Curso de C

Apontadores

Apontadores

Objetivos:

- Entender o conceito de apontadores em C
- Variáveis tipo apontadores

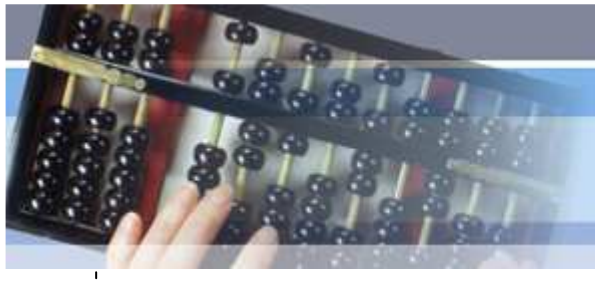
Apontadores

Apontador:

- Variável que armazena um endereço
- Endereço de outra variável, de uma função, etc.

Apontador p no endereço
100 com conteúdo 500

Inteiro x no endereço
500 com conteúdo -1



Apontadores

Declaração de variável tipo apontador:

```
tipo  *  nome;
```

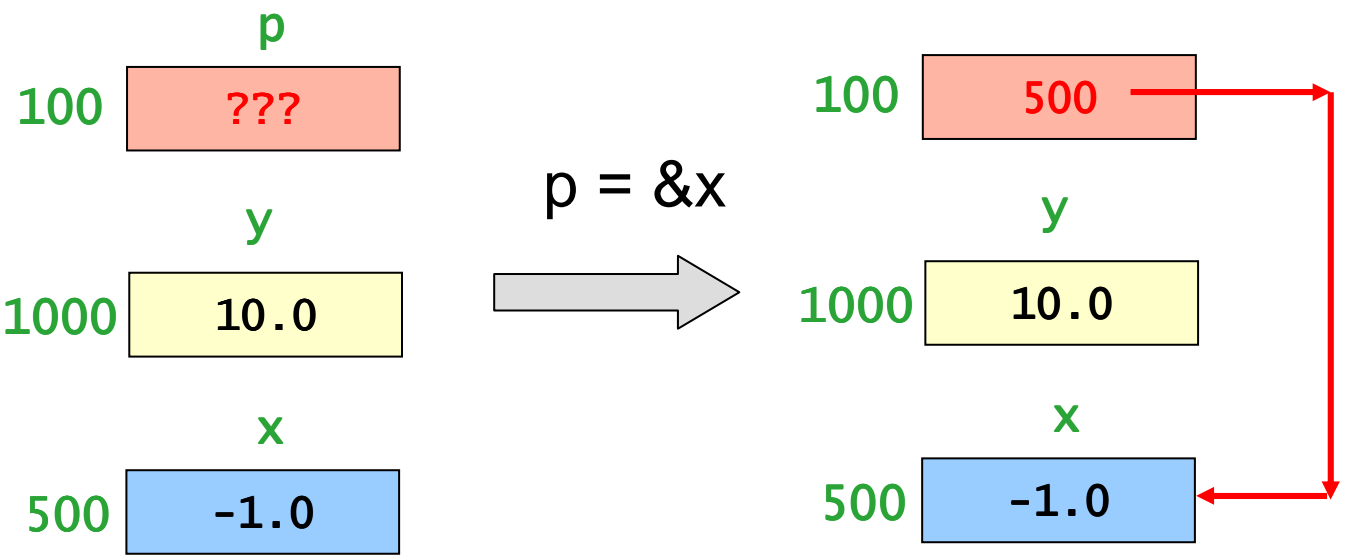
- `tipo`: Qualquer tipo da linguagem C
- `nome`: Nome do apontador

`nome` é uma variável que conterá o endereço de variáveis cujo tipo é `tipo`.

Apontadores

Operador &: retorna o endereço do operando

```
float  x=-1.0, y=10.0;
float  *p;
p = &x;
```

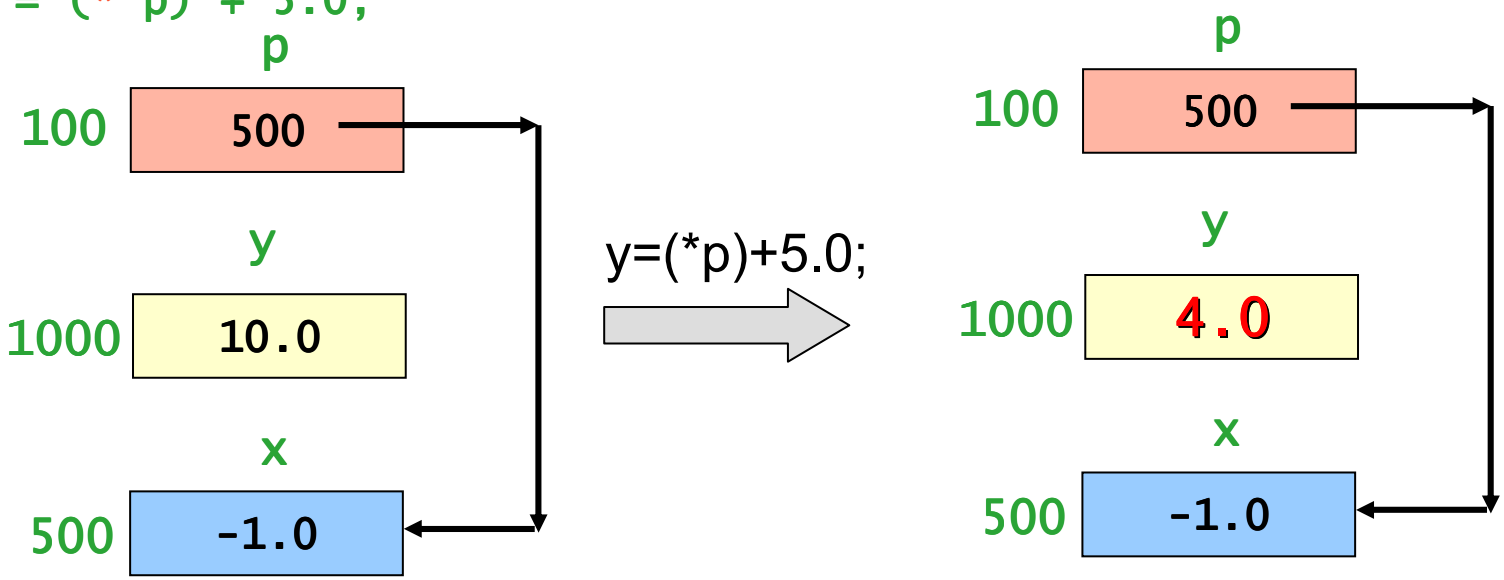


Apontadores

Operador * : **Indica que vamos usar o endereço de memória armazenado em p.**

Usado no lado direito de uma expressão, pega o valor no endereço apontado por p.

```
float  x=-1.0, y=10.0; float *p = &x;
y = (* p) + 5.0;
```

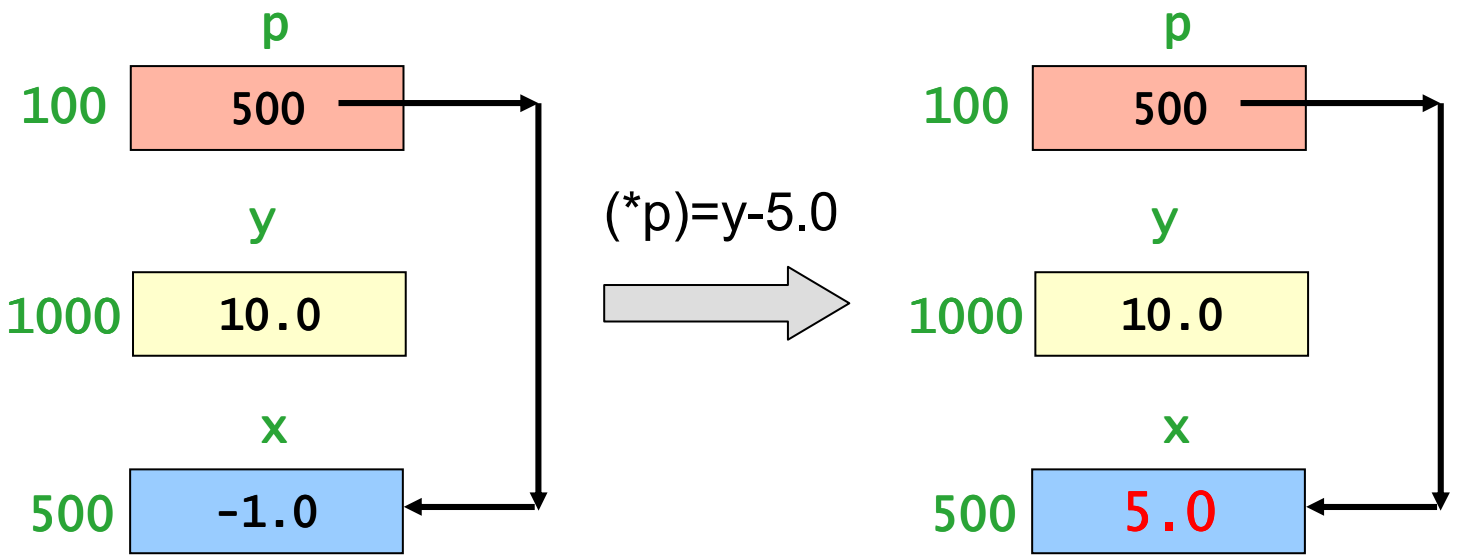


Apontadores

Operador *:

Usado no lado esquerdo da expressão,
Atribui para o endereço apontado por p.

```
float  x=-1.0, y=10.0; float *p=&x;
(*p) = y - 5.0f;
```



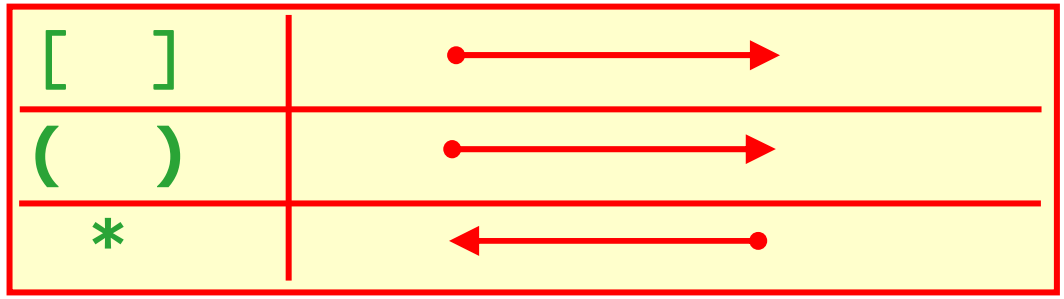
Apontadores

Declarações complexas usando [ ], ( ) e *

Precedências:

1	[ ] , ( )
2	*

Agrupamento:



Apontadores

Tipo da variável **x** é:

```
int x
```

tipo inteiro.

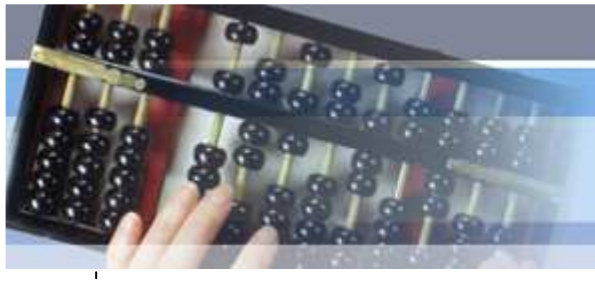
```
int *x
```

apontador para um inteiro.

```
int x[ ]
```

vetor que contém inteiros.

>



Apontadores

Tipo da variável **x** é:

```
int x( );
```

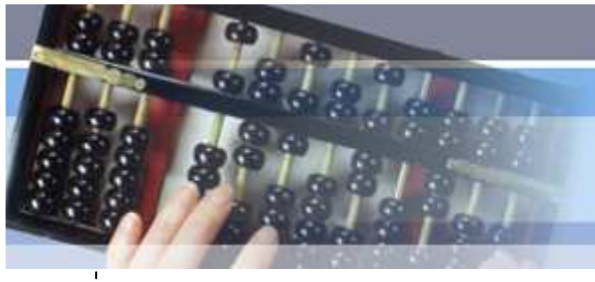
função que retorna um inteiro.

```
int *x[ ];
```

vetor que contém apontadores para inteiros.
Mesmo que `int *(x[ ]);`

```
int *x( );
```

função que retorna um apontador para inteiro.
Mesmo que `int *(x( ));`



Apontadores

- `int *x( )` `int *(x( ));`

  função que retorna um apontador para inteiro.

- `int x[ ]( )` `int (x[ ])( );`

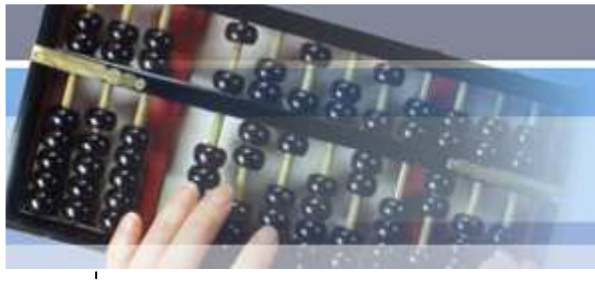
  vetor que contém função que retorna inteiros erro!

- `int x( )[ ]` `int (x( ))[ ];`

  função que retorna um vetor de inteiros. erro!

- `int x[ ][ ]` `int (x[ ])[ ];`

  vetor que contém um vetor de inteiros.



Apontadores

- `int x( )( )`

  `int (x( ))( )`

  função que retorna outra função que retorna um inteiro — erro!

- `int *x[ ]( )`

  `int *( (x[ ])( ) )`

  vetor que contém funções que retornam apontadores para inteiros — erro!

- `int *x( )[ ]`

  `int *( (x( ))[ ] )`

  função que retorna vetores de apontadores para inteiros — erro!

Apontadores

- `int *x[ ][ ]` `int *( (x([])[] )`

  vetor que contém vetores de apontadores para inteiros

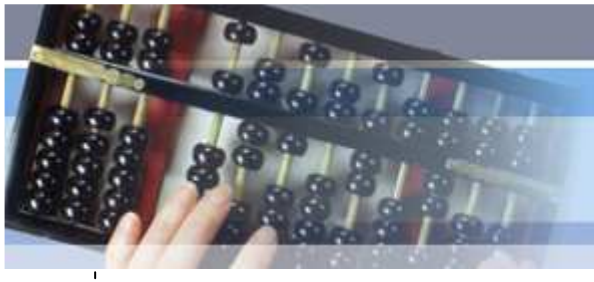
- `int *x( )( )` `int *( (x())() )`

  função que retorna uma segunda função que retorna um apontador para um inteiro

  erro!

- `int (*x)[ ]`

  apontador para um vetor de inteiros



Apontadores

- `int (*x)( )`

  apontador para uma função que retorna inteiro

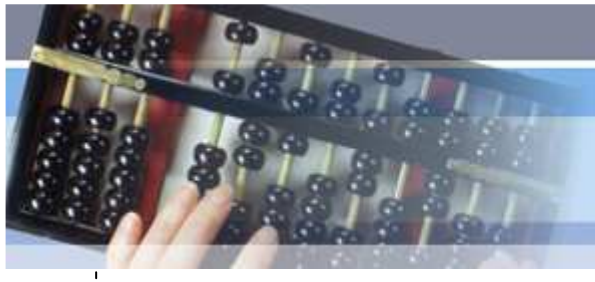
- `int **x`

  `int *(*x)`

  apontador para um apontador para um inteiro

- `int (*x[ ])( )`

  vetor de apontadores para funções que retornam inteiros



Apontadores

- `int *(*(*x)( ))[ ]`
  - `(*x)( )` apontador para uma função que retorna...
  - `(*(...))` um apontador para ...
  - `int *(...)[ ]` um vetor de apontadores para inteiros.

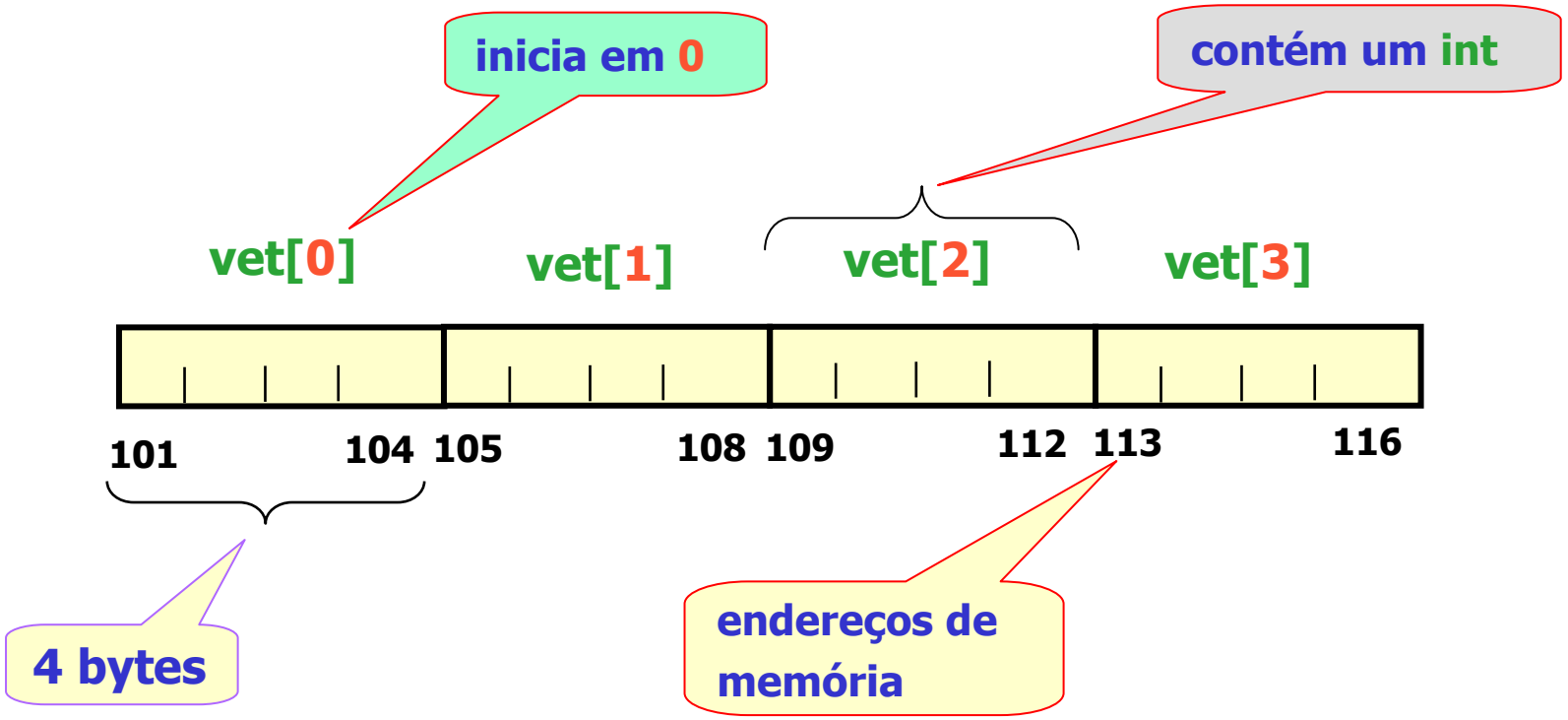
um apontador para uma função que retorna
um apontador para
um vetor de apontadores para inteiros.

>

Apontadores

Apontadores e vetores:

```
int  vet[4] ;
```

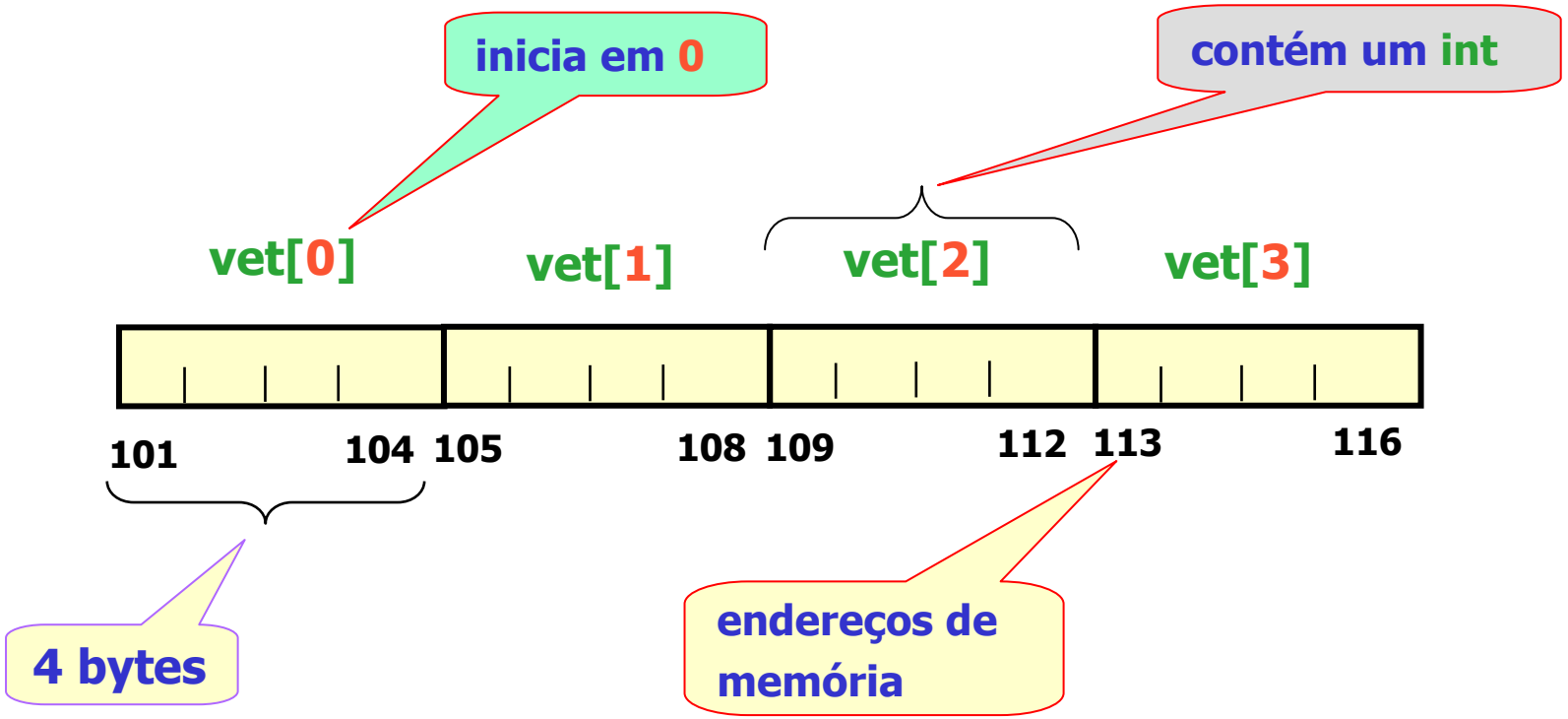


Apontadores

Apontadores e vetores:

vet 101

```
int  vet[4] ;
```



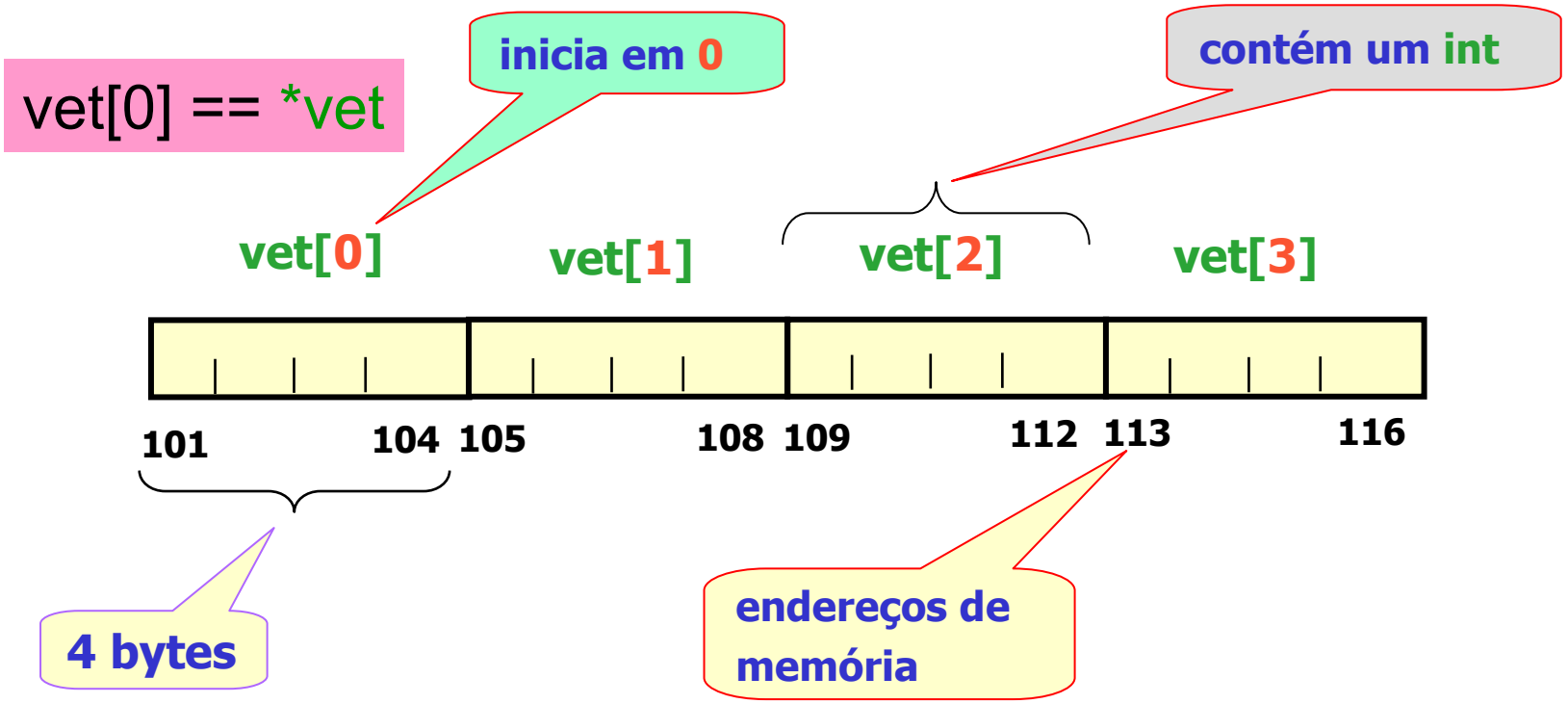
Apontadores

Apontadores e vetores:

vet 101

```
int  vet[4] ;
```

vet[0] == *vet



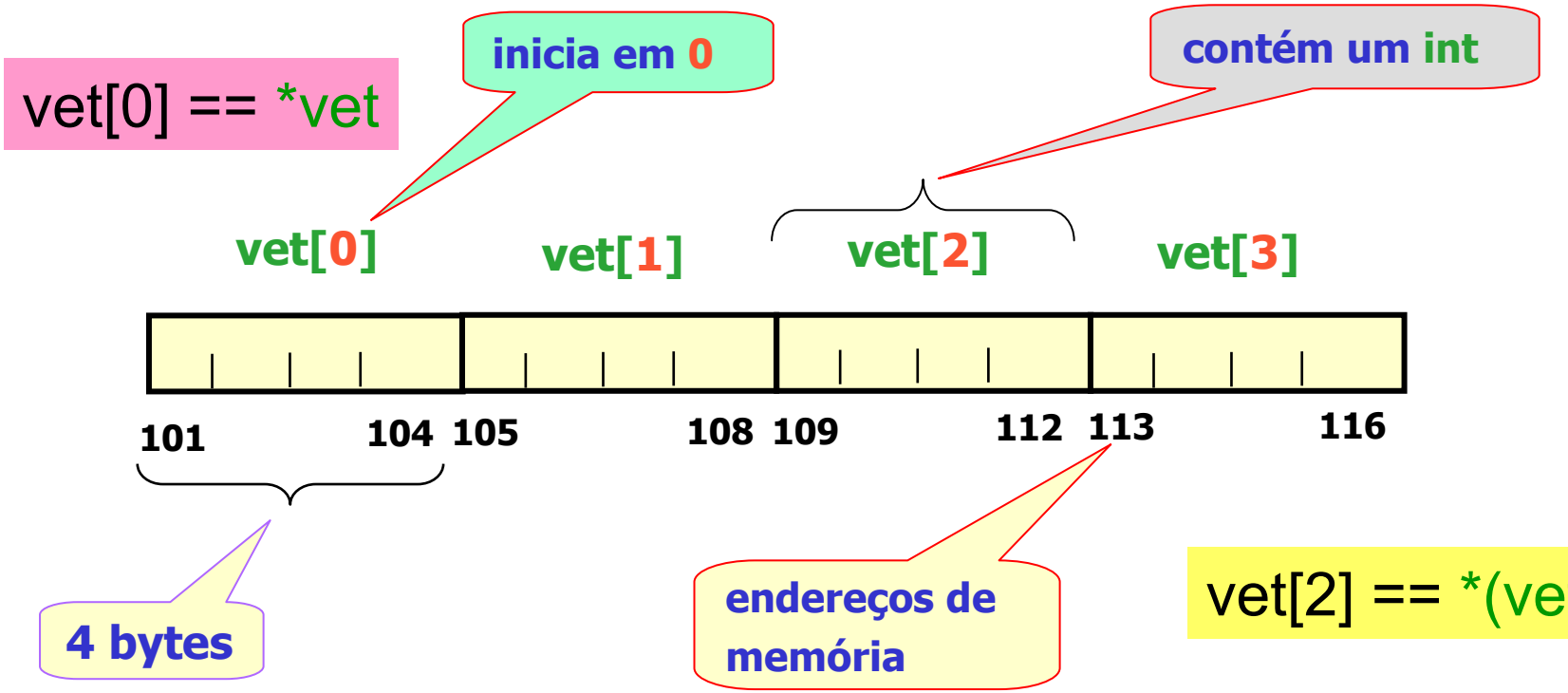
Apontadores

Apontadores e vetores:

vet 101

```
int  vet[4] ;
```

vet[0] == *vet

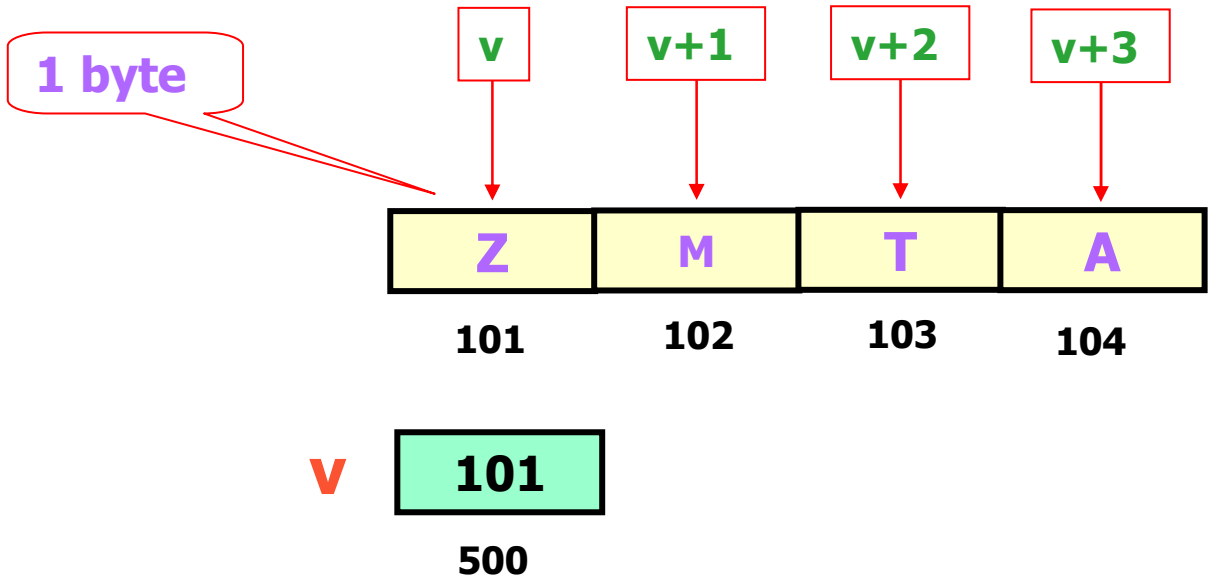


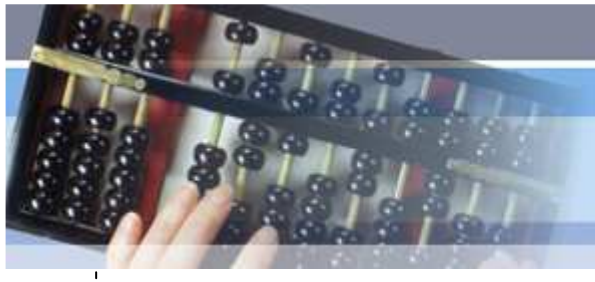
vet[2] == *(vet+2)

Apontadores

Apontadores e vetores:

char v[ ] = { 'Z', 'M', 'T', 'A' } ;

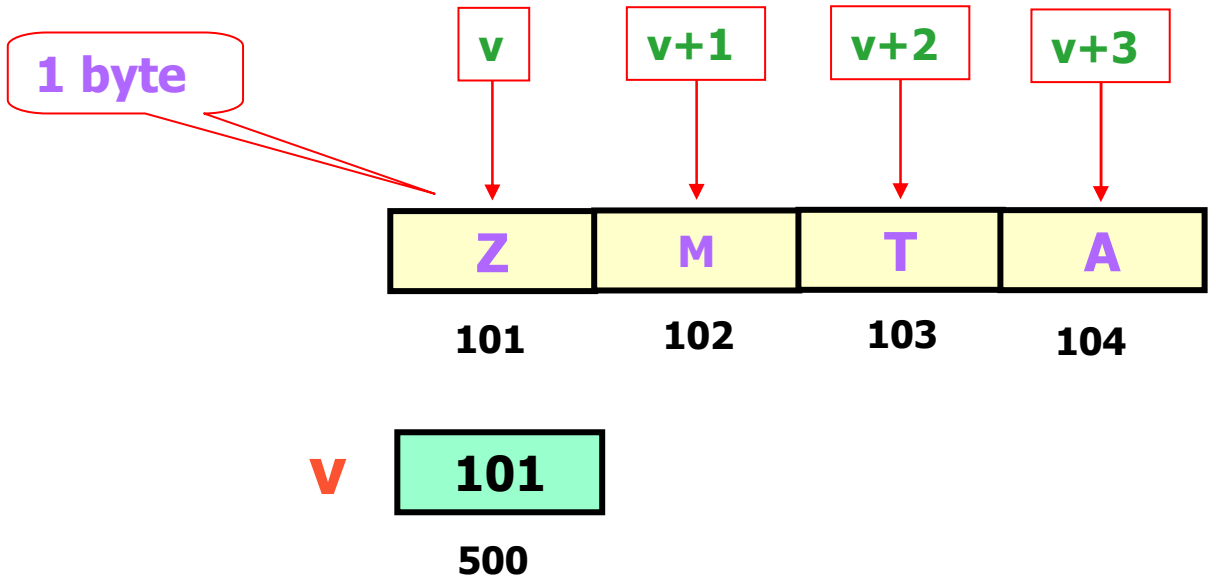




Apontadores

Apontadores e vetores:

char v[ ] = { 'Z', 'M', 'T', 'A' } ;

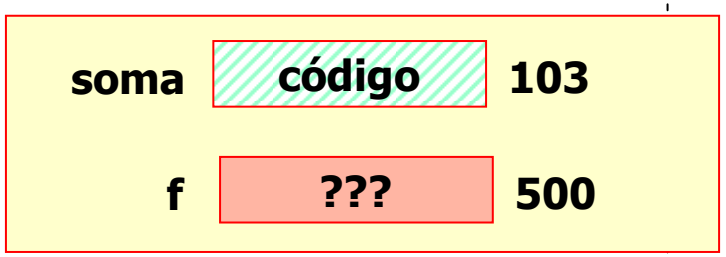


v[2] == *(v+2) == `T`

Apontadores

Exemplo: apontadores para funções

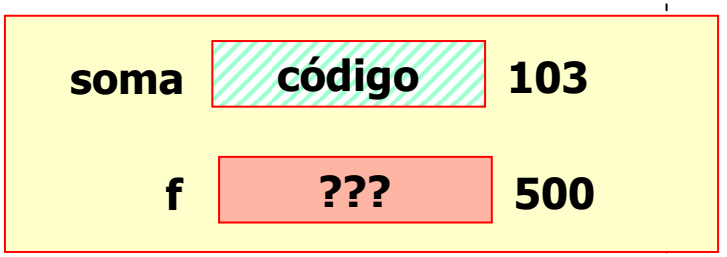
```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```



Apontadores

Exemplo: apontadores para funções

```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```




```
f=soma;
```

Apontadores

Exemplo: apontadores para funções

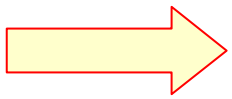
```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```

soma	código	103
f	???	500

```
f=soma;
```

soma	código	103
f	103	500

```
soma(2,3);
```



5

Apontadores

Exemplo: apontadores para funções

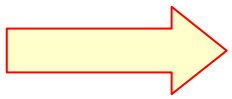
```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```

soma	código	103
f	???	500

```
f=soma;
```

soma	código	103
f	103	500

```
soma(2,3);
f (2,3);
```



5

Apontadores

Exemplo: apontadores para funções

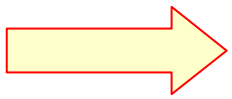
```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```

soma	código	103
f	???	500

```
f=soma;
```

soma	código	103
f	103	500

```
soma(2,3);
f (2,3);
(* soma)(2,3);
```



5

Apontadores

Exemplo: apontadores para funções

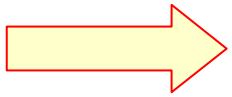
```
int soma(int x, int y) { return (x+y);}

int  (* f )( );
```

soma	código	103
f	???	500

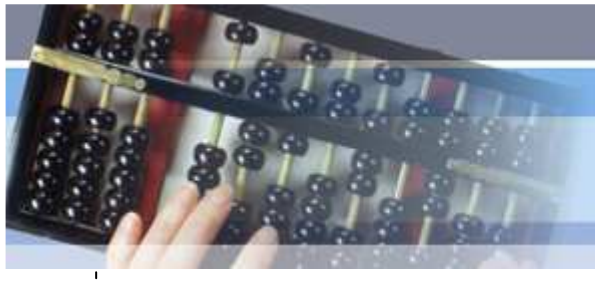
```
f=soma;
```

soma	código	103
f	103	500

```
soma(2,3);
f (2,3);
(* soma)(2,3);
(*** f )(2,3);
```

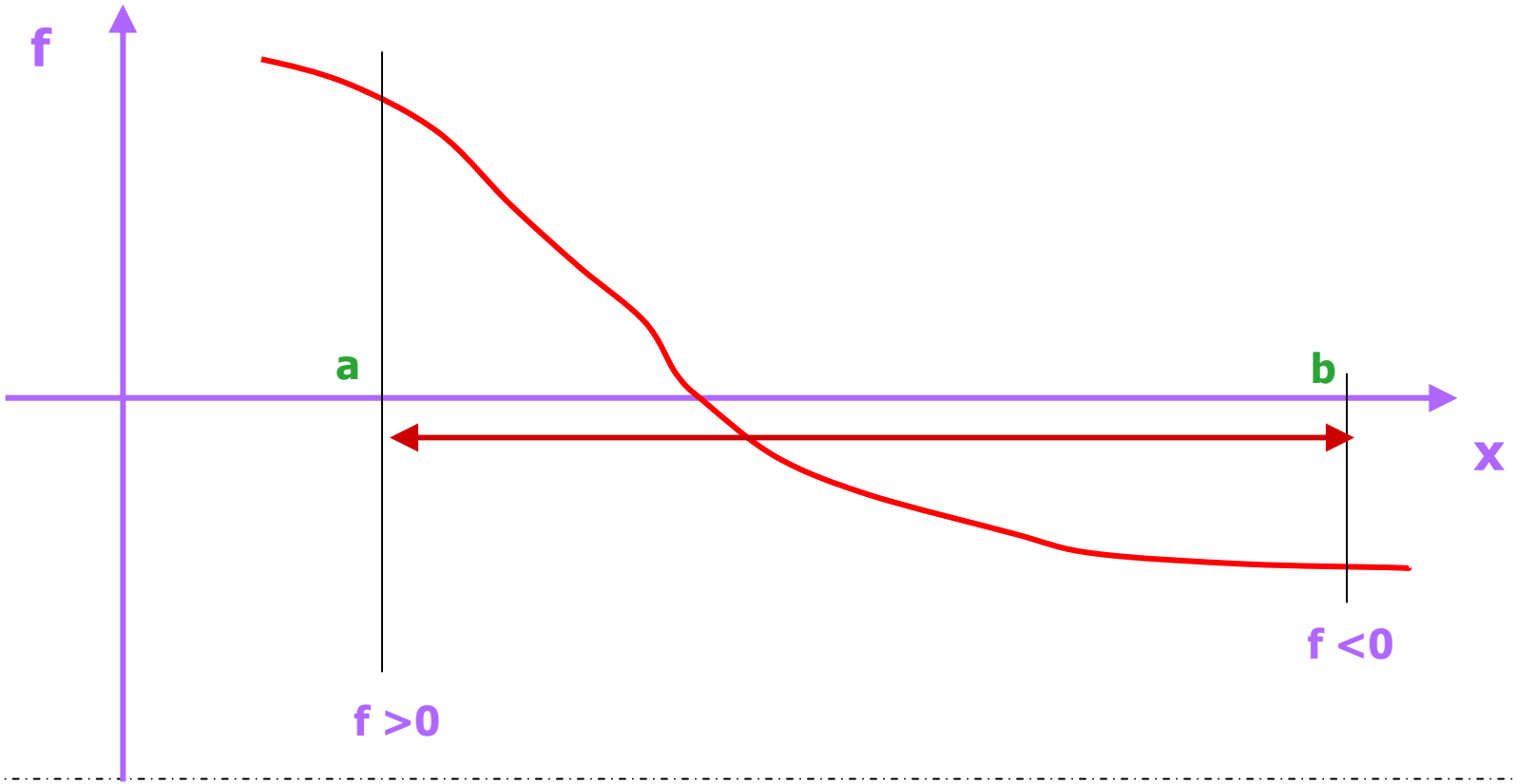


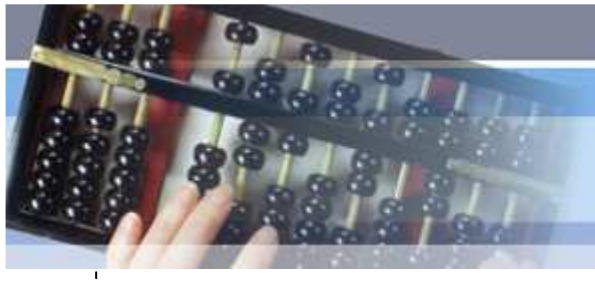
5



Apontadores

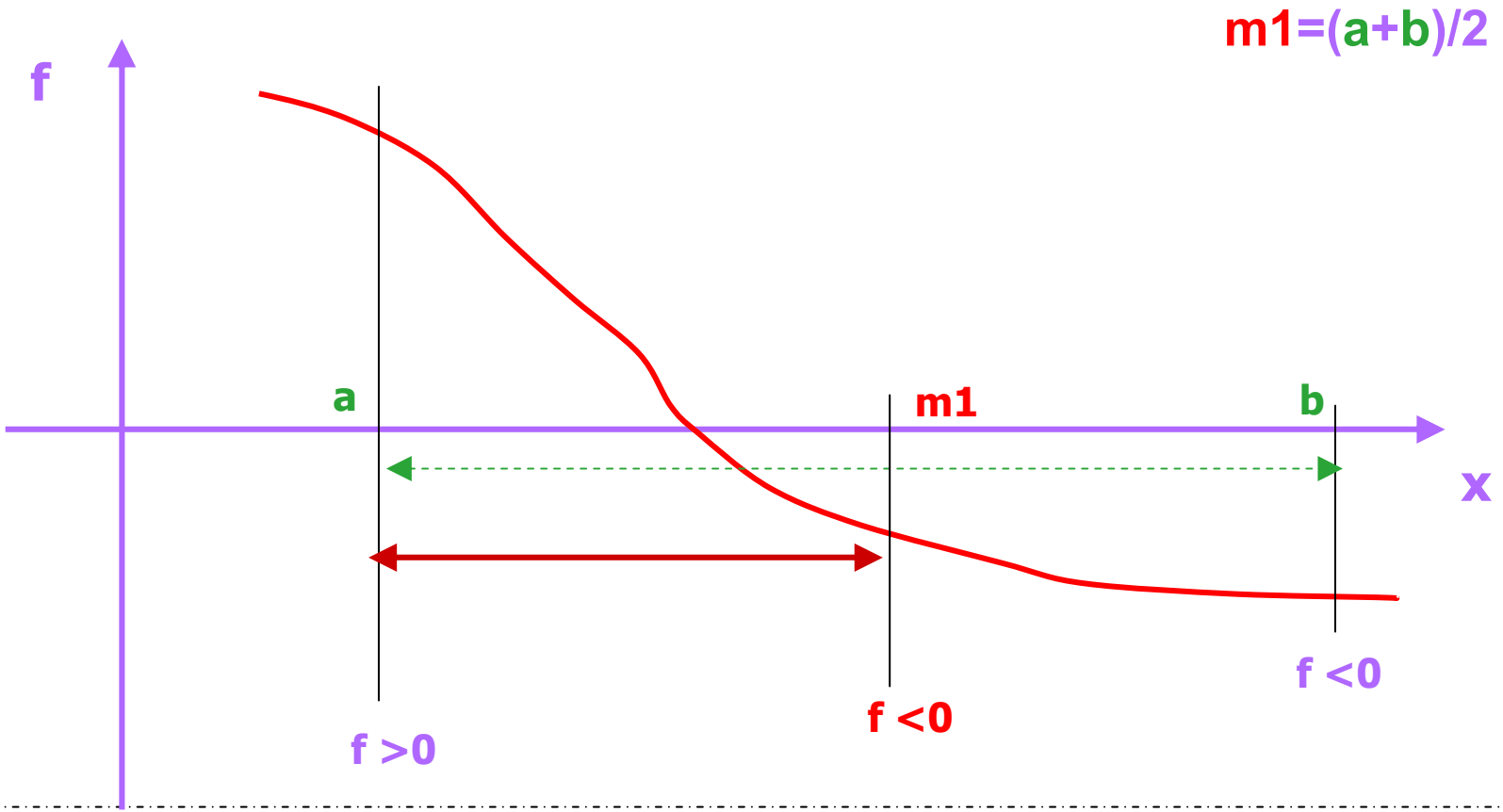
Apontadores para funções: Método de Newton

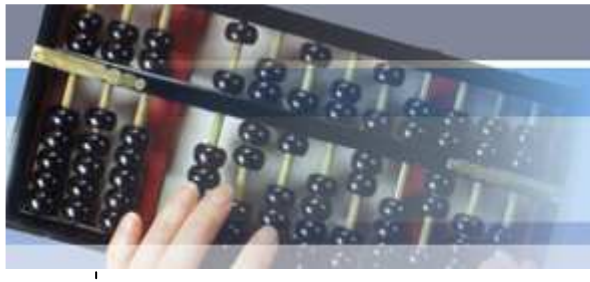




Apontadores

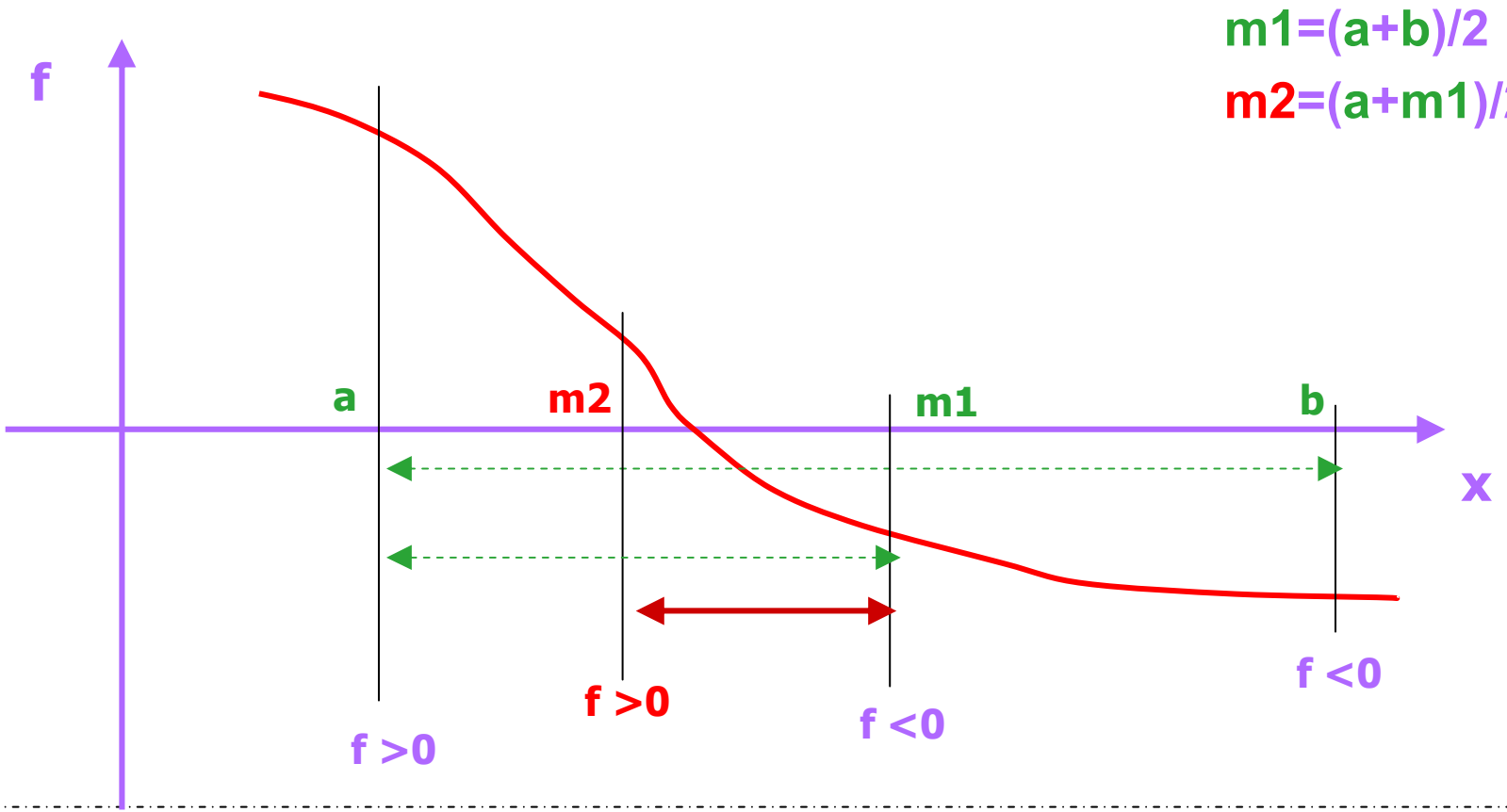
Apontadores para funções: Método de Newton

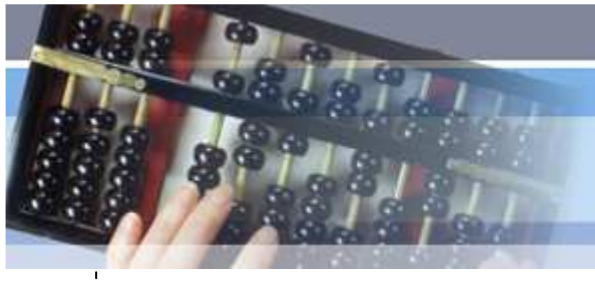




Apontadores

Apontadores para funções: Método de Newton

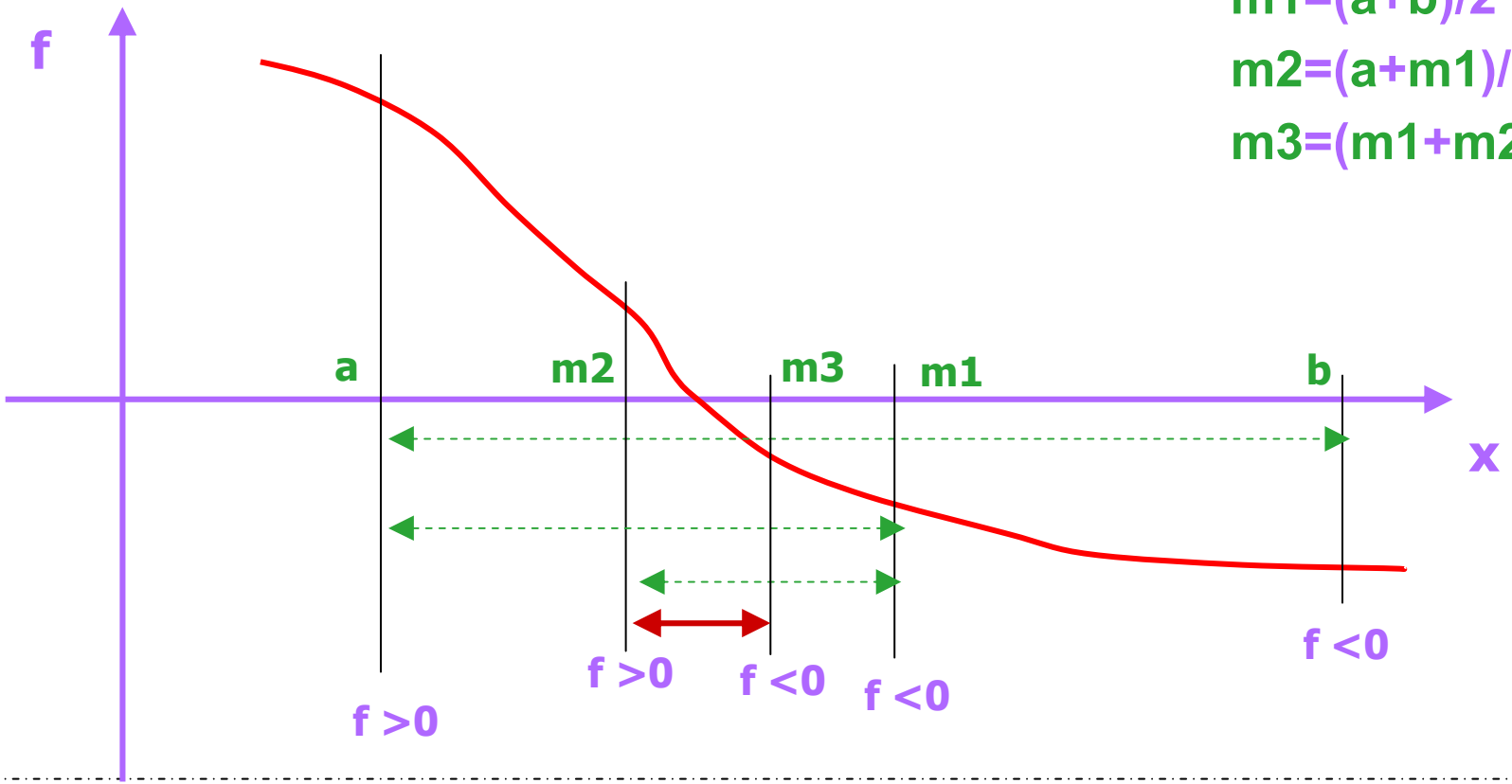




Apontadores

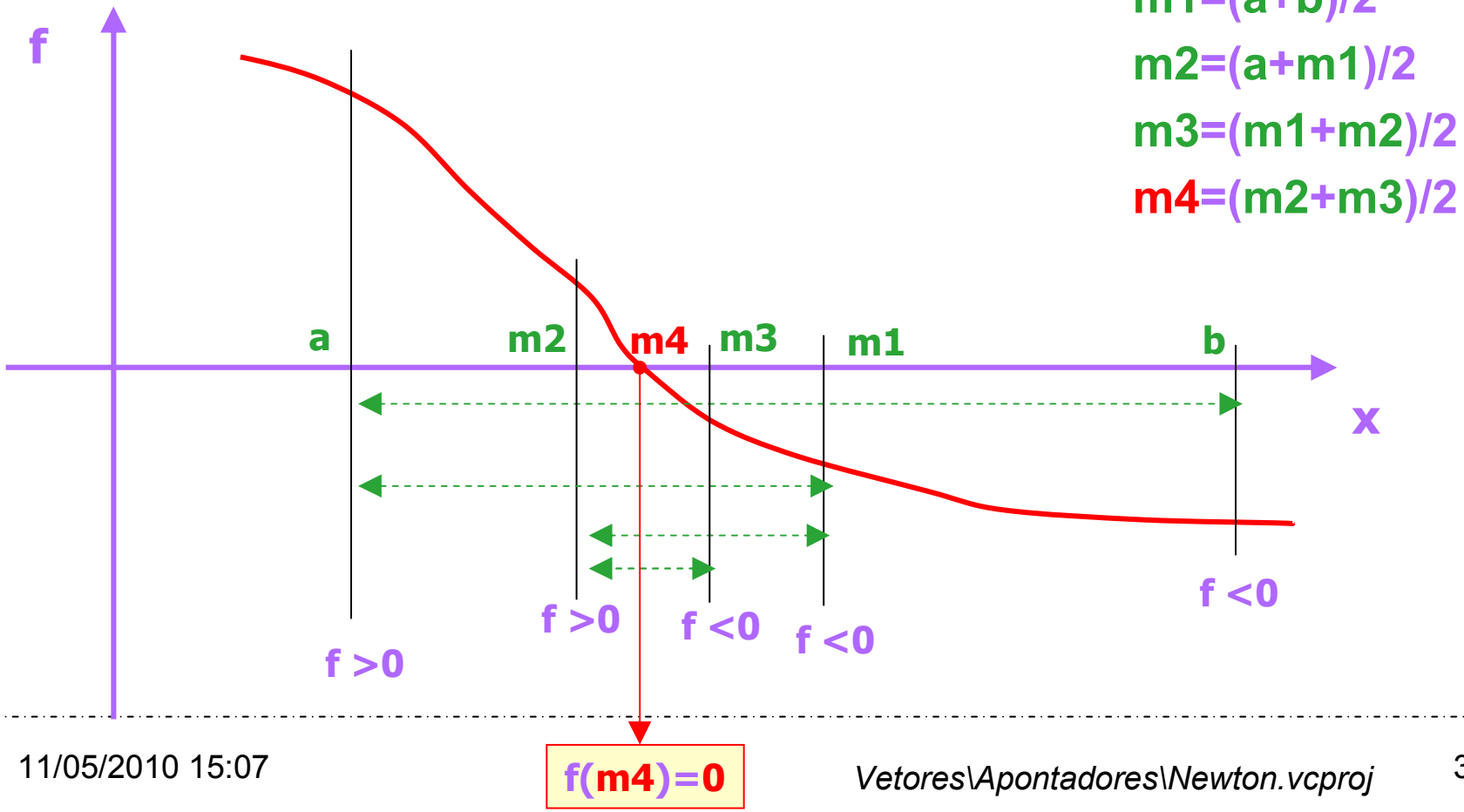
Apontadores para funções: Método de Newton

m1=(a+b)/2
m2=(a+m1)/2
m3=(m1+m2)/2



Apontadores

Apontadores para funções: Método de Newton



Apontadores

Fim