# Processamento da Informação

## Vetores – Parte 2
## (Arrays / Arranjos )

Prof. Jesús P. Mena-Chalco
CMCC/UFABC

Q2/2018

# Terminologia : Palavra reservada new

```
int v[] = new int[10];
```

Usada para criar um novo **objeto**.
Em Java, uma vetor é um objeto.

```
int a[]    = new int[100];
double b[] = new double[200];
float c[]  = new float[1000];
```

| Category | Types | Size (bits) |
|---|---|---|
| Integer | byte | 8 |
| | char | 16 |
| | short | 16 |
| | int | 32 |
| | long | 64 |
| Floating-point | float | 32 |
| | double | 64 |
| Other | boolean | 1 |
| | void | -- |

# Exercício da aula anterior:
## Soma dos elementos

```
static int somarElementos(int v[]) {
    int i;
    int soma = 0;

    for (i=0; i<v.length; i=i+1) {
        soma = soma + v[i];
    }

    return soma;
}
```

# Exercício da aula anterior:
## Maior elemento

```
static int maiorElemento(int v[]) {
    int i;
    int maior = v[0];

    for (i=1; i<v.length; i=i+1) {
        if (maior<v[i])
            maior = v[i];
    }

    return maior;
}
```

# Exercício da aula anterior:
## Busca de um elemento

```
static int buscaElemento (int v[], int x) {
    int i=v.length-1;

    while (i>=0 && x!=v[i]) {
        i = i-1;
    }

    return i;
}
```

## HOMEWORK **LISTA 2**

CONFIGURE | EDIT

| | |
|---|---|
| DISCIPLINE | PI-2018-Q2 - Diurno - SA |
| CREATED | June 30, 2018 at 2:43 PM |
| UPDATED | July 5, 2018 at 3:43 PM |
| LANGUAGES | JAVA JAVA 8 |

**6/30/18**
START

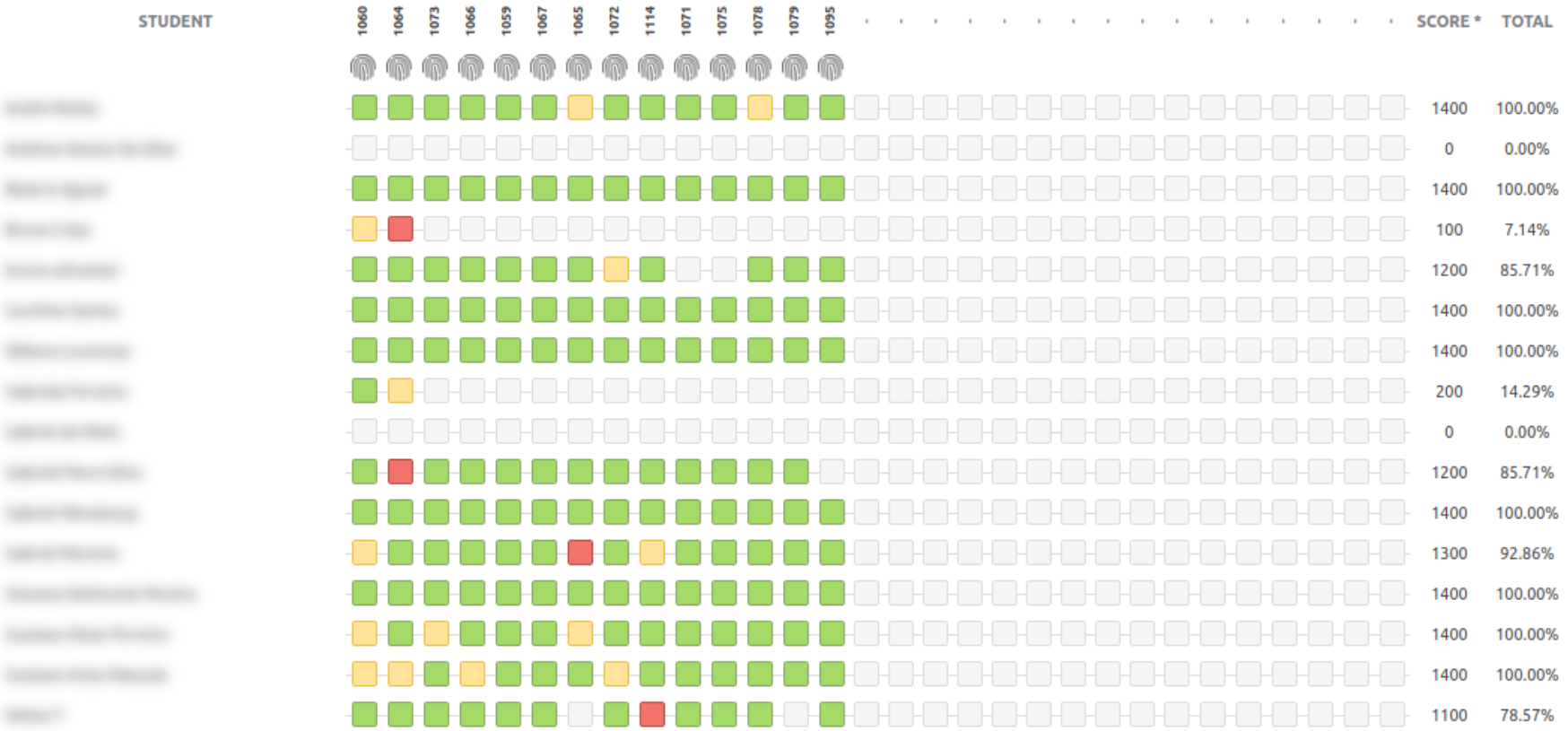
**7/16/18**
DEADLINE

**14**
PROBLEMS

**15**
DAYS TO SOLVE

## PROGRESS **Lista 2**

STUDENTS STATISTICS



## ©MOSS **SCAN**

MOSS last scanned this homework for plagiarism on **July 15, 2018 at 11:03 PM**

## ©MOSS **PLAGIARISM**

| # | POSSIBLE ORIGIN | SUBMISSION | RATE | LINES | POSSIBLE DESTINATION | SUBMISSION | RATE | OPTIONS |
|---|---|---|---|---|---|---|---|---|
| 1 | [redacted] | 10933830 | 32% | 11 | [redacted] | 10893306 | 30% | |
| 2 | [redacted] | 10933830 | 32% | 2 | [redacted] | 10934884 | 29% | |
| 3 | [redacted] | 10933830 | 32% | 2 | [redacted] | 10942324 | 29% | |
| 4 | [redacted] | 10934454 | 61% | 9 | [redacted] | 10941166 | 62% | |
| 5 | [redacted] | 10934454 | 61% | 9 | [redacted] | 10941479 | 62% | |
| 6 | [redacted] | 10934884 | 80% | 12 | [redacted] | 10914277 | 79% | |
| 7 | [redacted] | 10934884 | 52% | 14 | [redacted] | 10941686 | 55% | |
| 8 | [redacted] | 10937187 | 36% | 7 | [redacted] | 10891979 | 39% | |
| 9 | [redacted] | 10937187 | 40% | 4 | [redacted] | 10920224 | 40% | |
| 10 | [redacted] | 10937187 | 34% | 7 | [redacted] | 10939668 | 35% | |
| 11 | [redacted] | 10939356 | 69% | 5 | [redacted] | 10891559 | 61% | |
| 12 | [redacted] | 10939356 | 69% | 5 | [redacted] | 10926371 | 61% | |
| 13 | [redacted] | 10939356 | 69% | 9 | [redacted] | 10934454 | 61% | |
| 14 | [redacted] | 10939356 | 89% | 15 | [redacted] | 10941166 | 81% | |
| 15 | [redacted] | 10939356 | 95% | 15 | [redacted] | 10941232 | 95% | |
| 16 | [redacted] | 10939356 | 89% | 15 | [redacted] | 10941479 | 81% | |
| 17 | [redacted] | 10939356 | 95% | 15 | [redacted] | 10942127 | 95% | |
| 18 | [redacted] | 10939356 | 95% | 15 | [redacted] | 10942217 | 95% | |
| 19 | [redacted] | 10939817 | 32% | 14 | [redacted] | 10891559 | 33% | |
| 20 | [redacted] | 10939817 | 32% | 8 | [redacted] | 10926371 | 33% | |
| 21 | [redacted] | 10939817 | 82% | 12 | [redacted] | 10928906 | 84% | |
| 22 | [redacted] | 10939817 | 32% | 8 | [redacted] | 10934454 | 33% | |
| 23 | [redacted] | 10939817 | 39% | 3 | [redacted] | 10941686 | 40% | |
| 24 | [redacted] | 10940822 | 63% | 7 | [redacted] | 10939668 | 56% | |
| 25 | [redacted] | 10941063 | 31% | 6 | [redacted] | 10940822 | 40% | |

# Atividade

# Questão 1

```
static int m1 (int a[]) {
    int q = a.length/2;
    return a[q];
}
```

O método **m1**, dado um vetor de n inteiros, devolve o elemento que está na posição $\lceil \frac{n}{2} \rceil$

# Questão 2

```
static int m2 (int a[]) {
    int i, j, acc=0;

    for (i=0; i<a.length; i=i+2) {
        acc = acc+a[i];
    }

    for (j=1; j<a.length; j=j+2) {
        acc = acc+a[j];
    }

    return acc;
}
```

O método **m2**, dado um vetor de n inteiros, devolve a somatória dos valores de todos seus elementos.

# Questão 3

```
static int m3 (int a[], int b[]) {
    return m2(a)+m2(b);
}
```

O método **m3**, dados dois vetores de inteiros, devolve a somatória dos valores de todos seus elementos.

# Questão 4

```
static double m4 (int a[]) {
    int i, soma=0;

    for (i=a.length-1; i>=0; i=i-1) {
        soma = soma+a[i];
    }

    return (double)soma/a.length;
}
```

O método **m4**, dado um vetor de inteiros, devolve a média aritmética simples.

# Questão 5

```
static int m5 (int a[]) {
    int i, q=0;

    for (i=a.length-1; i>=0; i=i-1) {
        if (a[i]<=a[i]-a[i])
            q = q+1;
    }

    return q;
}
```

O método **m5**, dado um vetor de inteiros, devolve a quantidade total de elementos menores ou iguais a zero.

# Questão 6

```
static void m6 (int a[]) {
    int i;
    int m1 = a[0];
    int m2 = a[0];

    for (i=a.length-1; i>=0; i=i-1) {
        if (m1 > a[i]) {
            m1 = a[i];
        }
        if (m2 < a[i]) {
            m2 = a[i];
        }
    }

    System.out.println(m1);
    System.out.println(m2);
}
```

O método **m6**, dado um vetor de inteiros, imprime o menor e o maior elemento do vetor. **Min-Max**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| v = | 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |

v.lenght = 8

```
static void m6 (int a[]) {
    int i;
    int m1 = a[0];
    int m2 = a[0];

    for (i=a.length-1; i>=0; i=i-1) {
        if (m1 > a[i]) {
            m1 = a[i];
        }
        if (m2 < a[i]) {
            m2 = a[i];
        }
    }

    System.out.println(m1);
    System.out.println(m2);
}
```

| i | m1 | m2 |
|---|---|---|
| | 20 | 20 |
| 7 | 0 | 20 |
| 6 | 0 | 20 |
| 5 | 0 | 20 |
| 4 | -60 | 20 |
| 3 | -60 | 30 |
| 2 | -60 | 30 |
| 1 | -60 | 100 |
| 0 | -60 | 100 |
| -1 | | |

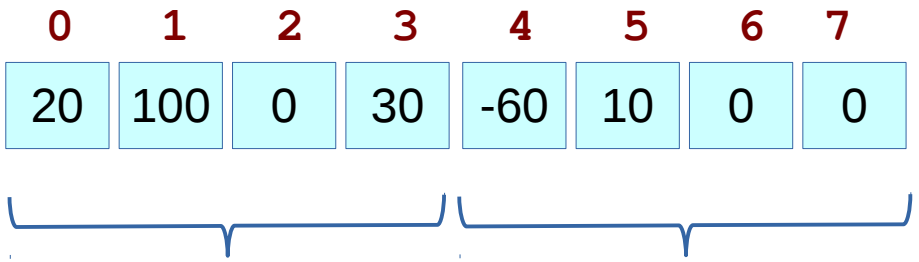
Imprime: -60
100

# Questão 7

```
static boolean m7 (int a[]) {
    int i;
    int n = a.length;
    int s1 = 0;
    int s2 = 0;

    for (i=0; i<=n/2; i=i+1) {
        s1 = s1+a[i];
        s2 = s2+a[n-i-1];
    }

    if (s1==s2)
        return true;
    else
        return false;
}
```

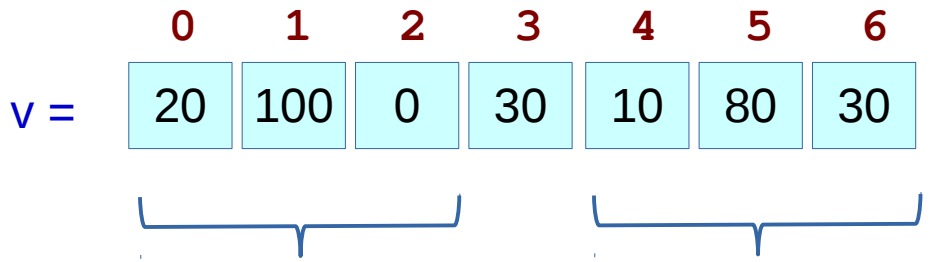
O método **m7**, dado um vetor de inteiros, devolve **true** se a somatória dos elementos da primeira metade é igual à somatória dos elementos da segunda metade. Caso contrário devolve **false**

v.lenght = 8




```java
static boolean m7 (int a[]) {
    int i;
    int n = a.length;
    int s1 = 0;
    int s2 = 0;

    for (i=0; i<n/2; i=i+1) {
        s1 = s1+a[i];
        s2 = s2+a[n-i-1];
    }

    if (s1==s2)
        return true;
    else
        return false;
}
```

v.lenght = 7




```
static boolean m7 (int a[]) {
    int i;
    int n = a.length;
    int s1 = 0;
    int s2 = 0;

    for (i=0; i<n/2; i=i+1) {
        s1 = s1+a[i];
        s2 = s2+a[n-i-1];
    }

    if (s1==s2)
        return true;
    else
        return false;
}
```

# Questão 8 - Desafio

Crie um método que permita inverter a ordem dos elementos de um array. *A inversão deve ser realizada no mesmo array* (não pode usar array auxiliar).

**Assinatura**:

```
static int[] inverter (int a[])
```

**Exemplos**:

a = [80,2,100]
Resposta: [100,2,80]

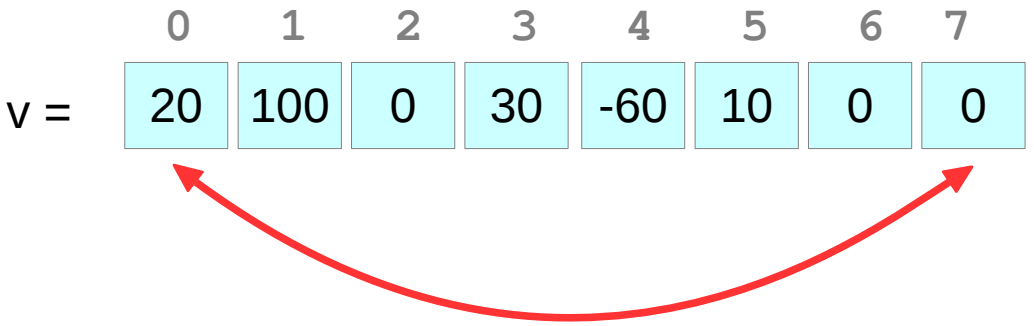
a = [1,2,3,4,555]
Resposta: [555,4,3,2,1]

# Questão 8

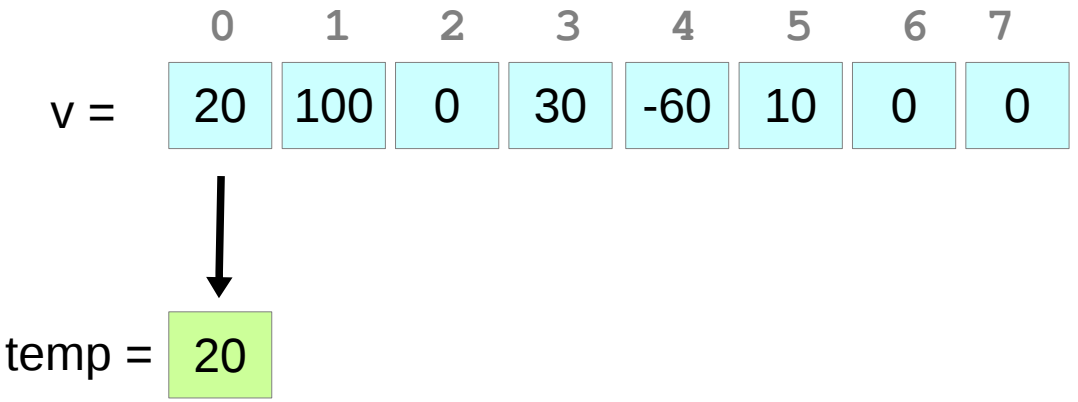
```
static int[] inverter (int a[]) {
    int i, temp, n=a.length;

    for (i=0; i<n/2; i=i+1) {
        temp = a[i];
        a[i] = a[n-i-1];
        a[n-i-1] = temp;
    }

    return a;
}
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| v = | 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |

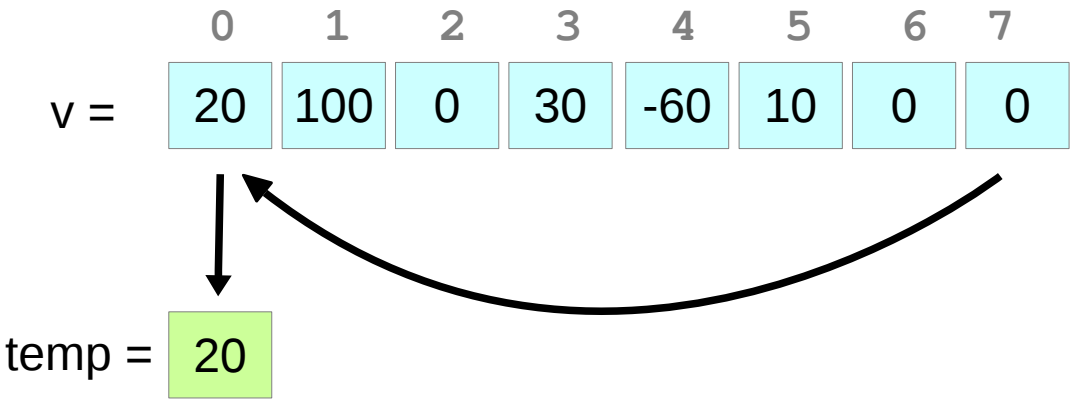
v.lenght = 8
n = 8



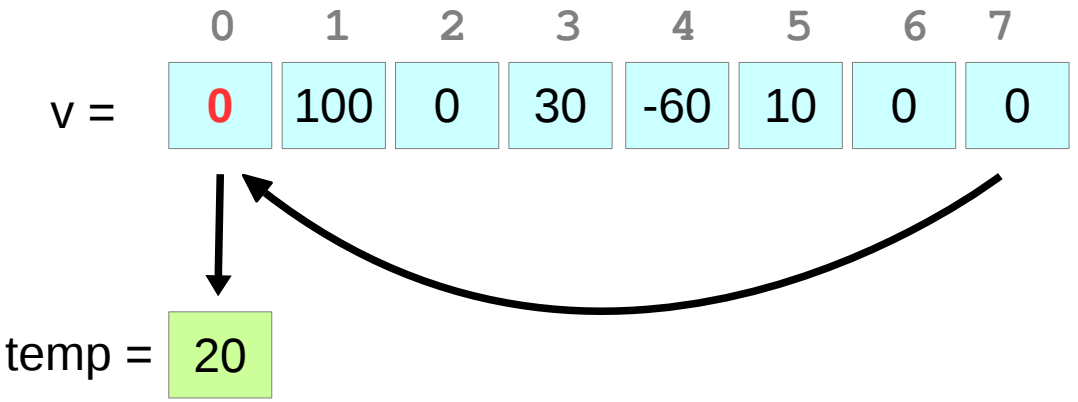

v.lenght = 8
n = 8



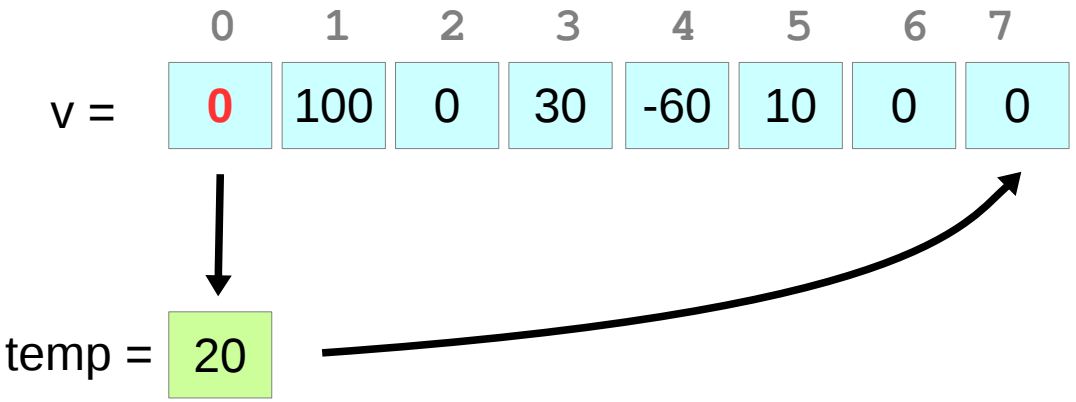

v.lenght = 8
n = 8



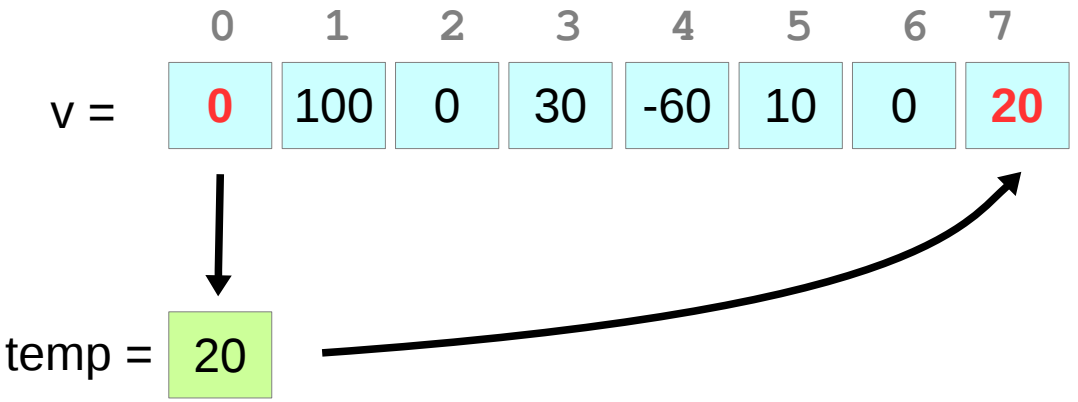

v.lenght = 8
n = 8




v.lenght = 8
n = 8




v.lenght = 8
n = 8




v.lenght = 8
n = 8

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| v = | 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |

v =

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |
| **0** | 100 | 0 | 30 | -60 | 10 | 0 | **20** |

v =

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |
| **0** | 100 | 0 | 30 | -60 | 10 | 0 | **20** |
| **0** | **0** | 0 | 30 | -60 | 10 | **100** | **20** |

v =

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |
| 0 | 100 | 0 | 30 | -60 | 10 | 0 | 20 |
| 0 | 0 | 0 | 30 | -60 | 10 | 100 | 20 |
| 0 | 0 | 10 | 30 | -60 | 0 | 100 | 20 |

v =

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |
| 0 | 100 | 0 | 30 | -60 | 10 | 0 | 20 |
| 0 | 0 | 0 | 30 | -60 | 10 | 100 | 20 |
| 0 | 0 | 10 | 30 | -60 | 0 | 100 | 20 |
| 0 | 0 | 10 | -60 | 30 | 0 | 100 | 20 |

v =

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |
| **0** | 100 | 0 | 30 | -60 | 10 | 0 | **20** |
| **0** | **0** | 0 | 30 | -60 | 10 | **100** | **20** |
| **0** | **0** | **10** | 30 | -60 | **0** | **100** | **20** |
| **0** | **0** | **10** | **-60** | **30** | **0** | **100** | **20** |

O elemento da posição i deverá ser trocado pelo elemento da posição 8-i-1

v.length = 8

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| v = | 20 | 100 | 0 | 30 | -60 | 10 | 0 | 0 |

v.lenght = 8
n = 8

```
static int[] inverter (int a[]) {
    int i, temp, n=a.length;

    for (i=0; i<n/2; i=i+1) {
        temp = a[i];
        a[i] = a[n-i-1];
        a[n-i-1] = temp;
    }

    return a;
}
```

Quando i=0
temp = 20
v[0] = 0
v[7] = 20