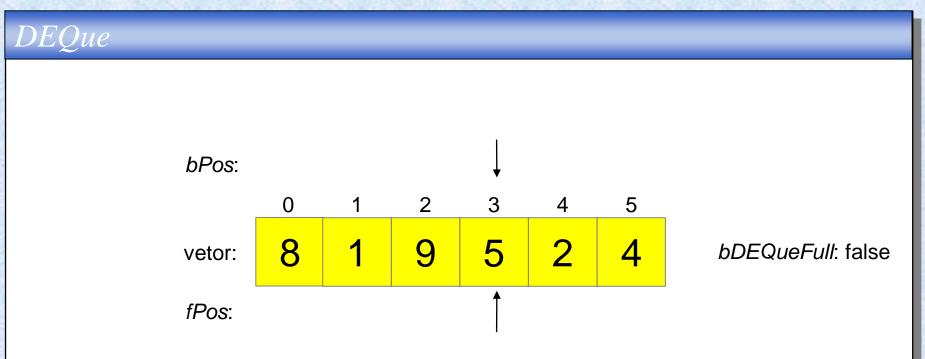
Estruturas de Dados e Análise de Algoritmos

DEQue



Não está vazia! Retira o elemento da frente

Prof. Calvetti 119/130

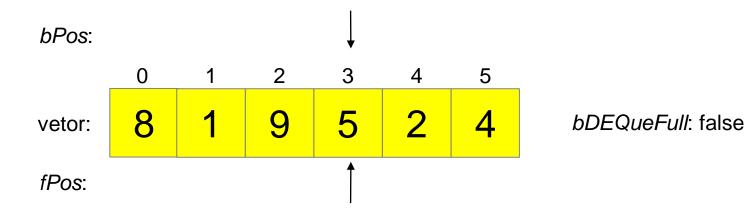


Incrementa fPos (front)

Autor: Prof. Robson Calvetti - Todos os dir

Prof. Calvetti 120/130





Repete qualquer operação: pushFront, pushBack, popFront, popBack, size, front, back, isEmpty e isFull conforme necessidade...

Insere ou retira elementos da *DEQue...*

utor: Prof. Robson Calvetti - Todos os direitos reservados ©.

Prof. Calvetti

121/130

DEQue

Exemplo de códigos escrito em Java:

Codificação dos Métodos

Prof. Calvetti

122/130

DEQue

```
static
                        iTAM = 10;
public
             int
public static
                        iDEQue [] = new int [iTAM];
            int
public static
                        iBPos = 0;
            int
public static int iFPos = 0;
public static
            boolean bDEQueFull = false;
public static int size()
if( iBPos >= iFPos && !bDEQueFull) return iBPos - iFPos;
      return iBPos + iDEQue.length - iFPos;
else
```

Método size()

Prof. Calvetti 123/130

DEQue

```
public static
                        iTAM = 10;
            int
public static
                        iDEQue [] = new int [iTAM];
            int
                        iBPos = 0;
public static
            int
public static int iFPos = 0;
public static boolean bDEQueFull = false;
public static int back()
if( iBPos == 0 ) return iDEQue[ iDEQue.length - 1 ];
return iDEQue[iBPos – 1];
public static int front()
return iDEQue[iFPos];
```

Métodos back() e front()

Prof. Calvetti 124/130

DEQue

```
public static
                        iTAM = 10;
            int
public static int
                        iDEQue [] = new int [iTAM];
public static int
                        iBPos = 0;
public static int iFPos = 0;
public static boolean bDEQueFull = false;
public static boolean isOver()
if( iBPos == iFPos && bDEQueFull ) return
                                        true:
return false;
public static boolean isEmpty()
if( iBPos == iFPos && !bDEQueFull ) return
                                        true:
return false;
```

Métodos isOver() e isEmpty()

Prof. Calvetti 125/130

DEQue

```
static
                         iTAM = 10;
public
              int
public static
                         iDEQue [] = new int [iTAM];
            int
public static
                         iBPos = 0;
            int
public static int
                        iFPos = 0;
public static
            boolean bDEQueFull = false;
public static int popBack()
if(--iBPos < 0) iBPos = iDEQue.length - 1;
bDEQueFull = false;
return iDEQue[iBPos];
```

Método popBack()

Prof. Calvetti 126/130

DEQue

```
static
                         iTAM = 10;
public
              int
public static
                         iDEQue [] = new int [iTAM];
            int
public static
                         iBPos = 0;
            int
public static int
                        iFPos = 0;
public static
            boolean bDEQueFull = false;
public static int popFront()
int iIndice = iFPos++;
if( iFPos >= iDEQue.length ) iFPos = 0;
bDEQueFull = false;
return iDEQue[iIndice];
```

Método popFront()

Prof. Calvetti 127/130

DEQue

```
static
                        iTAM = 10;
public
             int
public
     static
                        iDEQue [] = new int [iTAM];
            int
                        iBPos = 0;
public static
            int
public static
            int iFPos = 0;
public static
            boolean bDEQueFull = false;
                     pushBack( int iN )
public static void
iDEQue[ iBPos++ ] = iN;
if( iBPos >= iDEQue.length ) iBPos = 0;
if( iBPos == iFPos ) bDEQueFull = true;
```

Método pushBack()

Prof. Calvetti 128/130

DEQue

```
static
                         iTAM = 10;
public
              int
public
      static
                         iDEQue [] = new int [iTAM];
            int
                         iBPos = 0;
public static
            int
public static
                        iFPos = 0;
            int
public static
            boolean bDEQueFull = false;
public static void pushFront(int iN)
if( --iFPos < 0 ) iFPos = iDEQue.length - 1;
if( iFPos == iBPos ) bDEQueFull = true;
iDEQue[ iFPos ] = iN;
```

Método pushFront()

Prof. Calvetti 129/130

FIM

Prof. Calvetti

130/130