Tecnologia Em Analise e Desenv. de Sistemas

Estrutura de Dados

Aula 4 – Ponteiros – Parte 1

Prof. Dr. Ricardo Luis Balieiro

Problema

```
int obterX();

int main() {
    int x;

    x = obterX();

    cout << "Valor de X: " << x << endl;

    return 0;
}

int obterX() {
    return 100;
}
```

Problema

- Como obter o retorno das três por meio de funções?

3 funções

```
int obterX();
int obterY();
int obterZ();

int main() {
    int x, y, z;

    x = obterX();
    y = obterY();
    z = obterZ();

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    return 0;
}

int obterX() {
    return 100;
}

int obterY() {
    return 200;
}

int obterZ() {
    return 300;
}
```

Problema

- Como obter o retorno das três variáveis utilizando **uma única função**?

```
Valor de X: 0
Valor de Y: 0
Valor de Z: 0
```

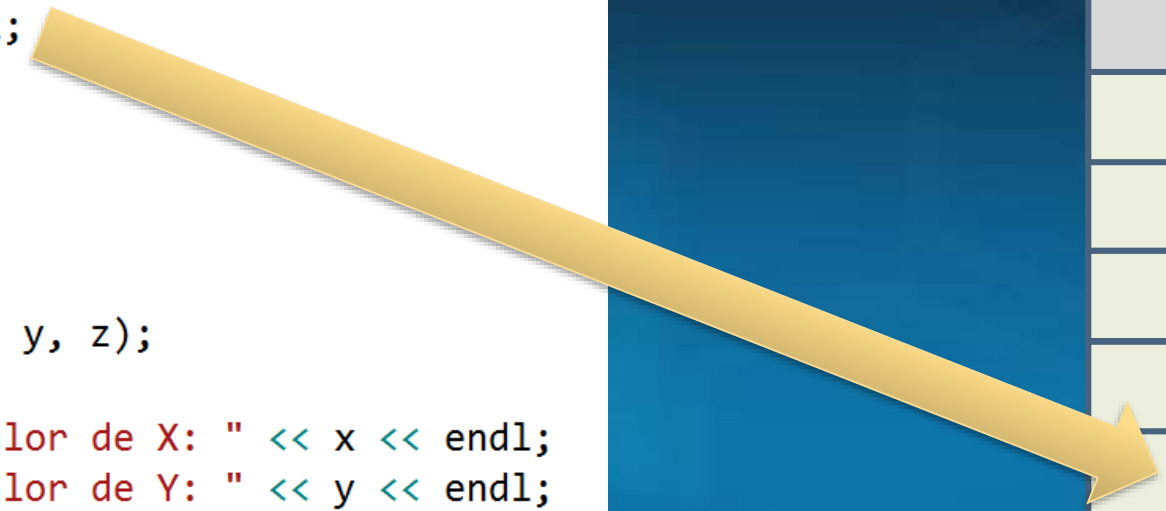
```
int obterX();
int obterY();
int obterZ();

int main() {
    int x, y, z;

    x = obterX();
    y = obterY();
    z = obterZ();

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    return 0;
}

int obterX() {
    return 100;
}

int obterY() {
    return 200;
}

int obterZ() {
    return 300;
}
```

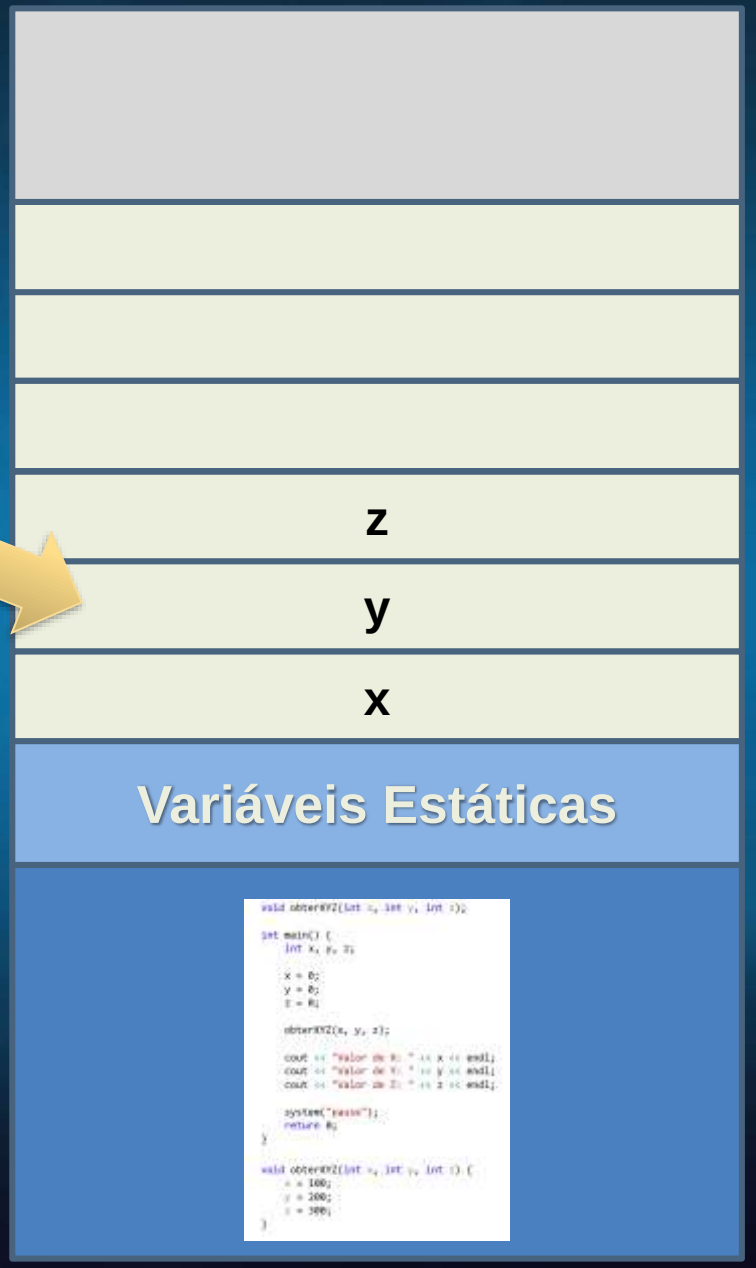
Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

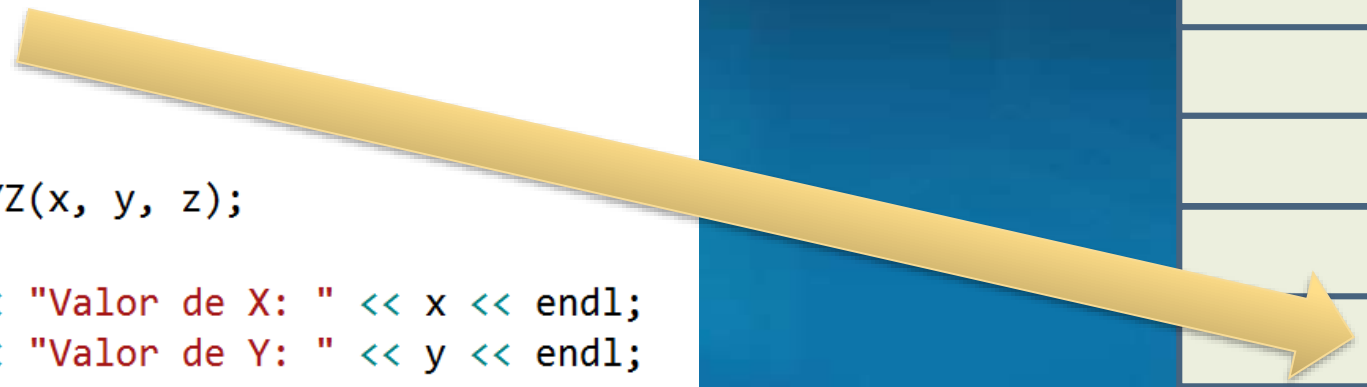


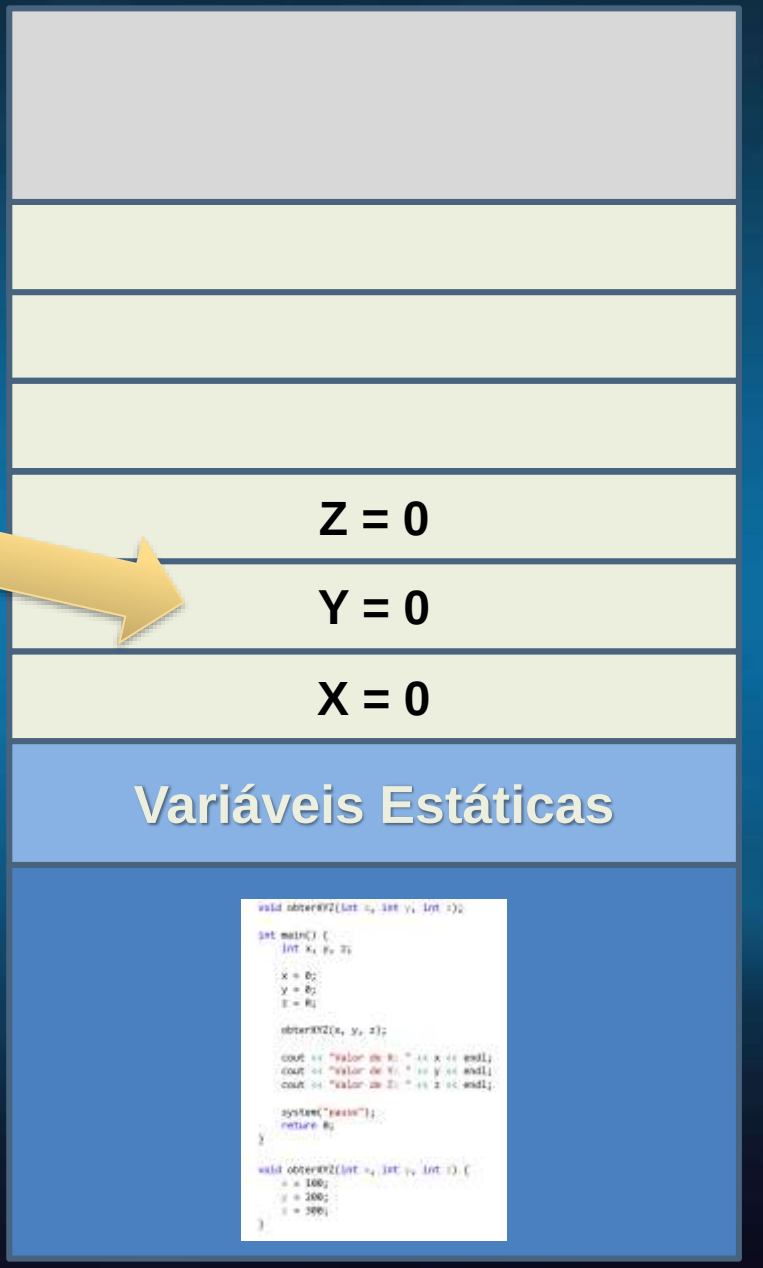


Saída

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



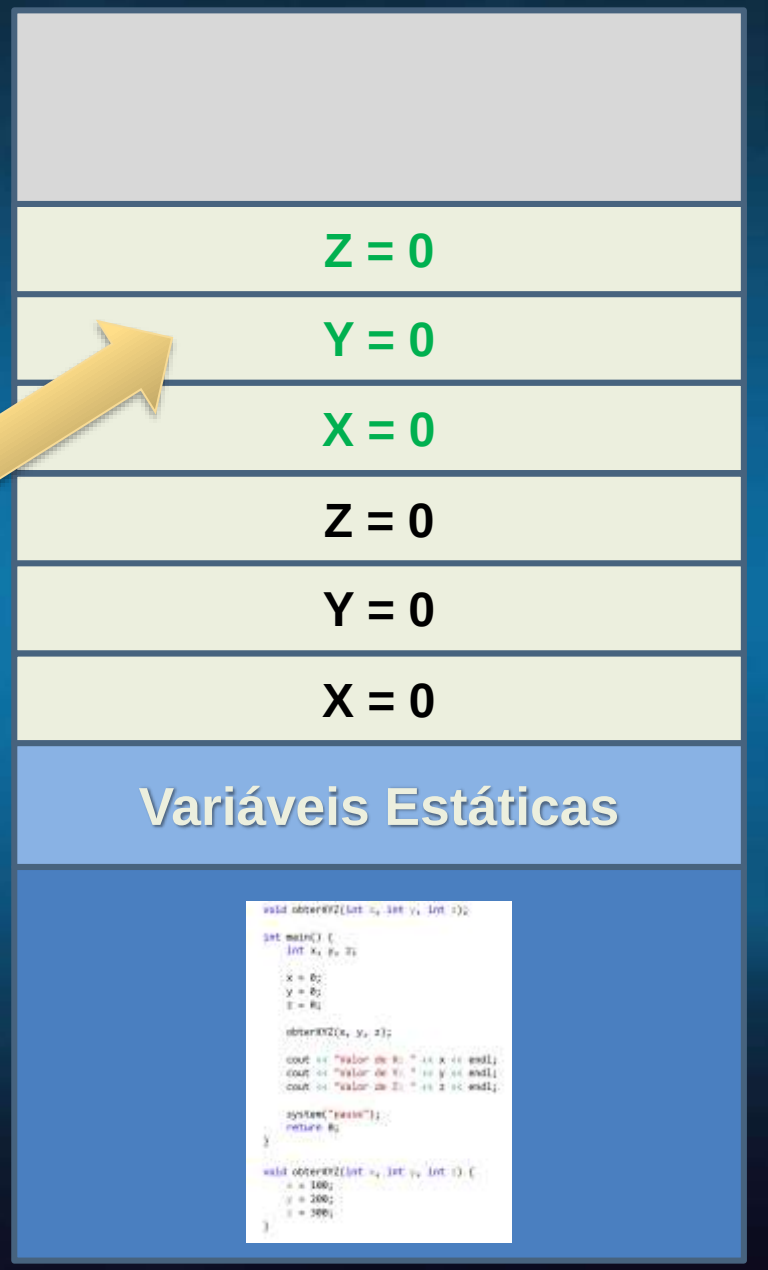


Saída

Problema

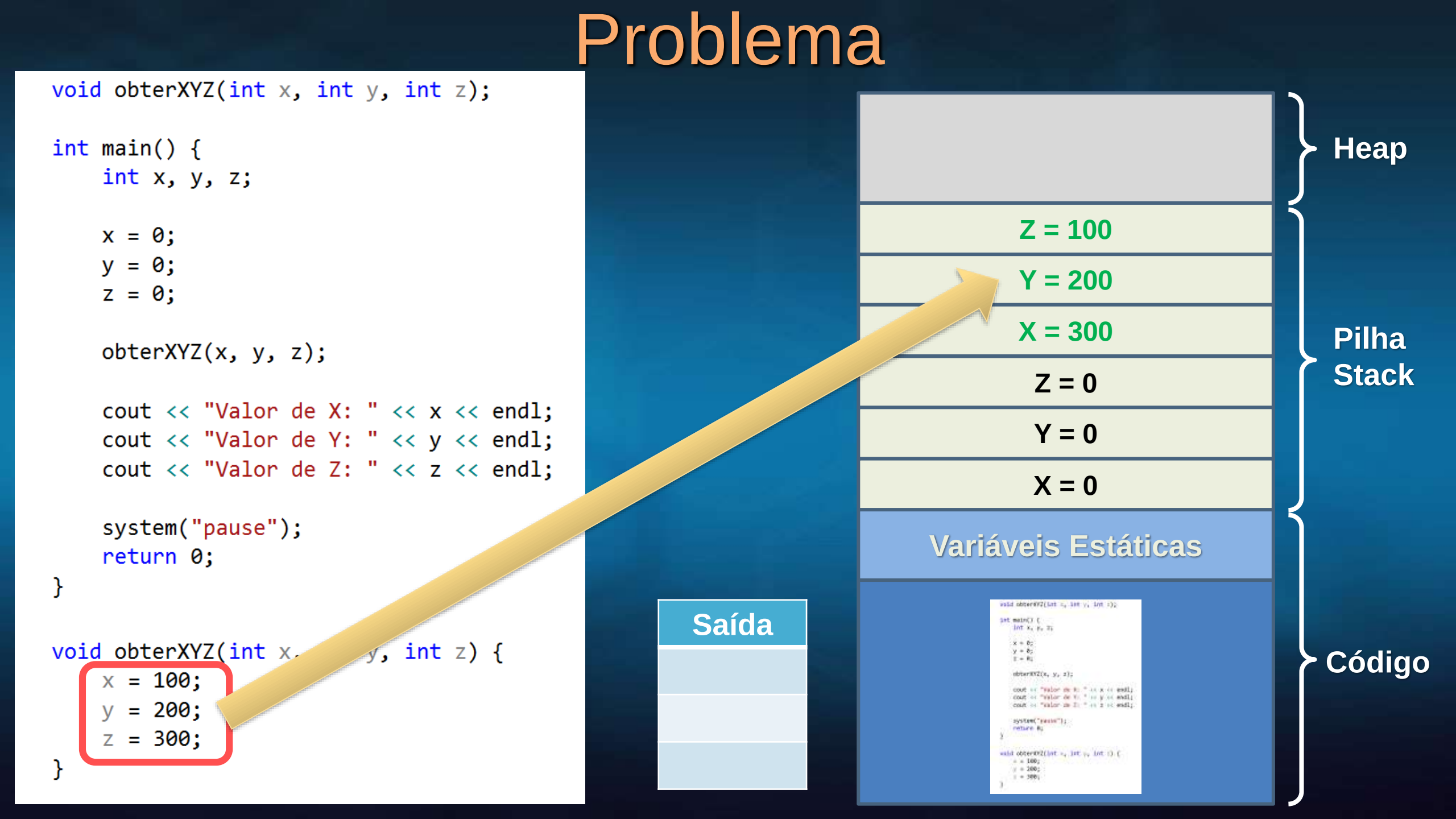
```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída



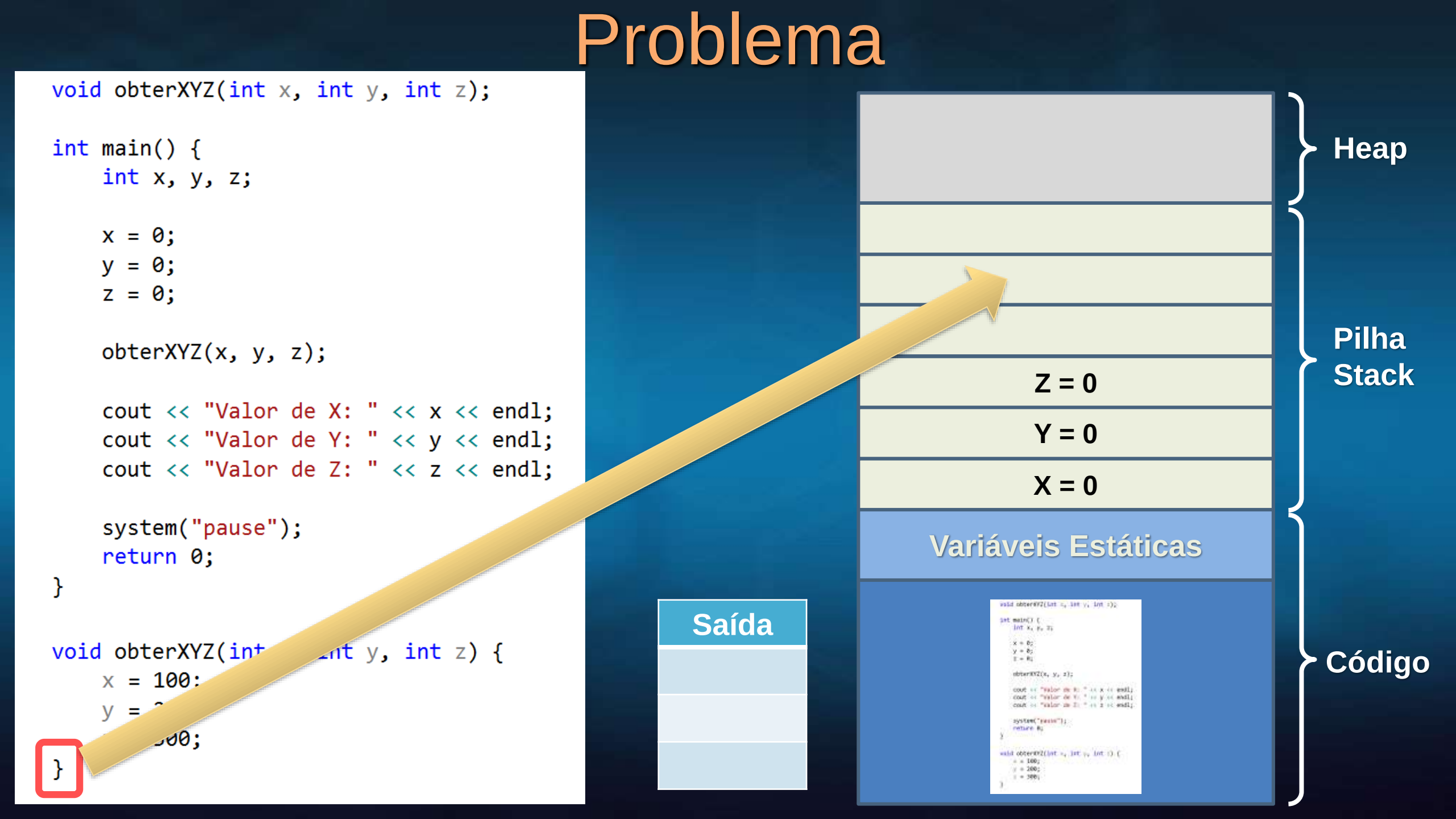
Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x,      y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



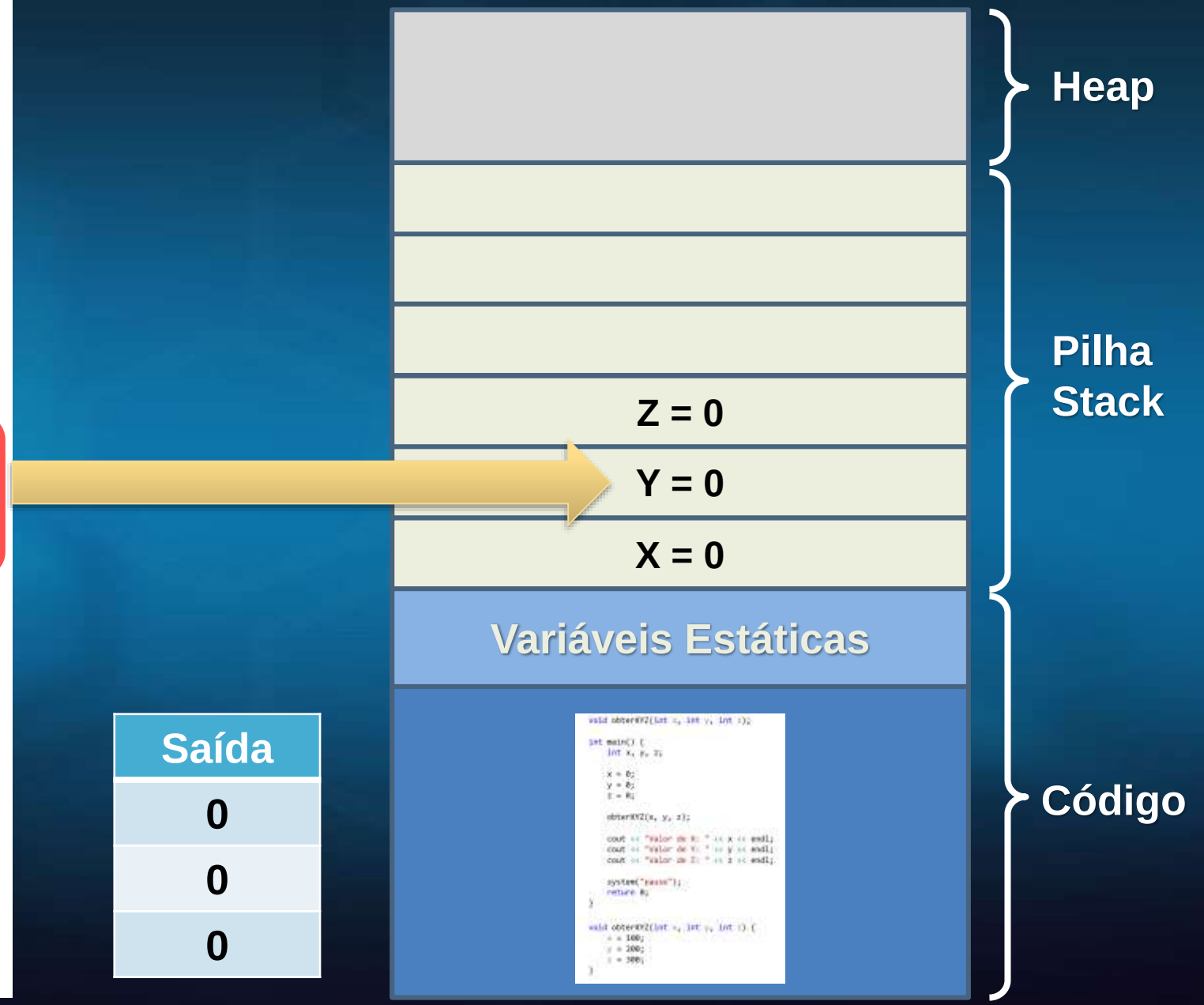
Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



Saída
0
0
0

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída
0
0
0

Zoom na memória

Ponteiros

Problema

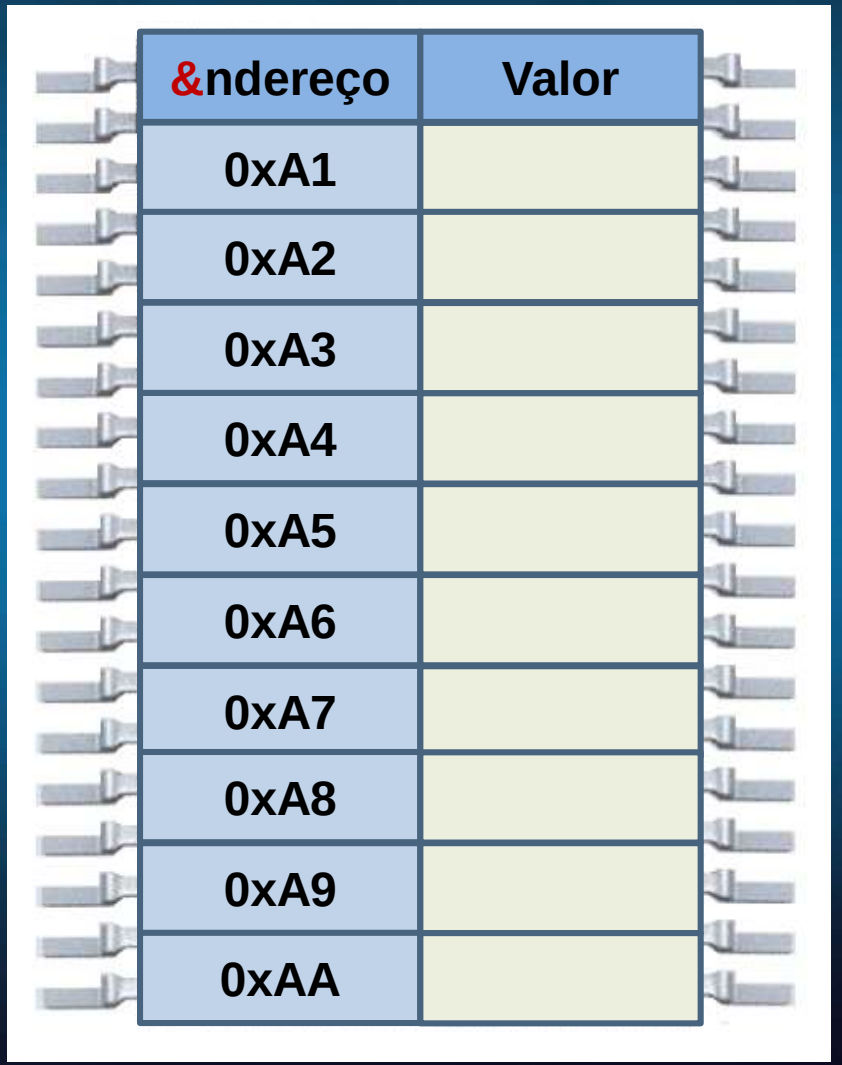
```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída



Problema

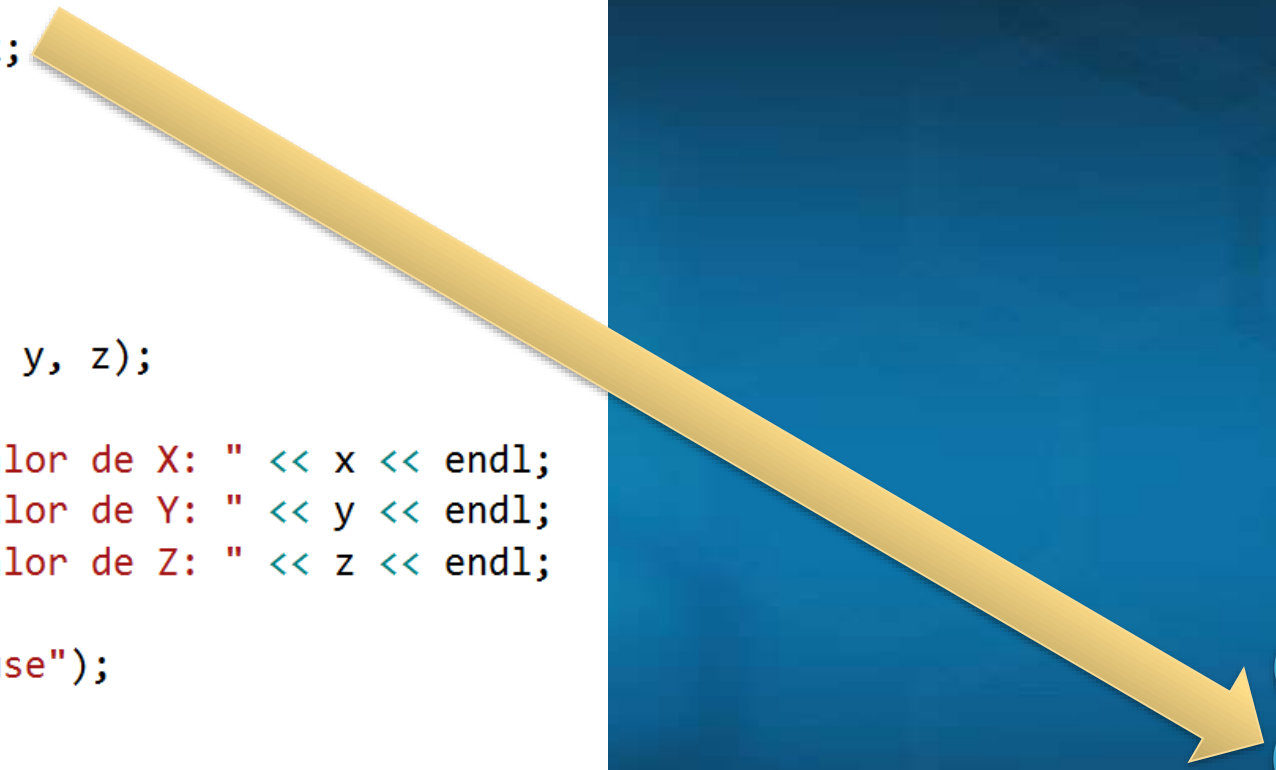
```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



Saída

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

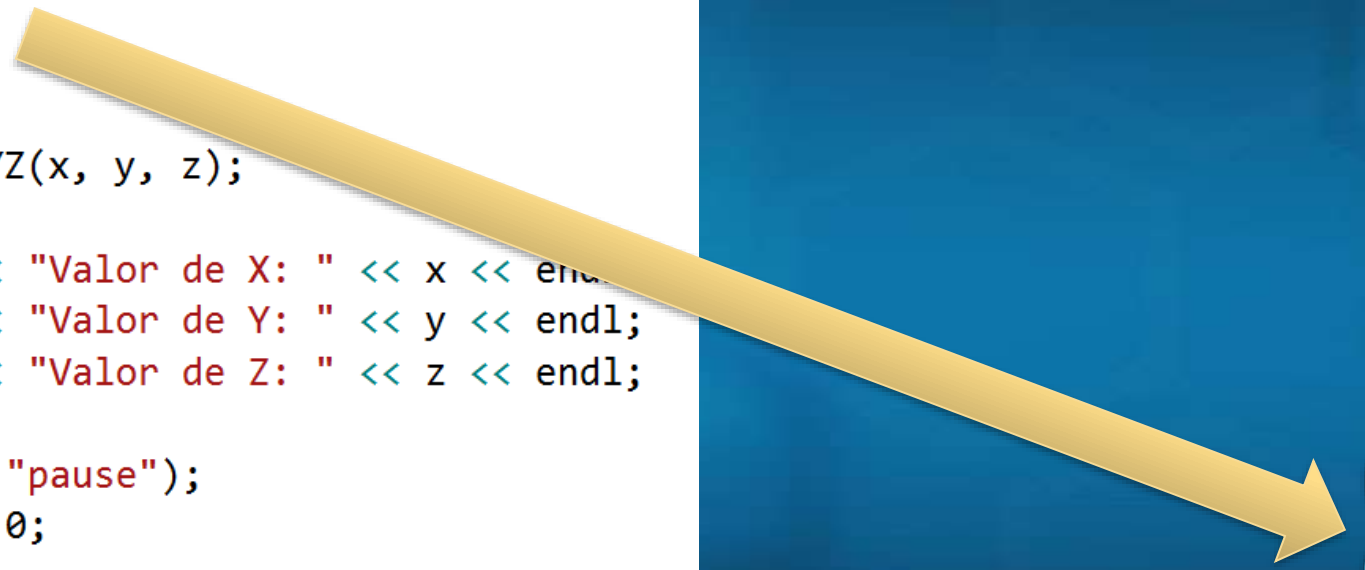
x

y

z

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << end
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



Saída

&ndereço	Valor	
0xA1		
0xA2		
0xA3		
0xA4		
0xA5		
0xA6	0	x
0xA7	0	y
0xA8	0	z
0xA9		
0xAA		

Problema

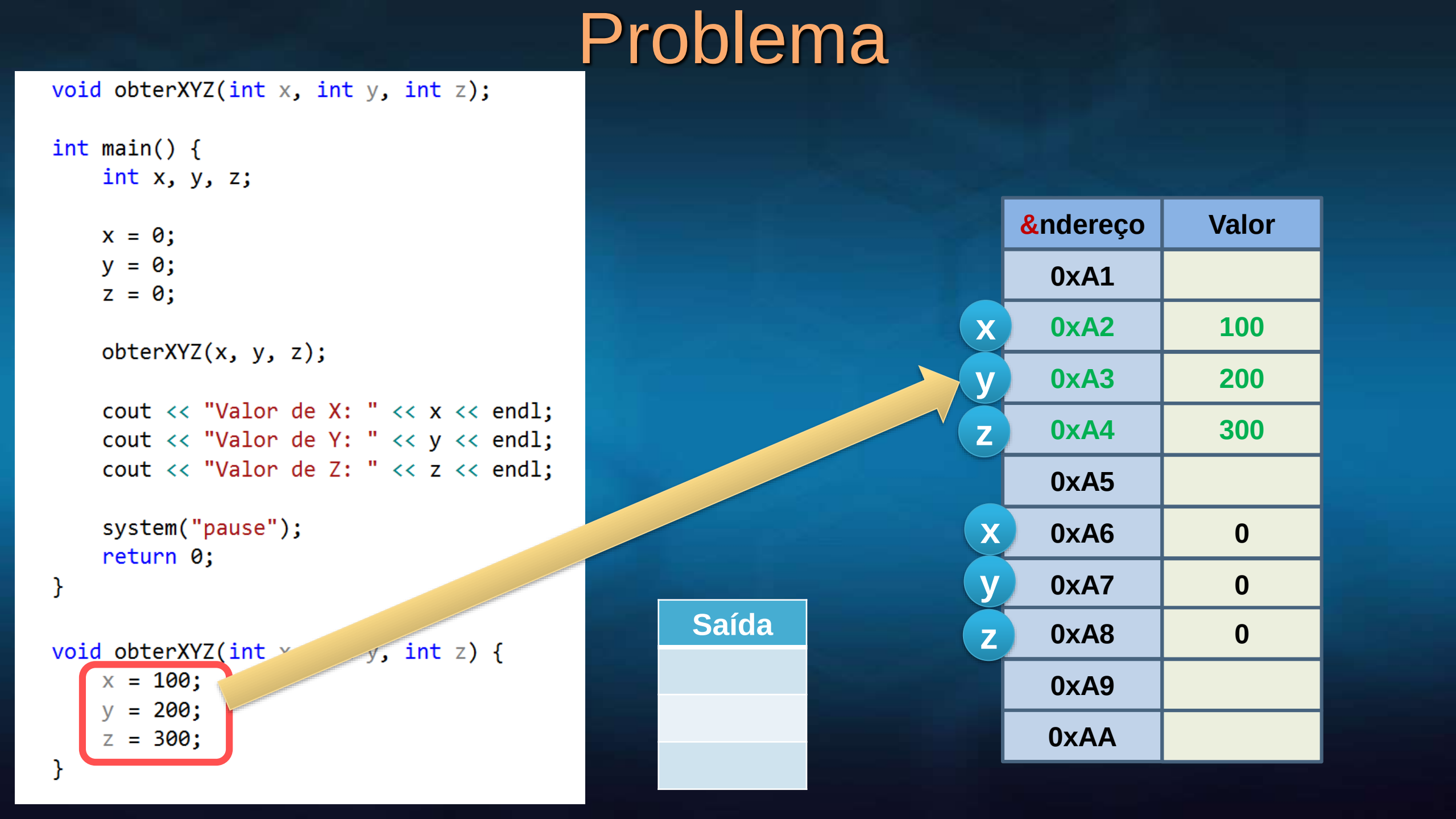
```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída

	&ndereço	Valor
	0xA1	
x	0xA2	0
y	0xA3	0
z	0xA4	0
	0xA5	
x	0xA6	0
y	0xA7	0
z	0xA8	0
	0xA9	
	0xAA	

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

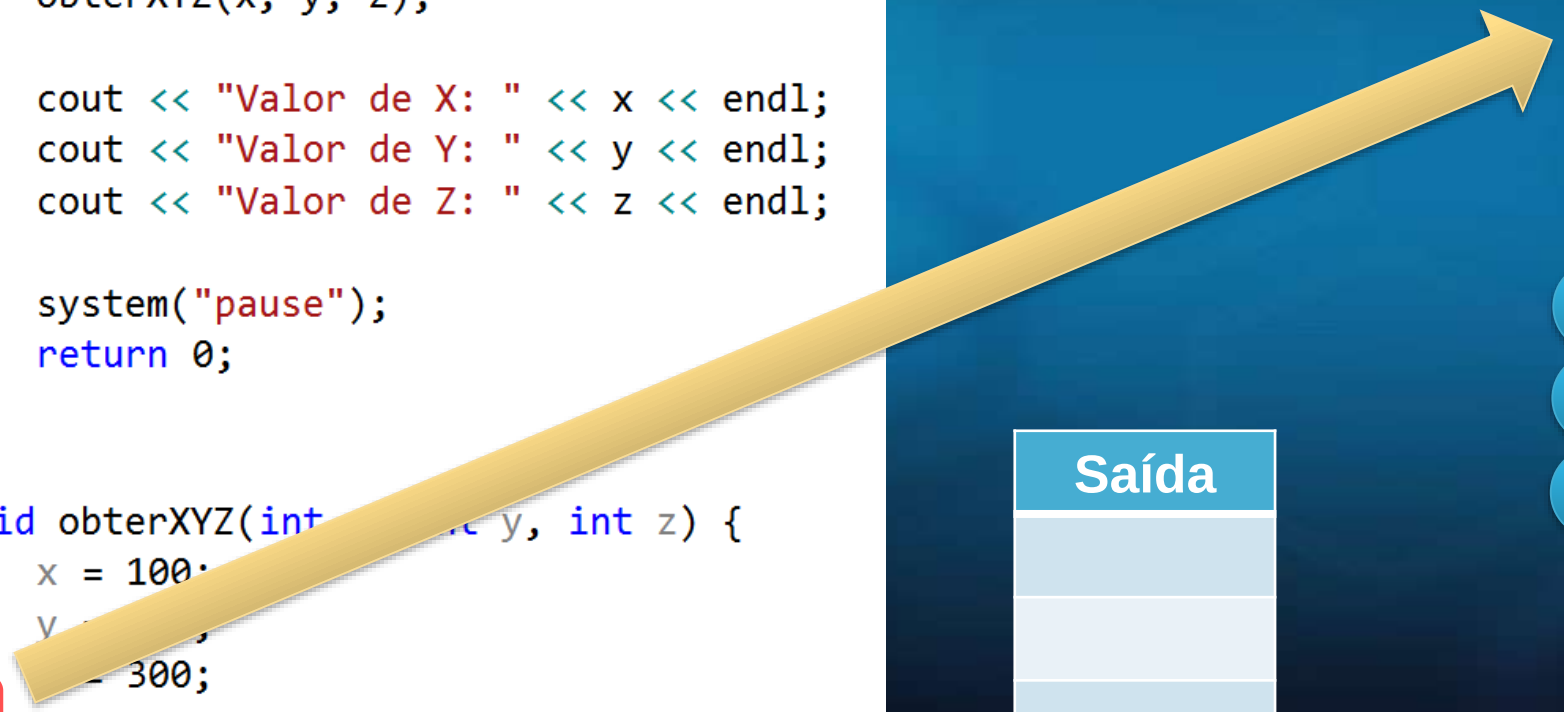


Saída

	&ndereço	Valor
	0xA1	
x	0xA2	100
y	0xA3	200
z	0xA4	300
	0xA5	
x	0xA6	0
y	0xA7	0
z	0xA8	0
	0xA9	
	0xAA	

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```



Saída

&ndereço	Valor	
0xA1		
0xA2		
0xA3		
0xA4		
0xA5		
0xA6	0	x
0xA7	0	y
0xA8	0	z
0xA9		
0xAA		

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída
0
0
0

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
x 0xA6	0
y 0xA7	0
z 0xA8	0
0xA9	
0xAA	

Problema

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Saída
0
0
0

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Endereços de variáveis

Ponteiros

Endereços de variáveis

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;

    x = 100;

    cout << "Valor de X: " << x << endl;


    system("pause");
    return 0;
}
```

Saída
100

x

&ndereço	Valor
0xA1	
0xA2	100
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Endereços de variáveis

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;

    x = 100;

    cout << "Endereço de X: " << &x << endl;
    cout << "Valor de X: " << x << endl;


    system("pause");
    return 0;
}
```

Saída
0xA2
100

x

&ndereço	Valor
0xA1	
0xA2	100
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Exercício

- Crie 3 variáveis, uma do tipo int, outra do tipo float e a terceira do tipo char. Atribua valores para as mesmas e em seguida, exiba os dados e os seus endereços de memória.

Ponteiros

Ponteiros

Ponteiros

&ndereço	Valor
0xA1	
0xA2	100
0xA3	30.50
0xA4	R
0xA5	
0xA6	
0xA7	0xA2
0xA8	
0xA9	
0xAA	

int

float

char

Endereço

Ponteiros

Ponteiros são um tipo especial de variáveis que permitem armazenam endereços de memória ao invés de dados números (como os tipos **int**, **float** e **double**) ou caracteres (como o tipo **char**).

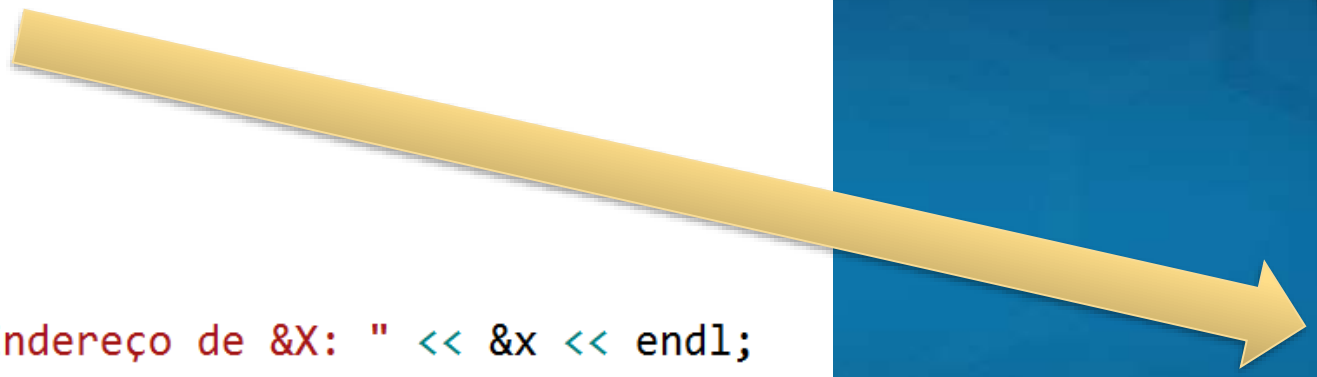
Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl;
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl;
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```



&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

X

pX

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl;
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```

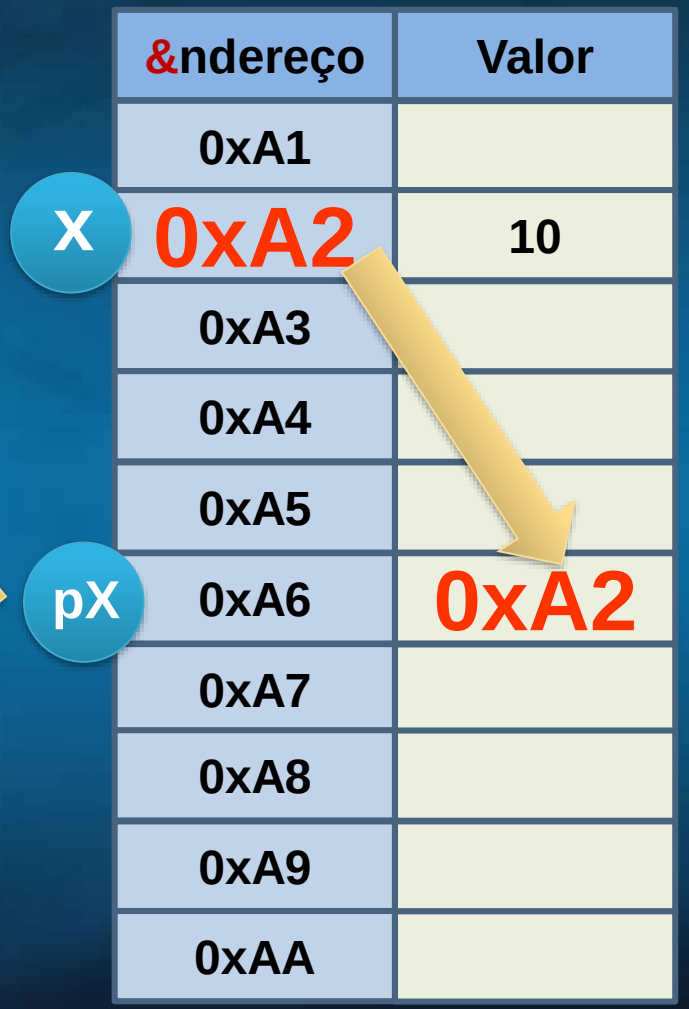
&ndereço	Valor
0xA1	
0xA2	10
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

X

pX

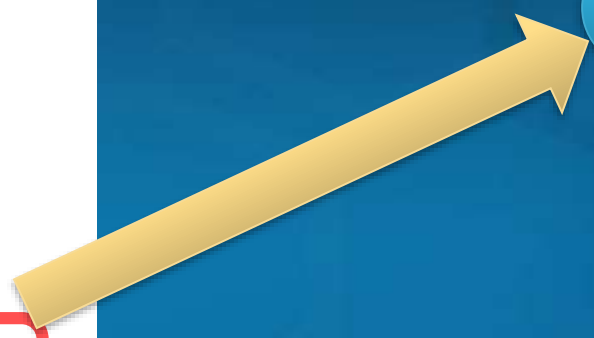
Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl;
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```



Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl;
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```



&ndereço	Valor
0xA1	
0xA2	10
0xA3	
0xA4	
0xA5	
0xA6	0xA2
0xA7	
0xA8	
0xA9	
0xAA	

x

pX

Ponteiros

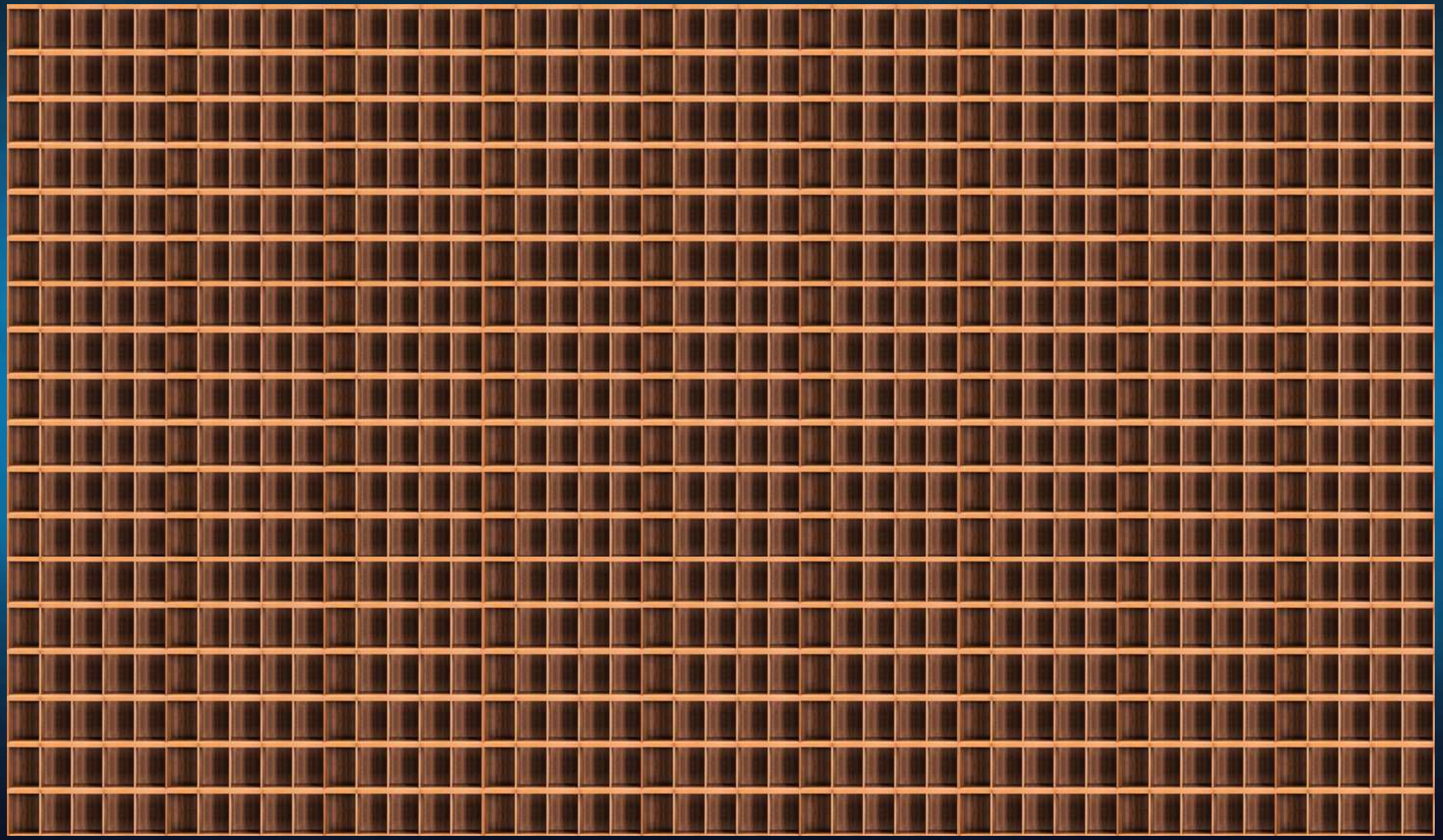
```
int main() {
    setlocale(LC_ALL, "Portuguese");

    int x;
    int *pX;

    x = 10;

    pX = &x;

    cout << "Endereço de &X: " << &x << endl;
    cout << "Valor de X: " << x << endl;

    cout << "Endereço de &pX: " << &pX << endl
    cout << "Conteúdo de pX: " << pX << endl;
    cout << "Valor de *pX: " << *pX << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
	0xA1	
X	0xA2	10
	0xA3	
	0xA4	
	0xA5	
pX	0xA6	0xA2
	0xA7	
	0xA8	
	0xA9	
	0xAA	

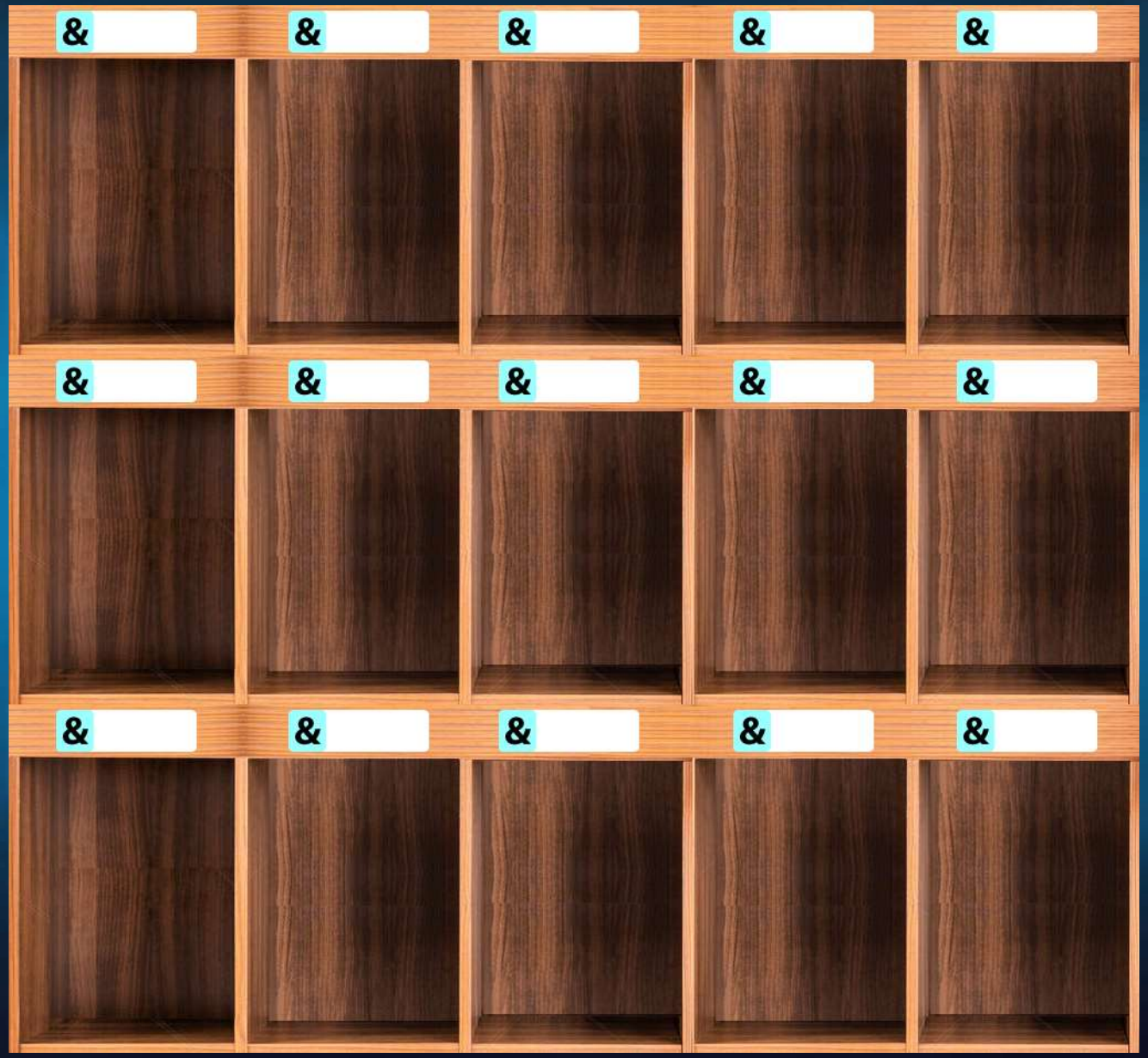
```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

O que será exibido na tela?

Ponteiros



Ponteiros



Ponteiros

Ponteiros

x

?

Ponteiros

Xavier

Ponteiros

Xavier

Ponteiros

Xavier

Ponteiros

Xavier

z

?

& 0xA1

0xA5

Ponteiros

Xavier

Zorro

& 0xA1

0xA5

Ponteiros

Xavier

Zorro

Ponteiros

Ponteiros

=

=

```
Z = X;
```

Ponteiros

Xavier

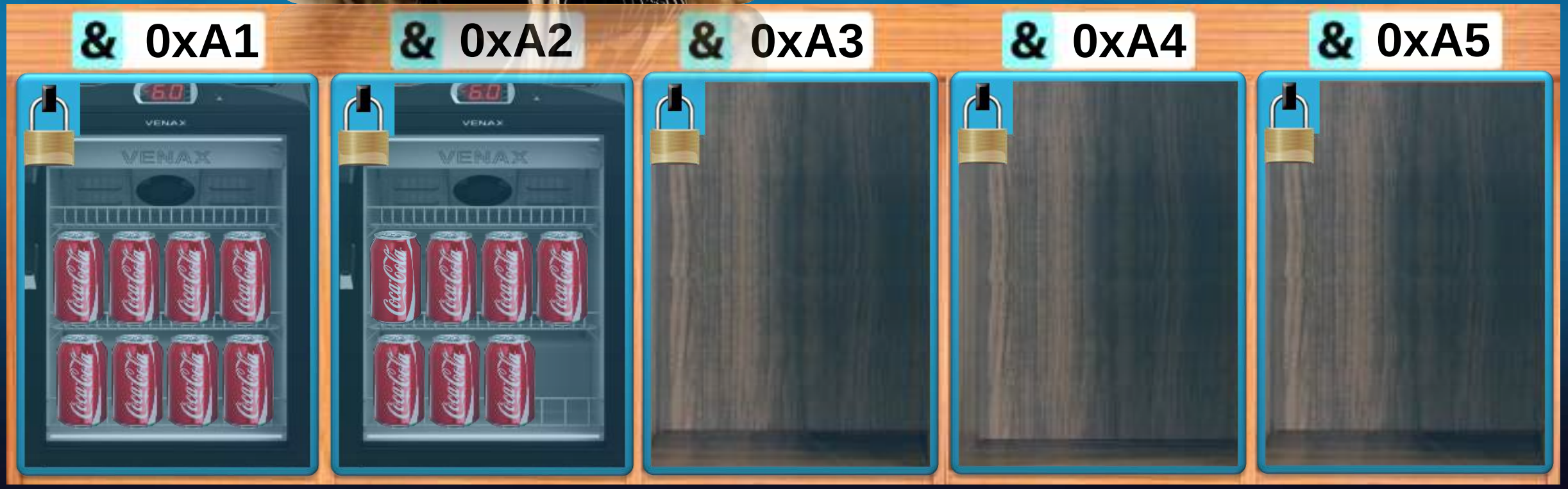
Zorro

Ponteiros

Xavier

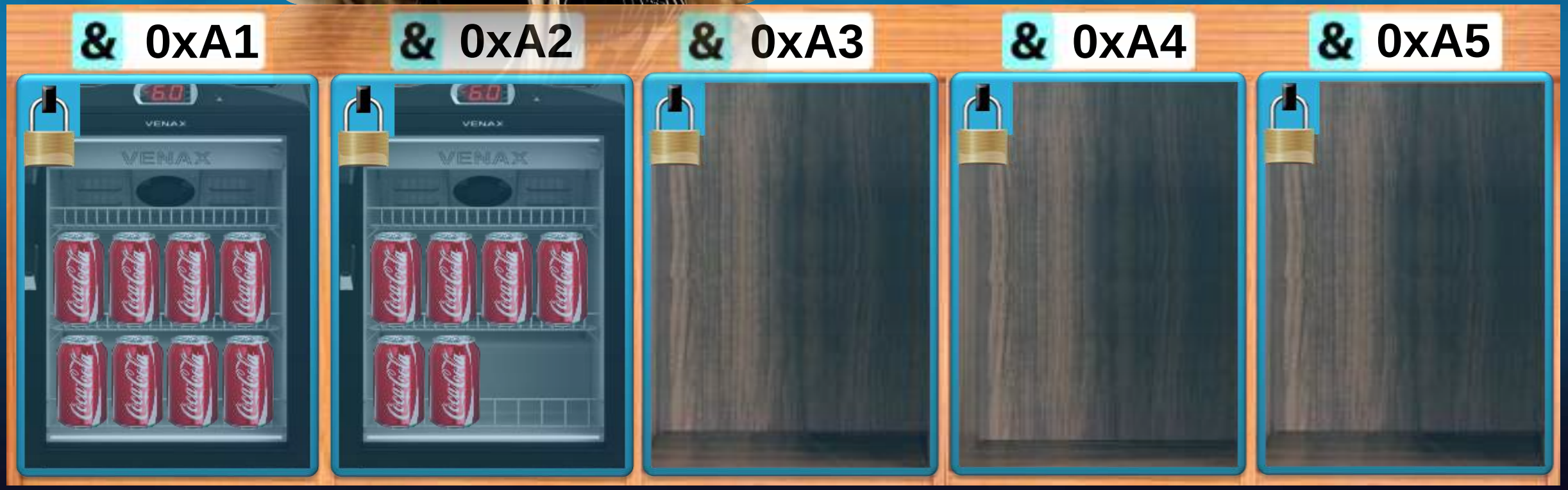
Ponteiros

Xavier



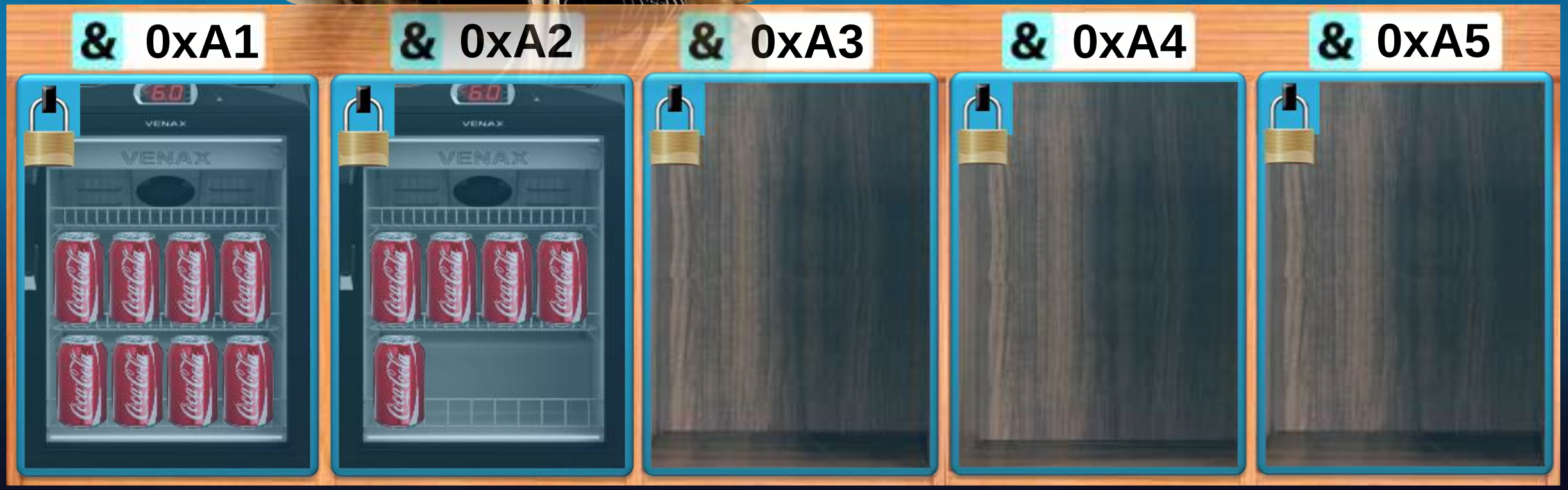
Ponteiros

Xavier



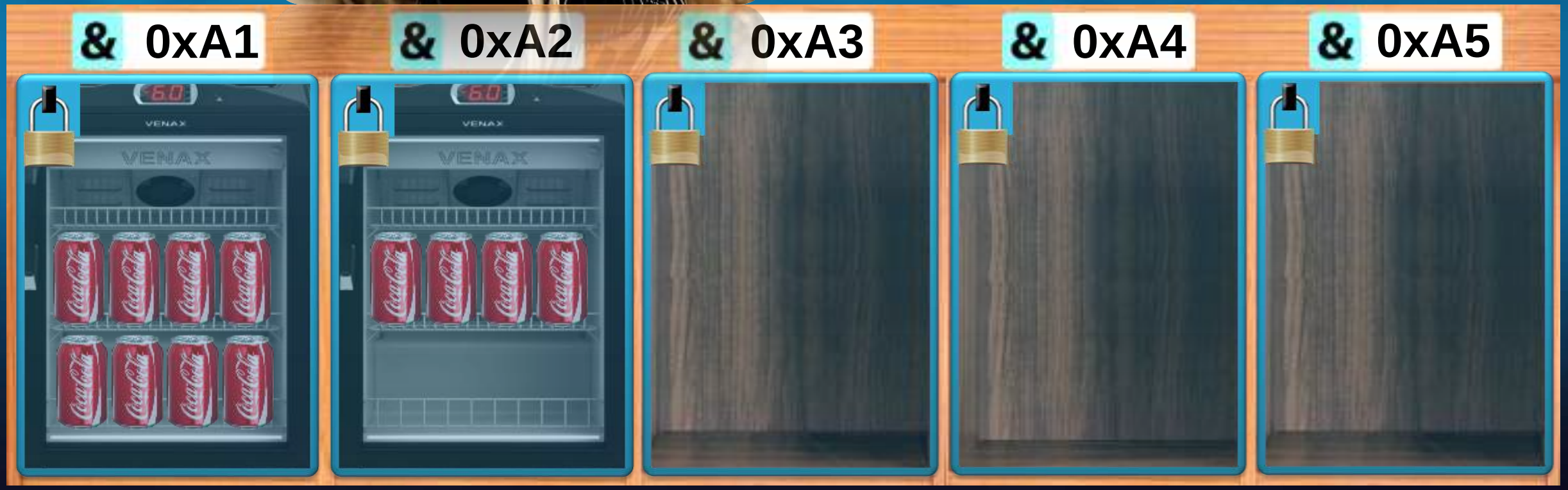
Ponteiros

Xavier



Ponteiros

Xavier



Ponteiros

Xavier

Zorro

Ponteiros

Xavier

PH

?

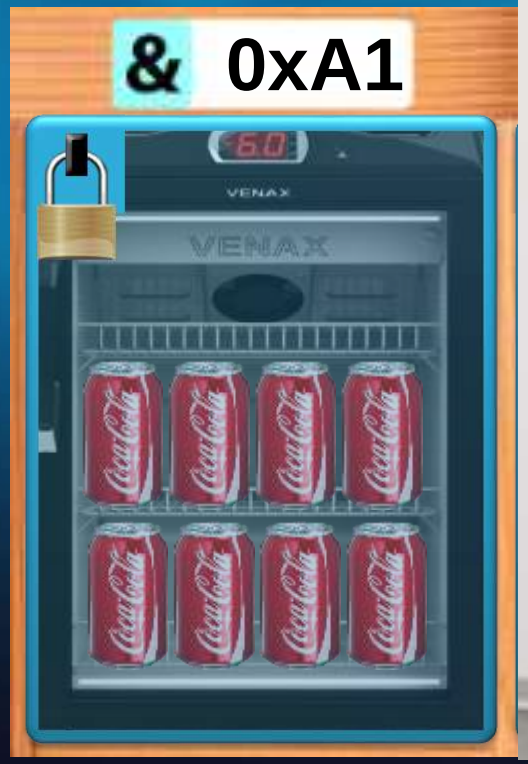
& 0xA1

0xA5

Ponteiros

Xavier

Paulo **H**enrique

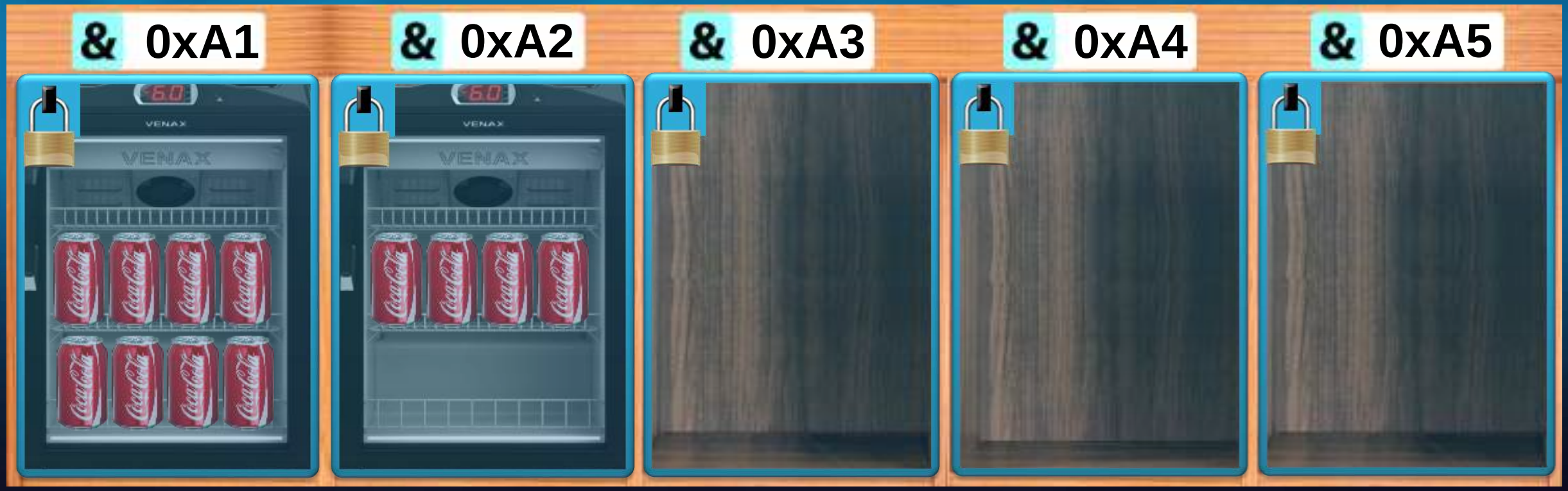


Ponteiros

Xavier

Zorro

Paulo **H**enrique



Ponteiros

Ponteiros

Ponteiros

=

PH = &X;

Ponteiros

Xavier

Zorro

Paulo **H**enrique

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Paulo **H**enrique

Ponteiros

Xavier

Zorro

Ponteiros

Xavier

Zorro

Paulo **H**enrique

Ponteiros

Zorro

Paulo Henrique

Ponteiros

Xavier

Zorro

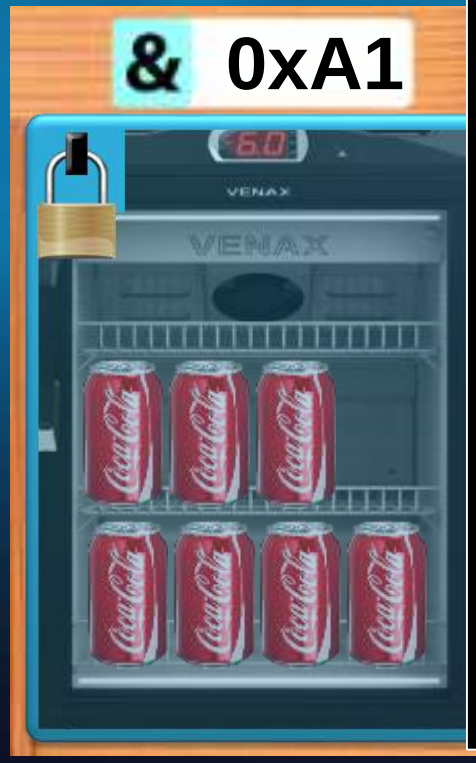
Paulo **H**enrique

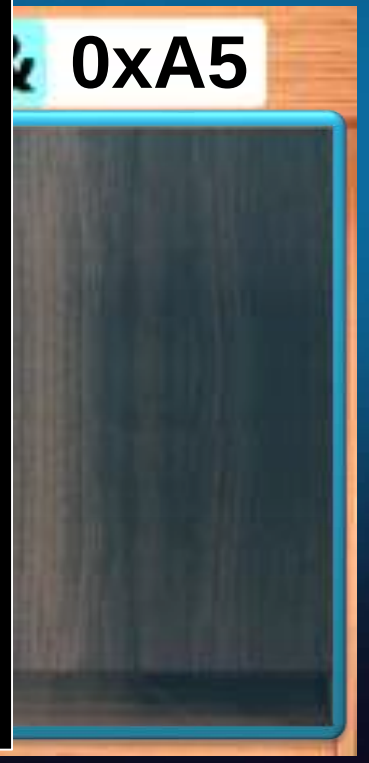
Ponteiros

Xavier

PC

?

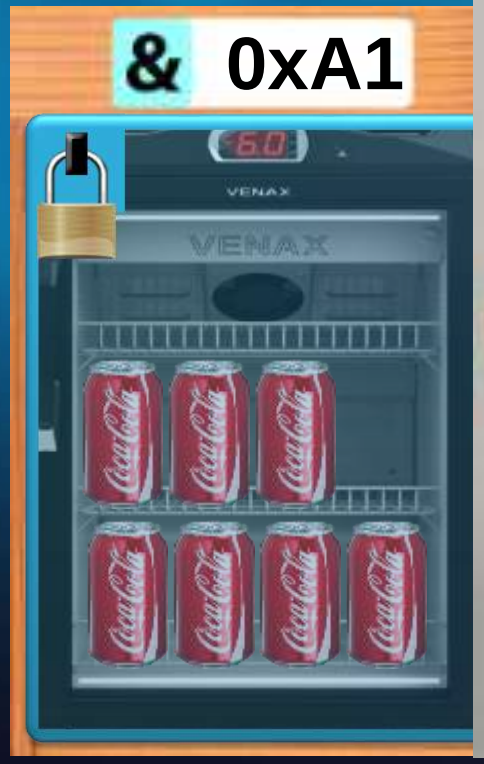


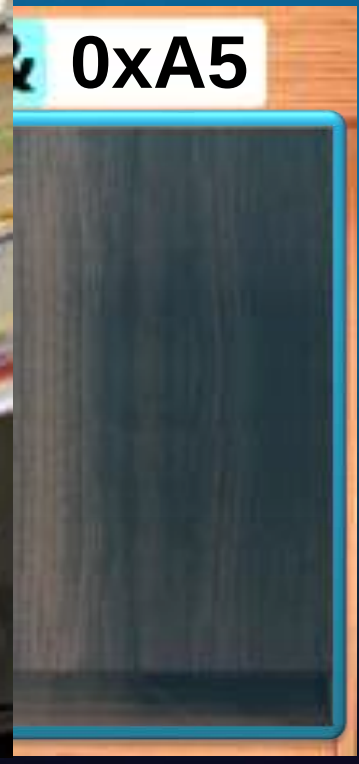


Ponteiros

Xavier

Paula **C**ristina





Ponteiros

Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina

Ponteiros

Ponteiros

=

Ponteiros

Ponteiros

PC = &PH;

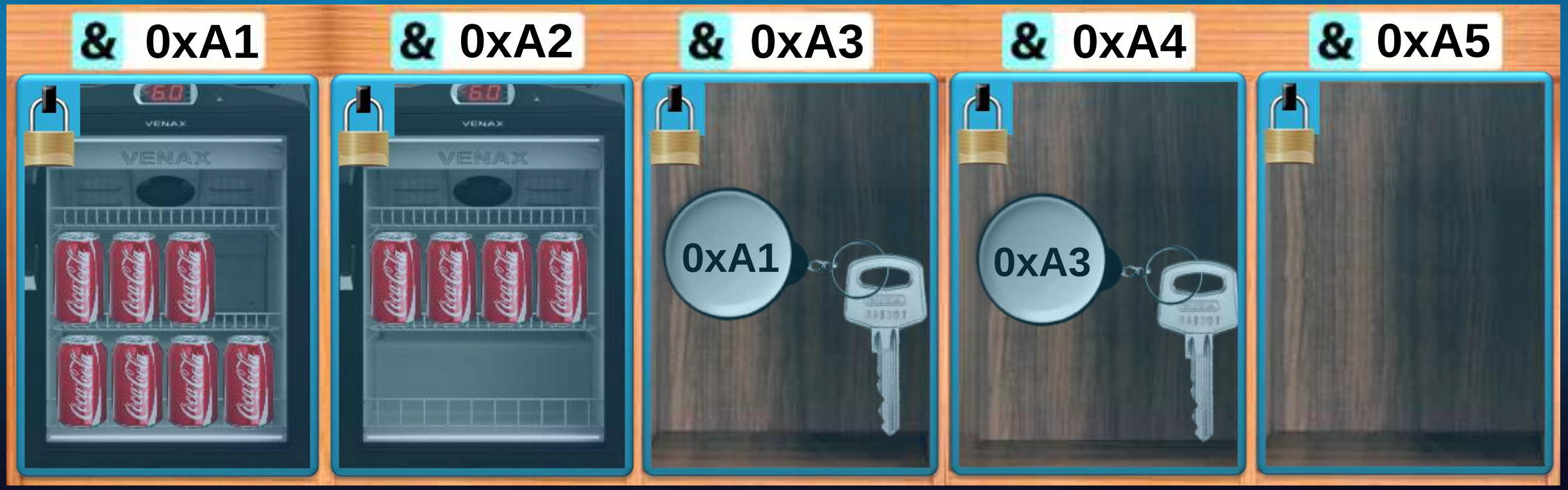
Ponteiros

Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina

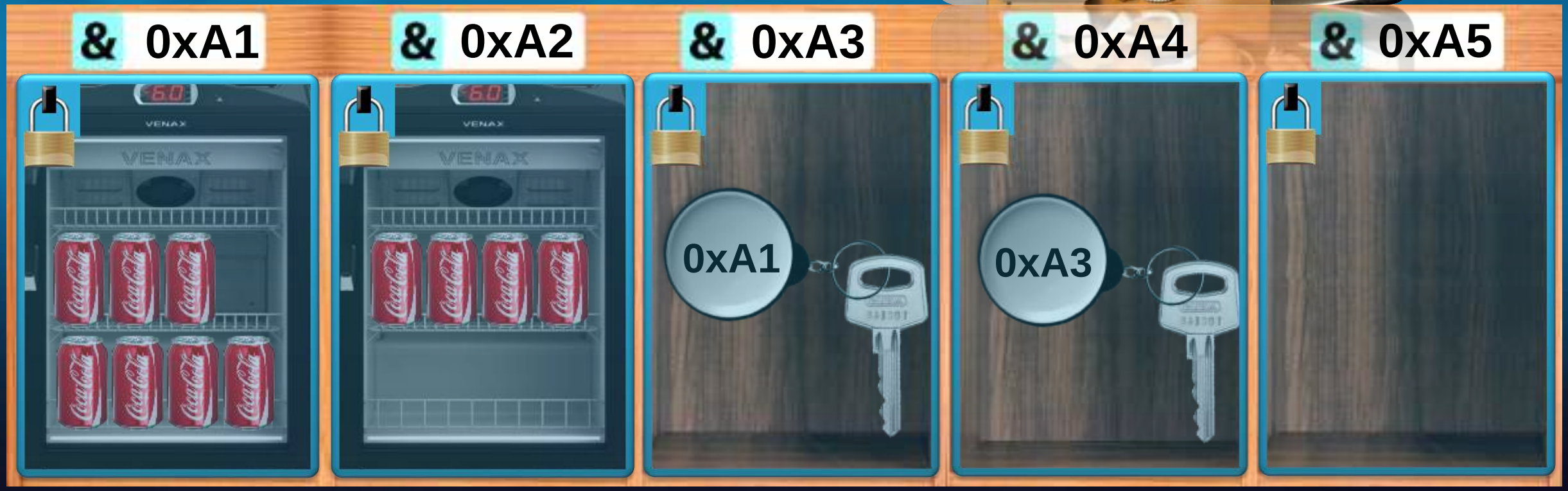


Ponteiros

Xavier

Zorro

Paulo **H**enrique



Ponteiros

Xavier

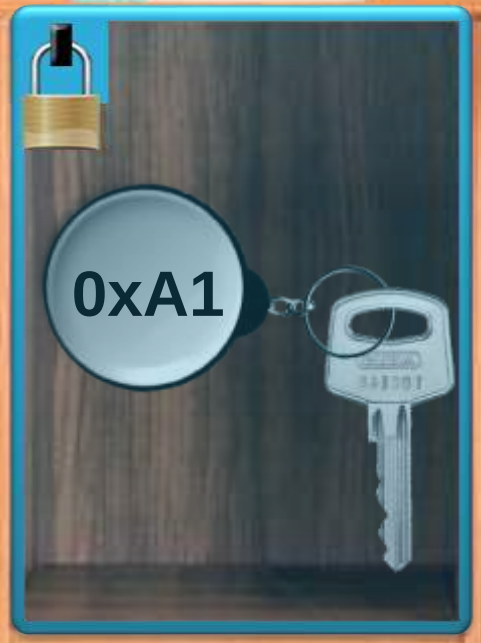
Zorro

Paulo **H**enrique

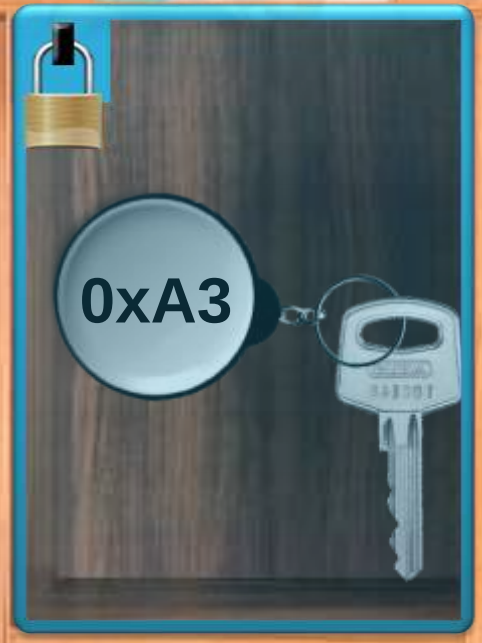
& 0xA1

& 0xA2

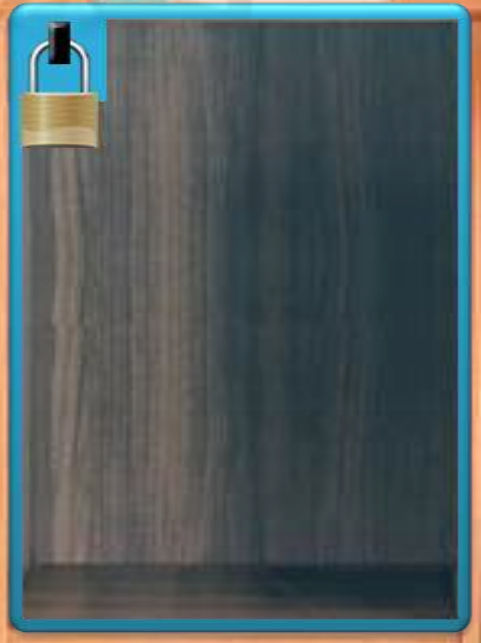
& 0xA3



& 0xA4



& 0xA5

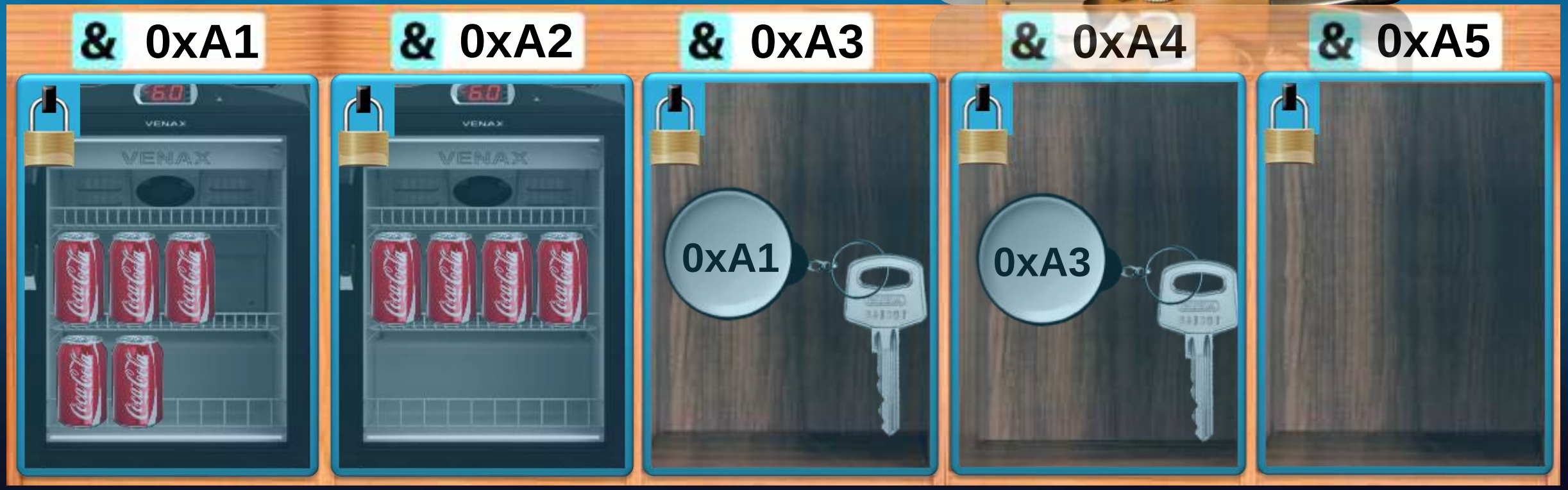


Ponteiros

Xavier

Zorro

Paulo **H**enrique

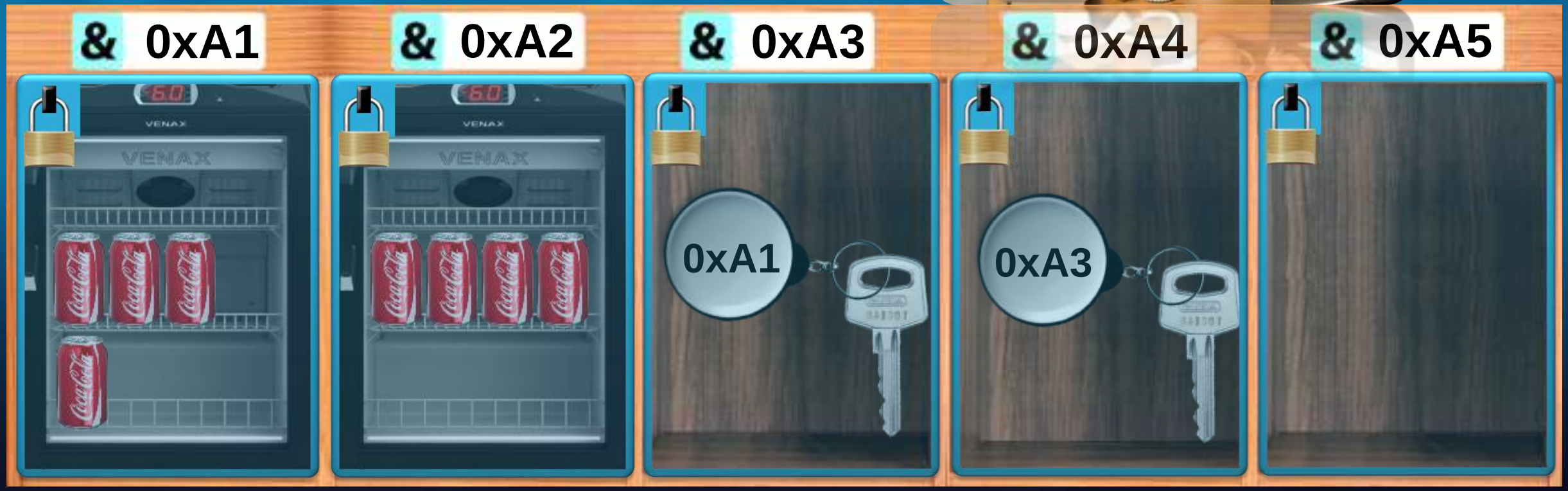


Ponteiros

Xavier

Zorro

Paulo **H**enrique



Ponteiros

Xavier

Zorro

Paulo **H**enrique

& 0xA1

& 0xA2

& 0xA3

& 0xA4

& 0xA5

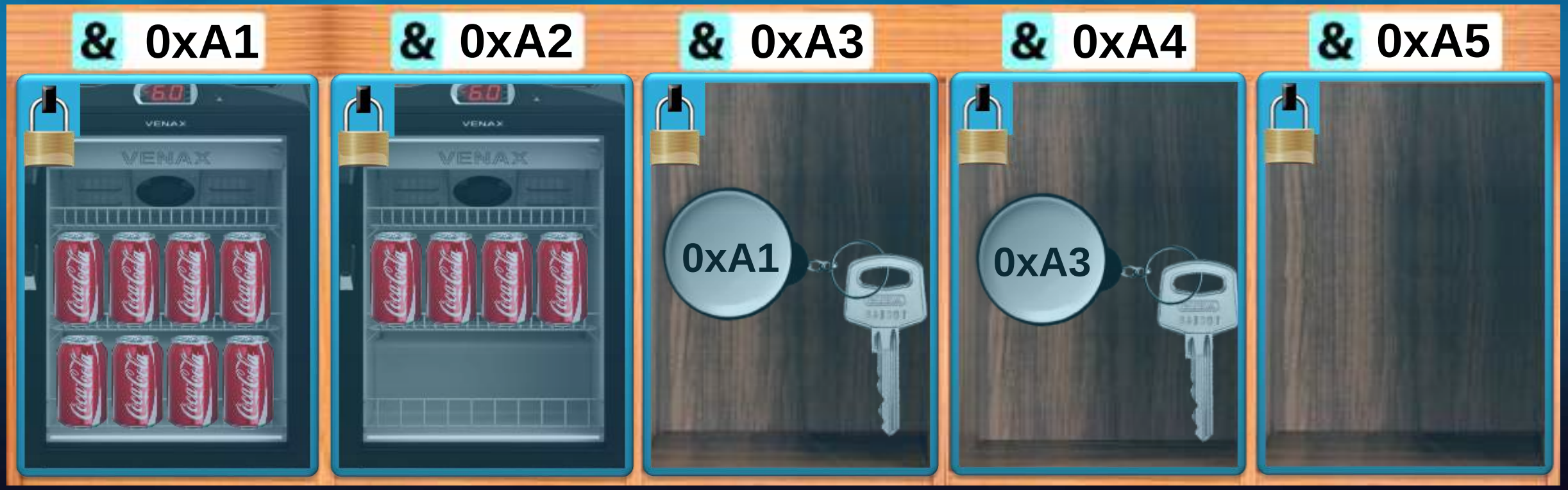
Ponteiros

Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina



Ponteiros

Xavier

PP

?

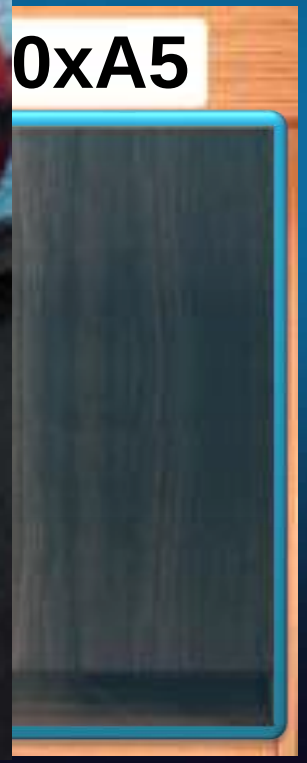
& 0xA1

0xA5

Ponteiros

Xavier

Peter **P**arker



Ponteiros

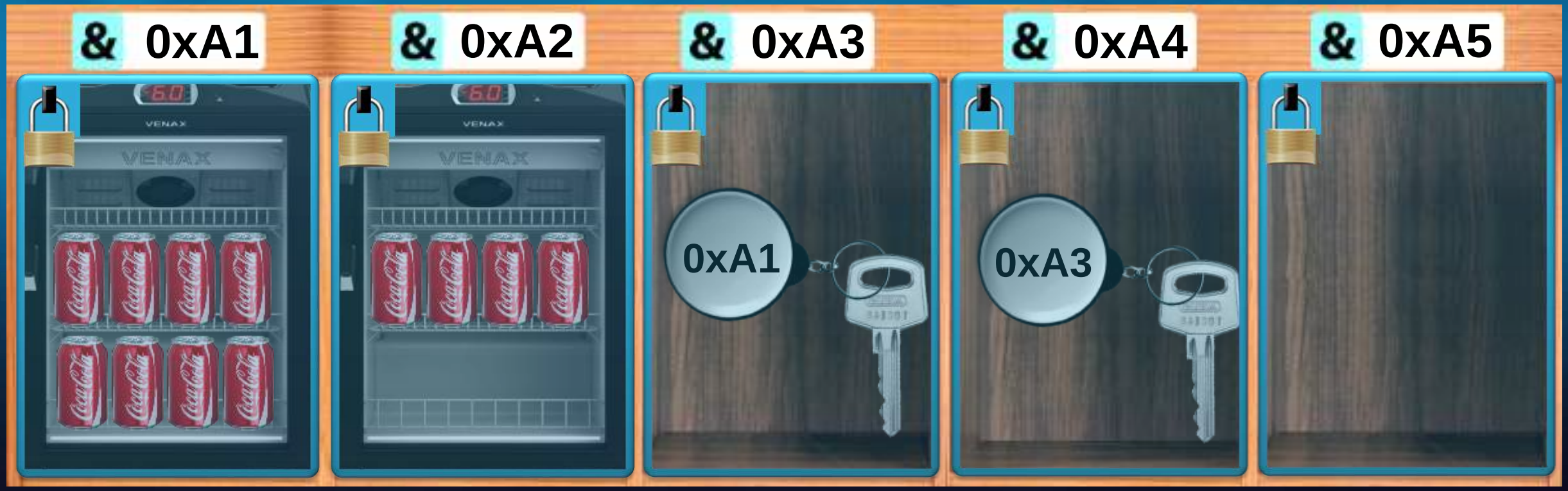
Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina

Peter **P**arker



Ponteiros

PP = &PC;

Ponteiros

Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina

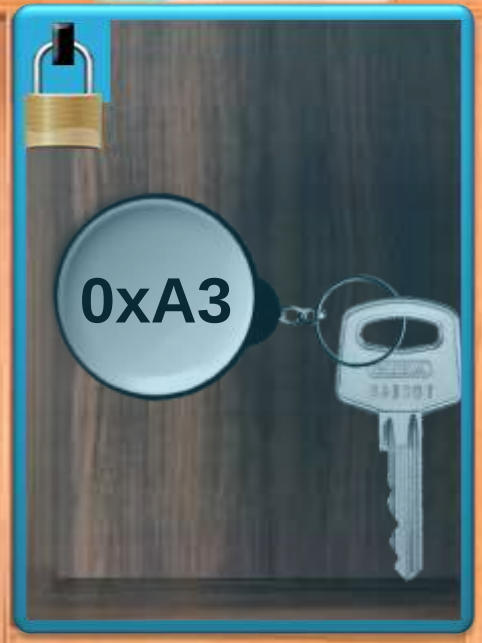
Peter **P**arker

& 0xA1

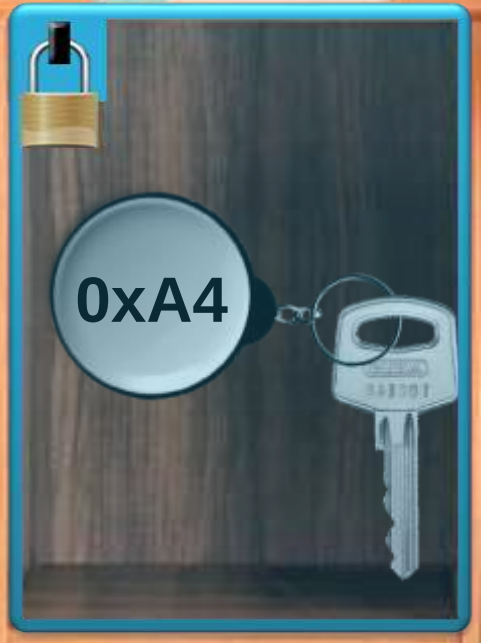
& 0xA2

& 0xA3

& 0xA4



& 0xA5



Ponteiros

Xavier

Zorro

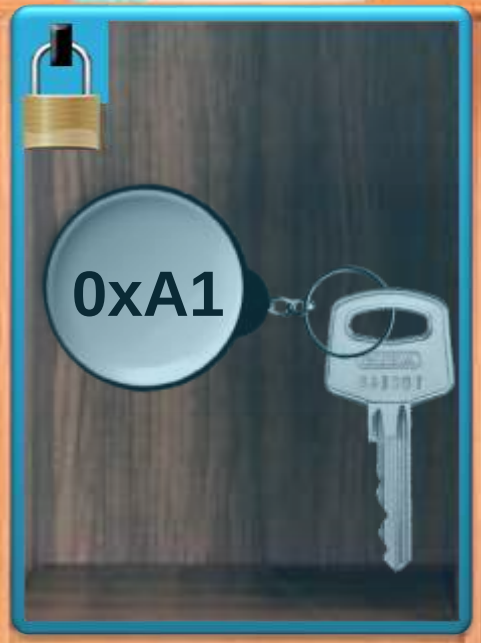
Paulo **H**enrique

Paula

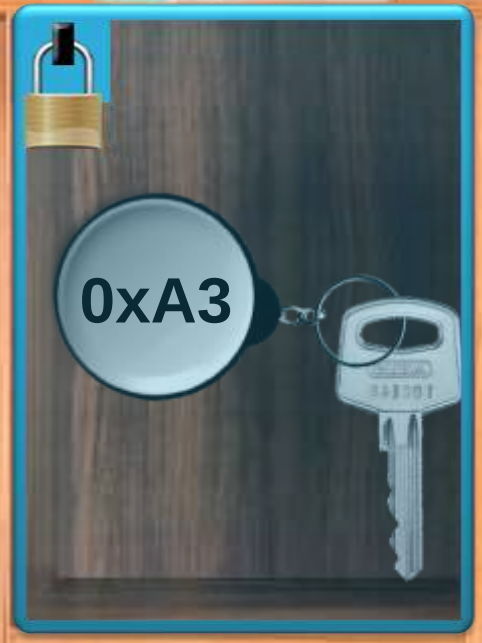
& 0xA1

& 0xA2

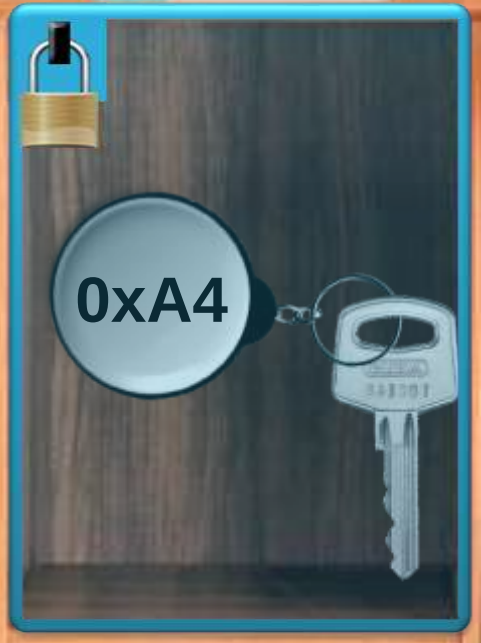
& 0xA3



& 0xA4



& 0xA5



Ponteiros

Xavier

Zorro

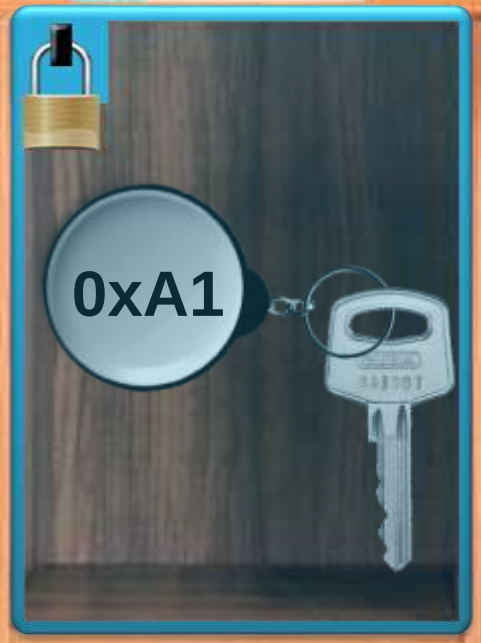
Paulo **H**enrique

Paula

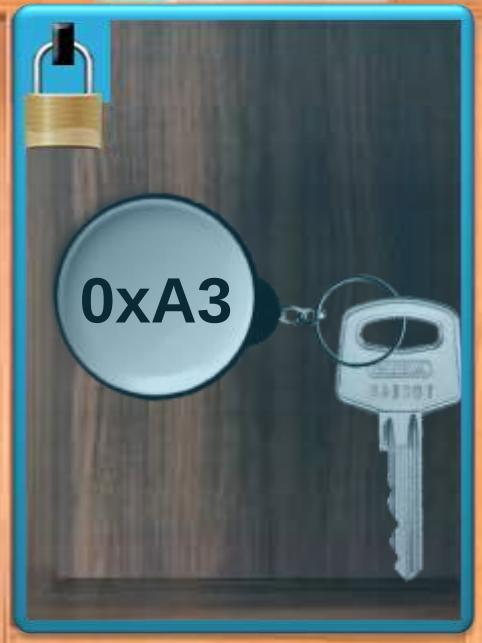
& 0xA1

& 0xA2

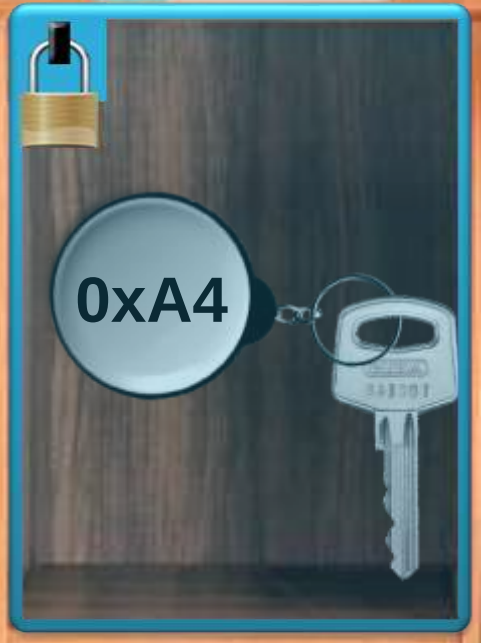
& 0xA3



& 0xA4



& 0xA5



Ponteiros

Xavier

Zorro

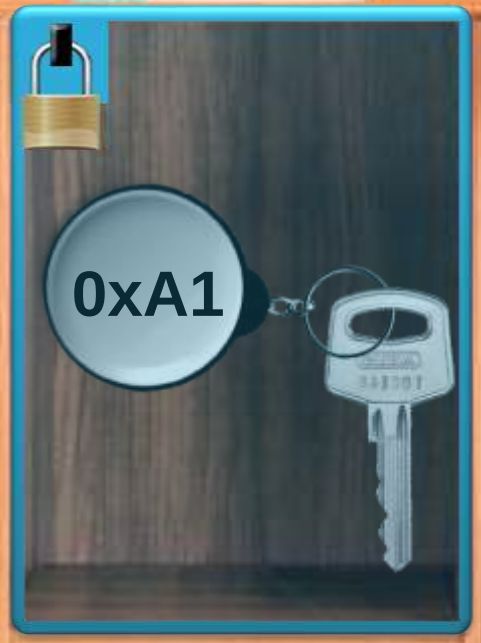
Paulo **H**enrique

Paula

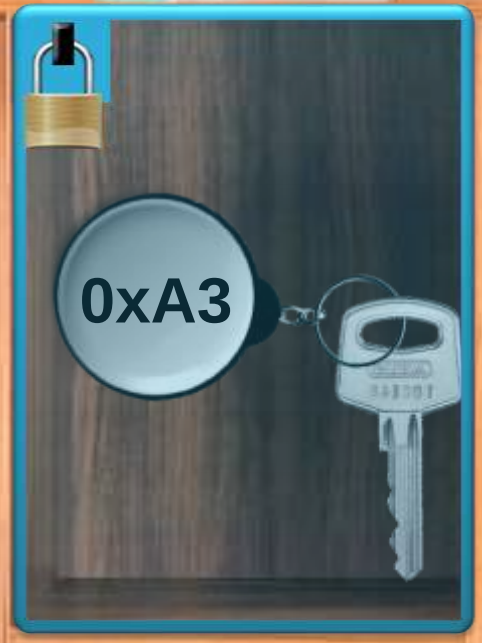
& 0xA1

& 0xA2

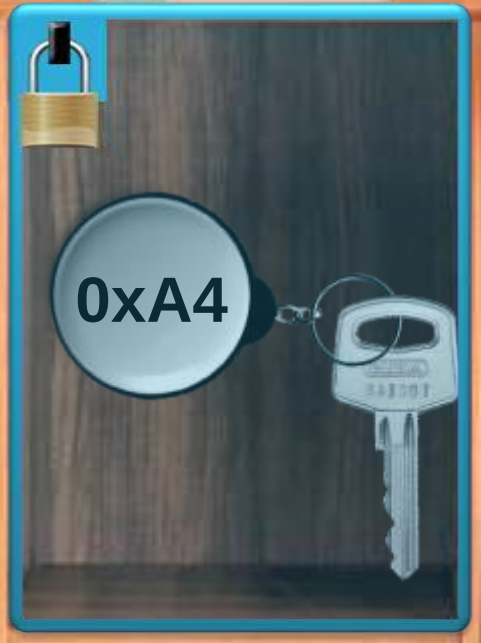
& 0xA3



& 0xA4



& 0xA5



Ponteiros

Xavier

Zorro

Paulo **H**enrique

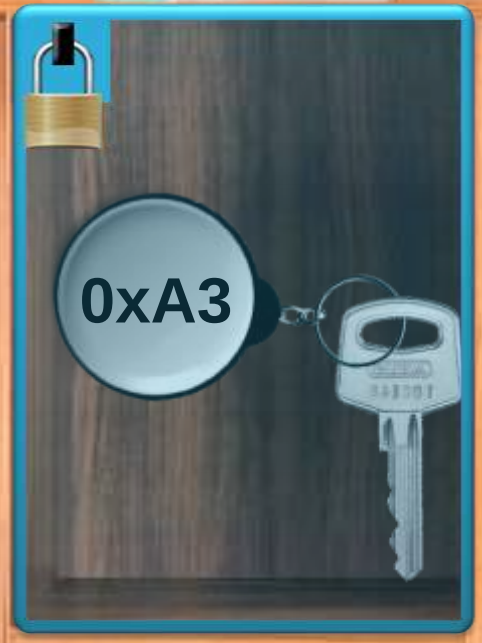
Paula

& 0xA1

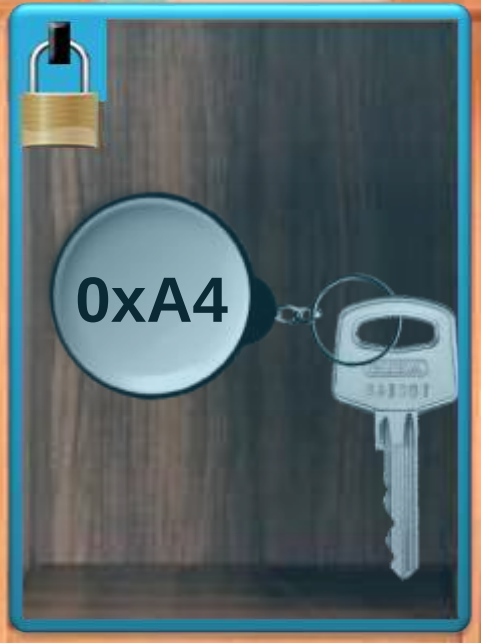
& 0xA2

& 0xA3

& 0xA4



& 0xA5



Ponteiros

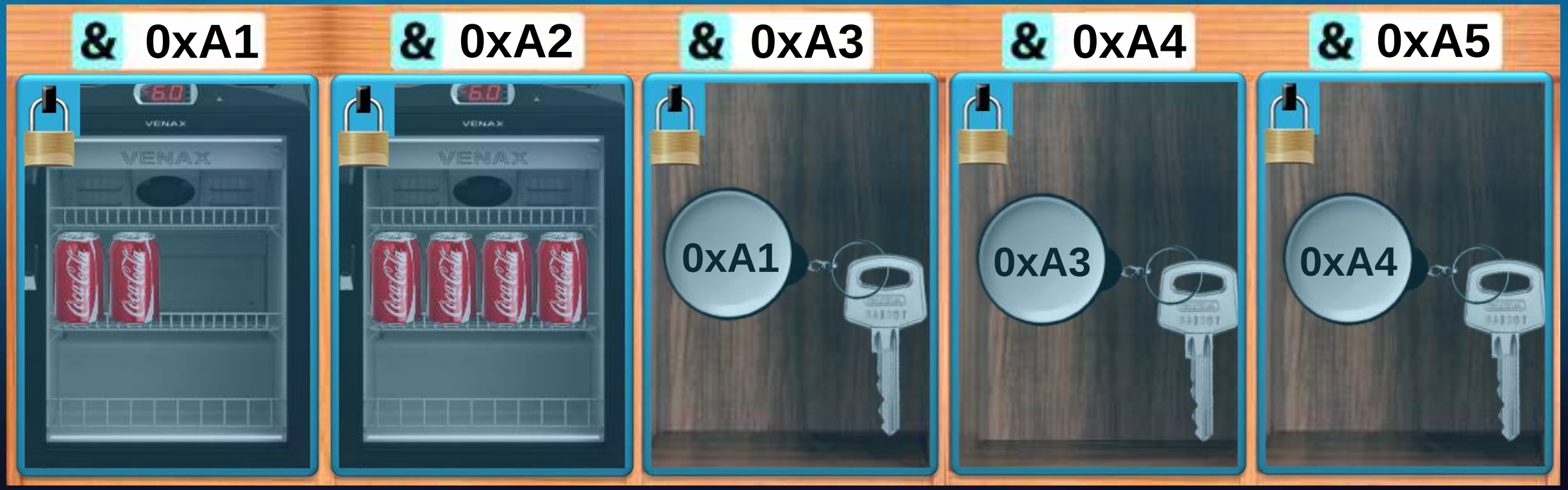
Xavier

Zorro

Paulo **H**enrique

Paula **C**ristina

Peter **P**arker



Ponteiros

Passos

...

...

...

Ponteiros

Xavier

1º Passo
(OBRIGATÓRIO)
Declarar a variável


```
int x;
```

Ponteiros

Xavier

(opcional)
Atribuir valor a variável


```
int x;
x = 8;
```

Ponteiros

Xavier

(opcional)
Exibir dados
da variável


```
int x;
x = 8;
cout << x;
```

Ponteiros

Xavier

(opcional)
Exibir dados da variável

```
int x;
x = 8;
cout << &x;
```

MUITO RARAMENTE

Ponteiros

Xavier

Passos

...

...

...

Ponteiros

Xavier

Paulo Henrique

1º Passo
(OBRIGATÓRIO)
Declarar o ponteiro

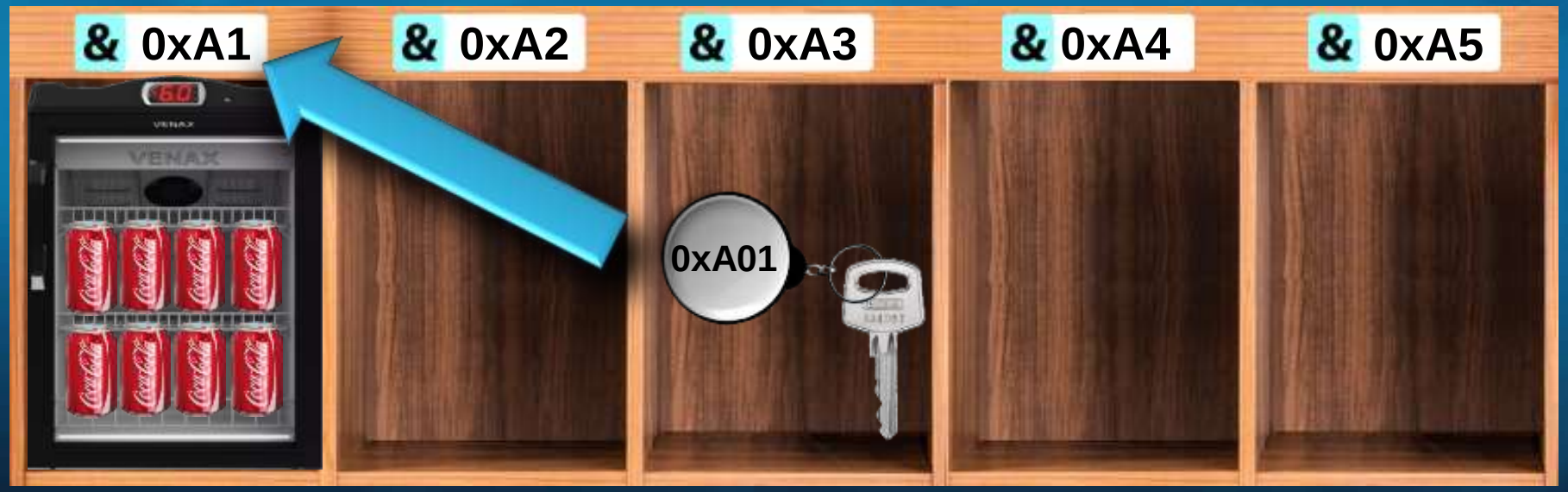

```
int *ph;
```

Ponteiros

Xavier

Paulo **H**enrique

2º Passo
(OBRIGATÓRIO)
Apontar para uma variável




```
int *ph;
ph = &x;
```

Ponteiros

Xavier

Paulo Henrique

(opcional)
Exibir dados do ponteiro


```
int *ph;
ph = &x;
cout << *ph;
```

Ponteiros

Xavier

Paulo Henrique

(opcional)
Atribuir algum valor ao ponteiro

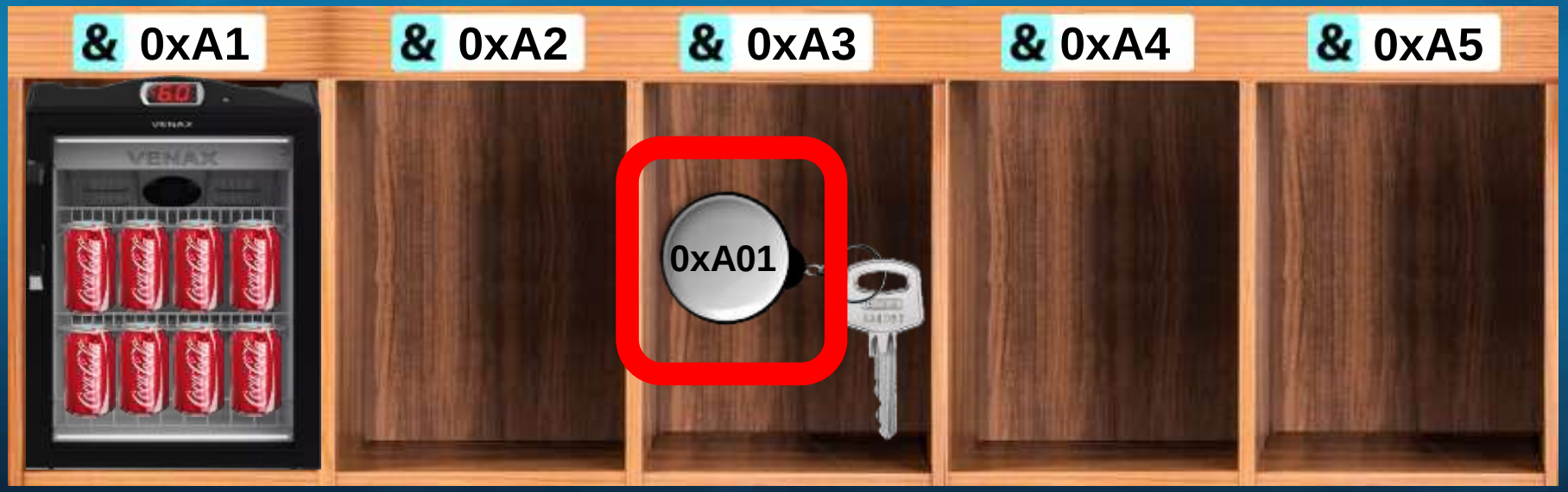
```
int *ph;
ph = &x;
*ph = 4;
```

Ponteiros

Xavier

Paulo **H**enrique

(opcional)
Exibir dados do ponteiro




```
int *ph;
ph = &x;
cout << ph;
```

MUITO RARAMENTE

Ponteiros

Xavier

Paulo **H**enrique

(opcional)
Exibir dados do ponteiro

```
int *ph;
ph = &x;
cout << &ph;
```

MAIS RARAMENTE AINDA

Ponteiros

Xavier

Paulo **H**enrique

Passos

...

...

...

Ponteiros

Xavier

Paulo **H**enrique

Paula **C**ristina

1º Passo
(OBRIGATÓRIO)
Declarar o ponteiro

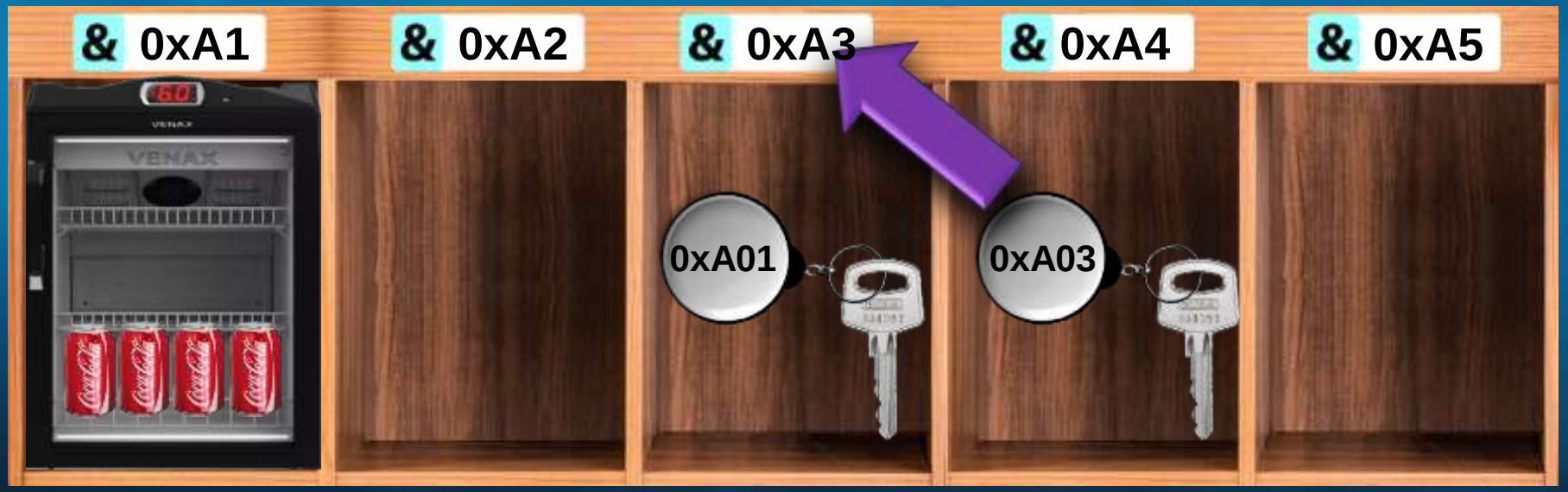

```
int **pc;
```

Ponteiros

Xavier

Paulo Henrique

Paula Cristina



2º Passo

(OBRIGATÓRIO)

Apontar para uma variável

```
int **pc;
pc = &ph;
```

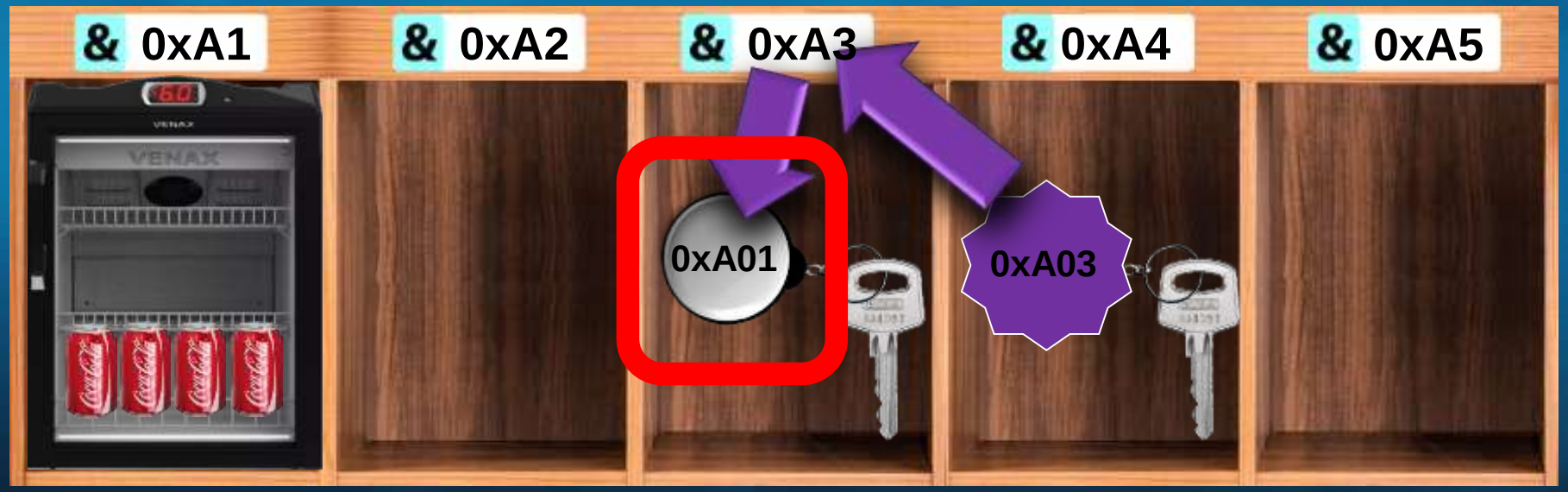
Ponteiros

Xavier

Paulo Henrique

Paula Cristina

(opcional)
Exibir dados do ponteiro




```
int **pc;
pc = &ph;
cout << *pc;
```

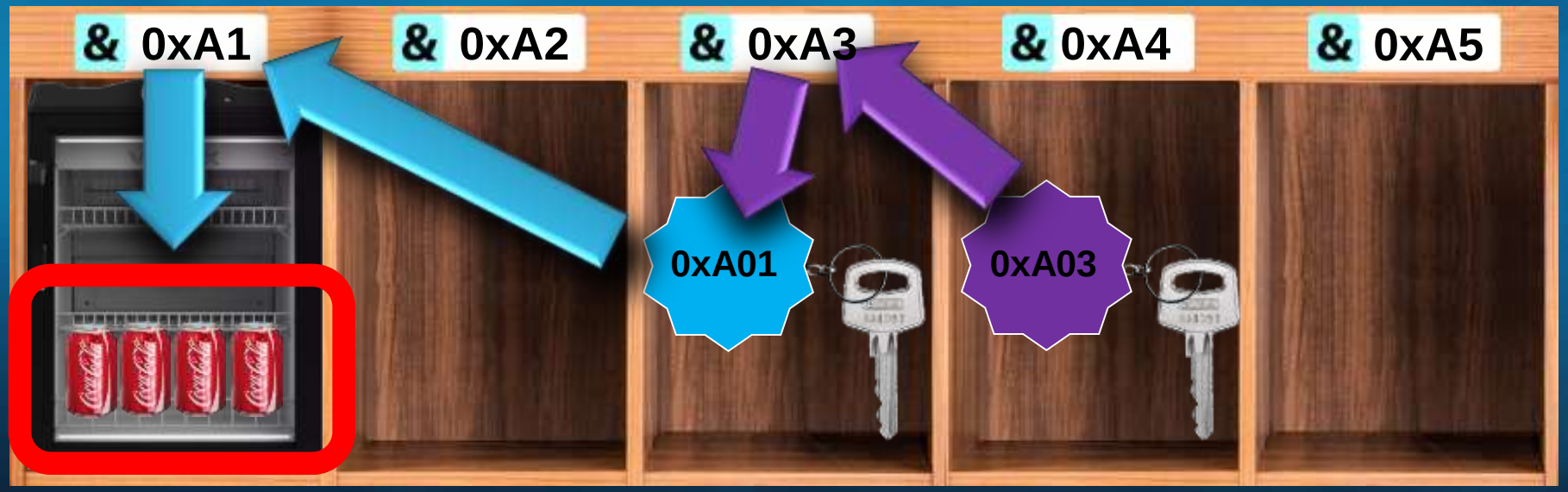
MUITO RARAMENTE

Ponteiros

Xavier

Paulo Henrique

Paula Cristina



(opcional)
Exibir dados do ponteiro

```
int **pc;
pc = &ph;
cout << **pc;
```

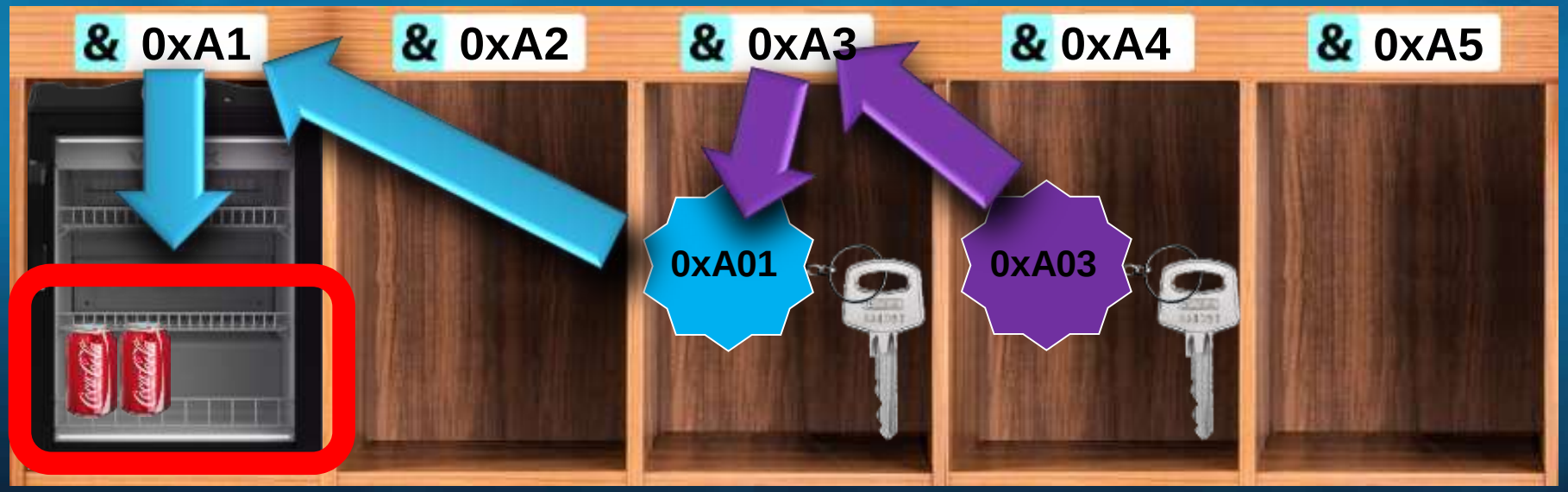
MUITO UTILIZADO

Ponteiros

Xavier

Paulo Henrique

Paula Cristina



(opcional)
Atribuir algum valor ao ponteiro

```
int **pc;
pc = &ph;
**pc = 2;
```

MUITO UTILIZADO

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

(opcional)
Exibir dados do ponteiro


```
int **pc;
pc = &ph;
cout << pc;
```

MUITO RARAMENTE

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

(opcional)
Exibir dados do ponteiro

```
int **pc;
pc = &ph;
cout << &pc;
```

MAIS RARAMENTE AINDA

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Passos

...

...

...

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Peter Parker

1º Passo
(OBRIGATÓRIO)
Declarar o ponteiro


```
int ***pp;
```

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Peter Parker

2º Passo
(OBRIGATÓRIO)
Apontar para uma variável


```
int ***pp;
pp = &pc;
```

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Exibir dados do ponteiro


```
int ***pp;
pp = &pc;
cout << *pp;
```

MUITO RARAMENTE

Ponteiros

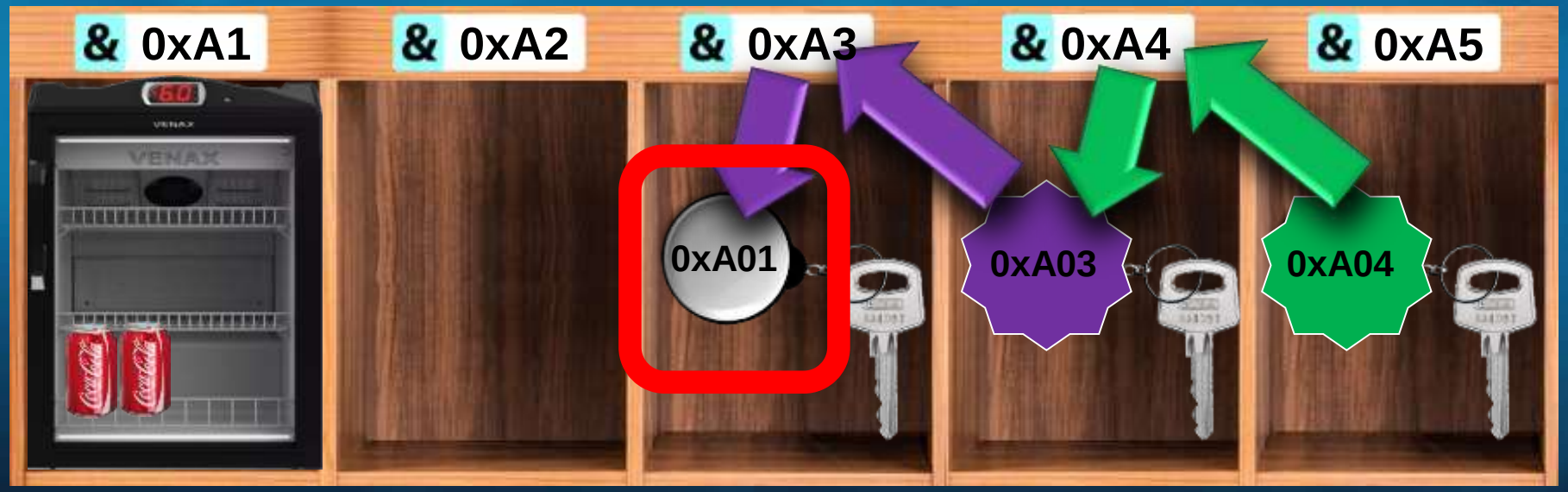
Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Exibir dados do ponteiro




```
int ***pp;
pp = &pc;
cout << **pp;
```

MUITO RARAMENTE

Ponteiros

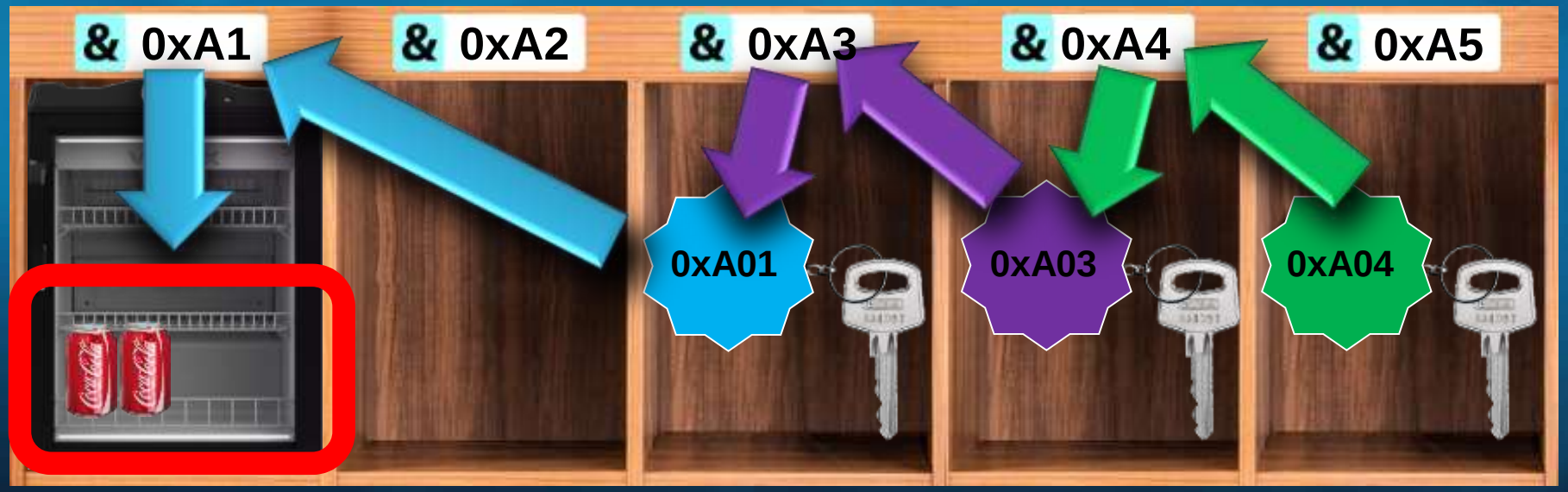
Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Exibir dados do ponteiro




```
int ***pp;
pp = &pc;
cout << ***pp;
```

MUITO UTILIZADO

Ponteiros

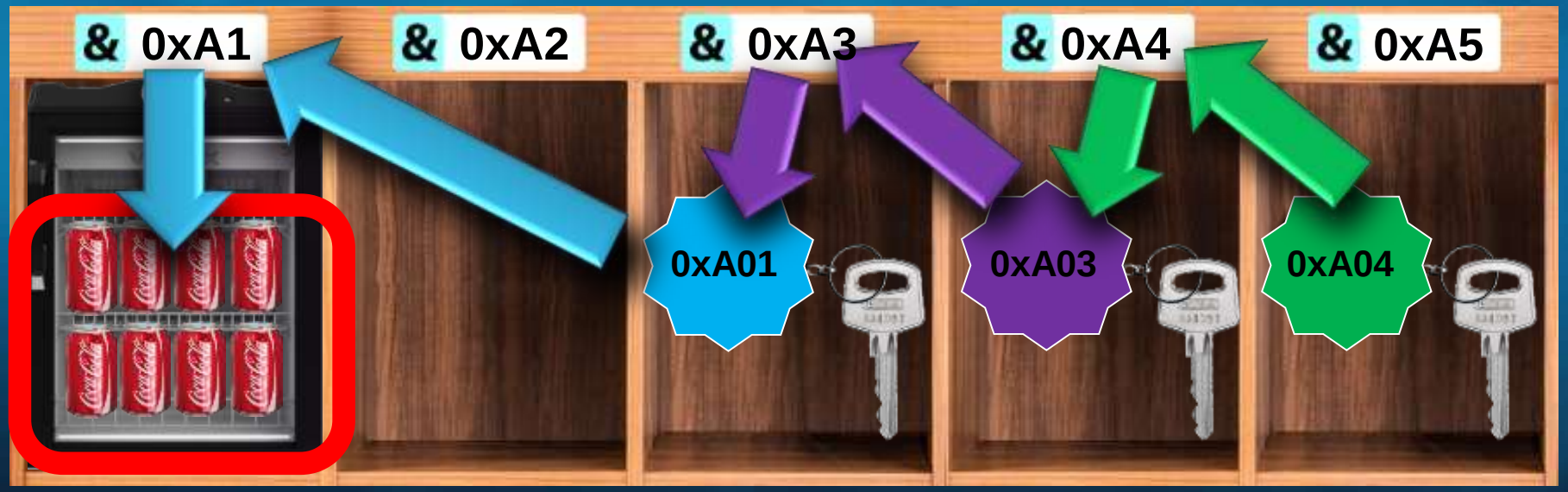
Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Atribuir algum valor ao ponteiro




```
int ***pp;
pp = &pc;
***pp = 8;
```

MUITO UTILIZADO

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Exibir dados do ponteiro


```
int ***pp;
pp = &pc;
cout << pp;
```

MUITO RARAMENTE

Ponteiros

Xavier

Paulo Henrique

Paula Cristina

Peter Parker

(opcional)
Exibir dados do ponteiro


```
int ***pp;
pp = &pc;
cout << &pp;
```

MAIS RARAMENTE AINDA

```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

O que será exibido na tela?

```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

X

Z

```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor
	0xA1	
X	0xA2	
	0xA3	
Z	0xA4	
	0xA5	
pH	0xA6	
	0xA7	
pC	0xA8	
	0xA9	
pP	0xAA	

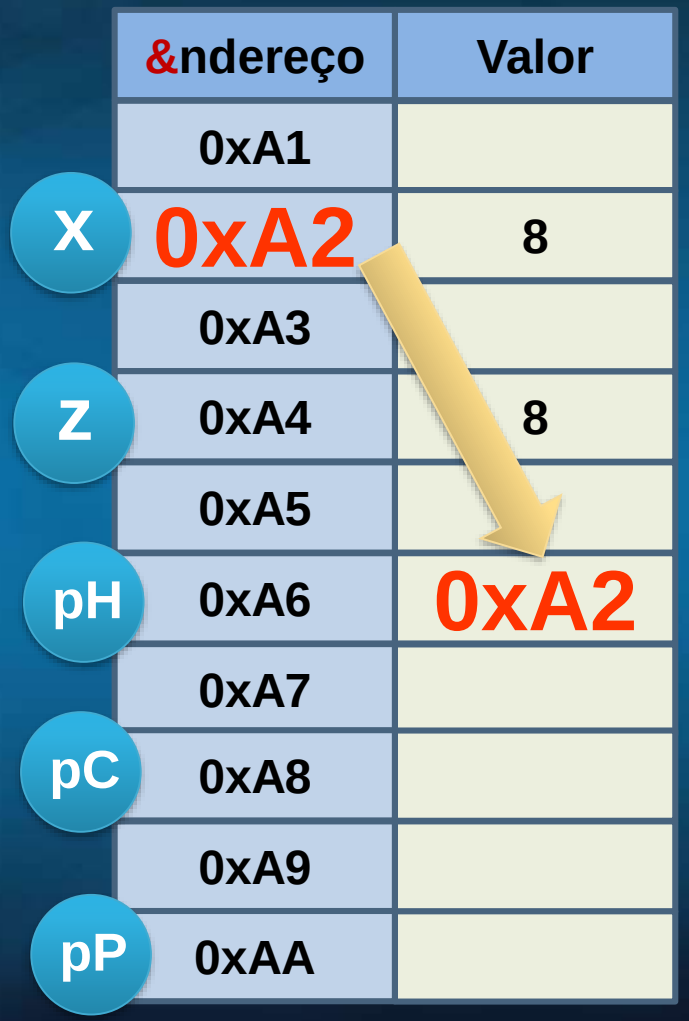
```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor
	0xA1	
X	0xA2	8
	0xA3	
Z	0xA4	
	0xA5	
pH	0xA6	
	0xA7	
pC	0xA8	
	0xA9	
pP	0xAA	

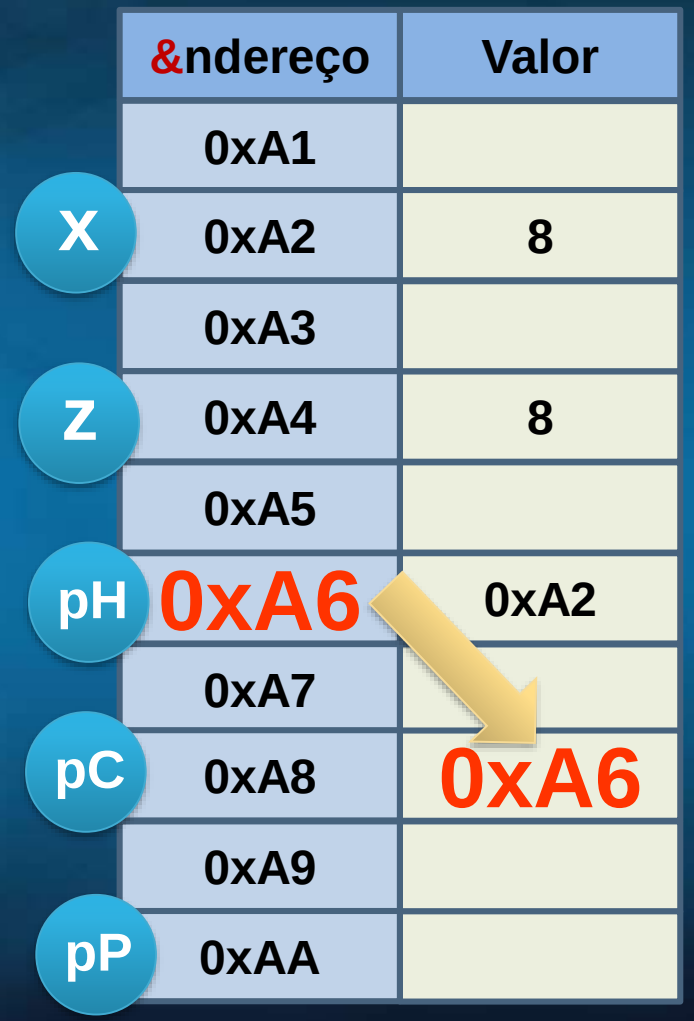
```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor
	0xA1	
X	0xA2	8
	0xA3	
Z	0xA4	8
	0xA5	
pH	0xA6	
	0xA7	
pC	0xA8	
	0xA9	
pP	0xAA	

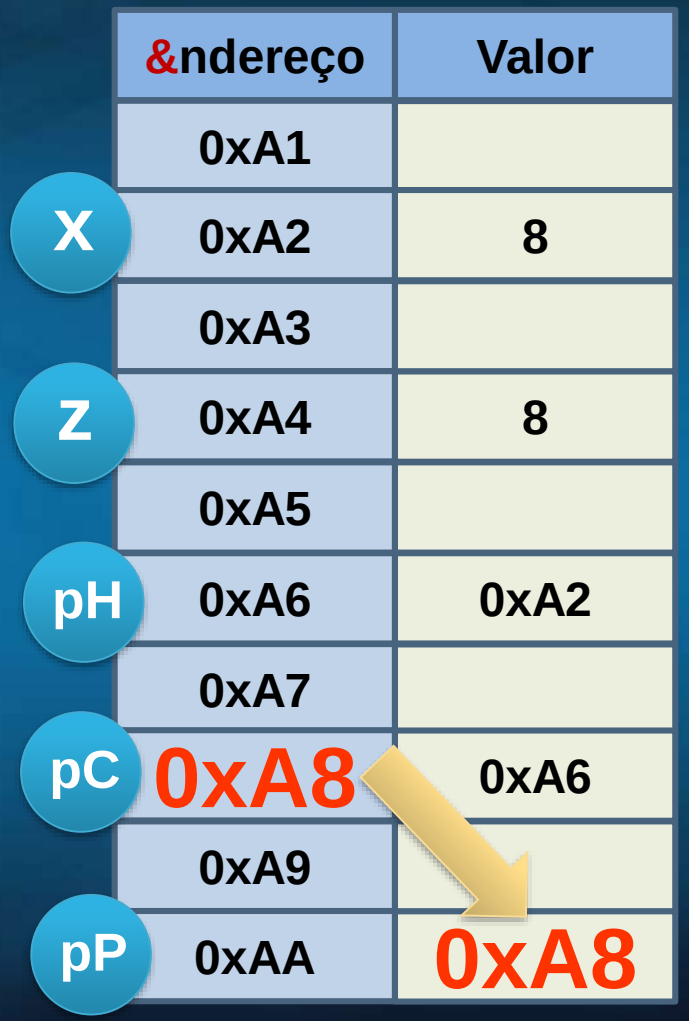
```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```



```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```



```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```



```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor	
	0xA1		
X	0xA2	8	*
	0xA3		
Z	0xA4	8	
	0xA5		
pH	0xA6	0xA2	
	0xA7		
pC	0xA8	0xA6	
	0xA9		
pP	0xAA	0xA8	

```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor	
	0xA1		
X	0xA2	8	**
	0xA3		
Z	0xA4	8	
	0xA5		
pH	0xA6	0xA2	*
	0xA7		
pC	0xA8	0xA6	
	0xA9		
pP	0xAA	0xA8	

```
int main() {
    int x, z;
    int *pH, **pC, ***pP;

    x = 8;
    z = x;

    pH = &x;
    pC = &pH;
    pP = &pC;

    cout << "Endereço de &pH: " << &pH << endl;
    cout << "Conteúdo de pH: " << pH << endl;
    cout << "Valor de *pH: " << *pH << endl << endl;

    cout << "Endereço de &pC: " << &pC << endl;
    cout << "Conteúdo de pC: " << pC << endl;
    cout << "Valor de *pC: " << *pC << endl;
    cout << "Valor de **pC: " << **pC << endl << endl;

    cout << "Endereço de &pP: " << &pP << endl;
    cout << "Conteúdo de pP: " << pP << endl;
    cout << "Valor de *pP: " << *pP << endl;
    cout << "Valor de **pP: " << **pP << endl;
    cout << "Valor de ***pP: " << ***pP << endl << endl;
}
```

	&ndereço	Valor	
	0xA1		
X	0xA2	8	***
	0xA3		
Z	0xA4	8	
	0xA5		
pH	0xA6	0xA2	**
	0xA7		
pC	0xA8	0xA6	*
	0xA9		
pP	0xAA	0xA8	

Declaração Ponteiros

tipo_do_ponteiro *nome_do_ponteiro;

É o operador *asterisco* (*) que informa ao compilador que aquela variável não vai guardar um valor, mas sim um endereço de memória para aquele tipo especificado.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main(){
4    //Declara um ponteiro para int
5    int *p;
6    //Declara um ponteiro para float
7    float *x;
8    //Declara um ponteiro para char
9    char *y;
10   //Declara uma variável do tipo int e um
         ponteiro para int
11   int soma, *p2,;
12
13   system(''pause'');
14   return 0;
15 }
```

Ponteiros

Apesar de usarem o mesmo símbolo, o operador * (**multiplicação**) não é o mesmo operador que o * (**referência de ponteiros**).

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main(){
4    int x = 3, y = 5, z;
5    z = y * x;
6    int *p;
7
8    system(``pause'');
9    return 0;
10 }
```

Ponteiros – Operador * e &

Ao se trabalhar com ponteiros, duas tarefas básicas serão sempre executadas:

- acessar o endereço de memória de uma variável;
- acessar o conteúdo de um endereço de memória;

Para realizar essas tarefas, iremos sempre utilizar apenas dois operadores: o operador "*" e o operador "&".

Operador "*" versus operador "&"	
"*"	Declara um ponteiro: **int *x;** Conteúdo para onde o ponteiro aponta: **int y = *x;**
"&"	Endereço onde uma variável está guardada na memória: **&y**

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float y;
    float *pY;

    y = 500;

    pY = &y;

    *pY += 200;

    cout << "Valor de Y: " << y << endl;
    cout << "Valor de *pY: " << *pY << endl;

    system("pause");
    return 0;
}
```

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

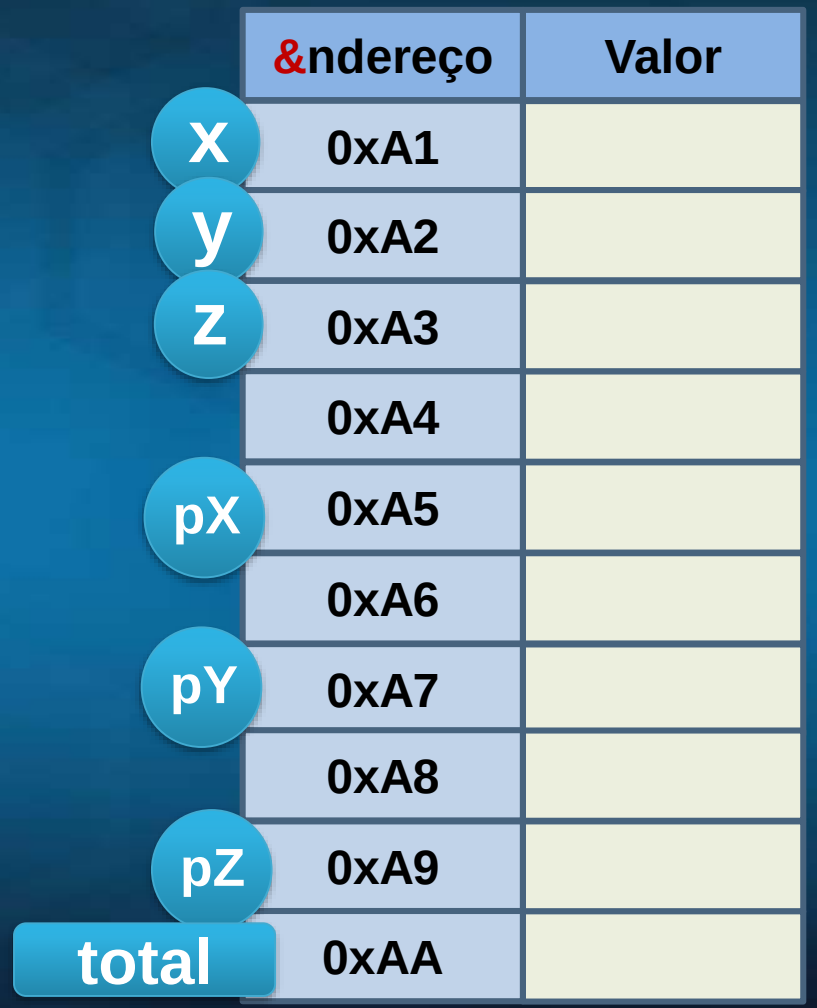
Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	
y	0xA2	
z	0xA3	
	0xA4	
	0xA5	
	0xA6	
	0xA7	
	0xA8	
	0xA9	
total	0xAA	

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```



Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	
	0xA6	
pY	0xA7	
	0xA8	
pZ	0xA9	
total	0xAA	

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

5

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	

*

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

5 50

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	

*

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

5 50 500

	&ndereço	Valor	
x	0xA1	5	
y	0xA2	50	
z	0xA3	500	*
	0xA4		
pX	0xA5	0xA1	
	0xA6		
pY	0xA7	0xA2	
	0xA8		
pZ	0xA9	0xA3	
total	0xAA		

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

5 50 500

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	555

Exemplos – Operador * e &

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x, y, z, total;
    float *pX, *pY, *pZ;

    x = 5;
    y = 50;
    z = 500;

    pX = &x;
    pY = &y;
    pZ = &z;

    total = *pX + *pY + *pZ;

    cout << "Total: " << total << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
x	0xA1	5
y	0xA2	50
z	0xA3	500
	0xA4	
pX	0xA5	0xA1
	0xA6	
pY	0xA7	0xA2
	0xA8	
pZ	0xA9	0xA3
total	0xAA	555

Exercício

```
60  #include <iostream>
61  #include <locale.h>
62  #include <stdlib.h>
63
64  using namespace std;
65
66  int main() {
67      setlocale(LC_ALL, "Portuguese");
68
69      int i, j, *p;
70
71      i = 99;
72      p = &i;
73      j = *p + 100;
74
75      cout << j << endl;
76
77      return 0;
78  }
```

Qual a saída?

Exercício

```
82  #include <iostream>
83  #include <locale.h>
84  #include <stdlib.h>
85
86  using namespace std;
87
88  int main() {
89      setlocale(LC_ALL, "Portuguese");
90
91      int a, b, c, *p, *q;
92
93      a = 5;
94      b = 12;
95      p = &a;
96      q = &b;
97      c = *p + *q;
98
99      cout << c << endl;
100
101     return 0;
102 }
```

Qual a saída?

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

X

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
X	0xA1	
	0xA2	
	0xA3	
	0xA4	
pA	0xA5	
	0xA6	
pB	0xA7	
	0xA8	
	0xA9	
	0xAA	

Ponteiros

```c
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
X	0xA1	300
	0xA2	
	0xA3	
	0xA4	
pA	0xA5	
	0xA6	
pB	0xA7	
	0xA8	
	0xA9	
	0xAA	

Ponteiros

```c
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
X	0xA1	300
	0xA2	
	0xA3	
	0xA4	
pA	0xA5	0xA1
	0xA6	
pB	0xA7	
	0xA8	
	0xA9	
	0xAA	

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
X	0xA1	300
	0xA2	
	0xA3	
	0xA4	
pA	0xA5	0xA1
	0xA6	
pB	0xA7	0xA1
	0xA8	
	0xA9	
	0xAA	

Ponteiros

```
int main() {
    setlocale(LC_ALL, "Portuguese");

    float x;
    float *pA, *pB;

    x = 300;

    pA = &x;
    pB = pA;

    cout << "Endereço de X: " << &x << endl;
    cout << "Conteúdo de pA: " << pA << endl;
    cout << "Conteúdo de pB: " << pB << endl;
    cout << "Valor de *pB: " << *pB << endl;

    system("pause");
    return 0;
}
```

	&ndereço	Valor
X	0xA1	300
	0xA2	
	0xA3	
	0xA4	
pA	0xA5	0xA1
	0xA6	
pB	0xA7	0xA1
	0xA8	
	0xA9	0xA1
	0xAA	

Funções
Passagem de parâmetros por valor

Ponteiros

Ponteiros e funções

- Passagem por **VALOR**:

```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portugues");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```

&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros e funções

- Passagem por **VALOR**:

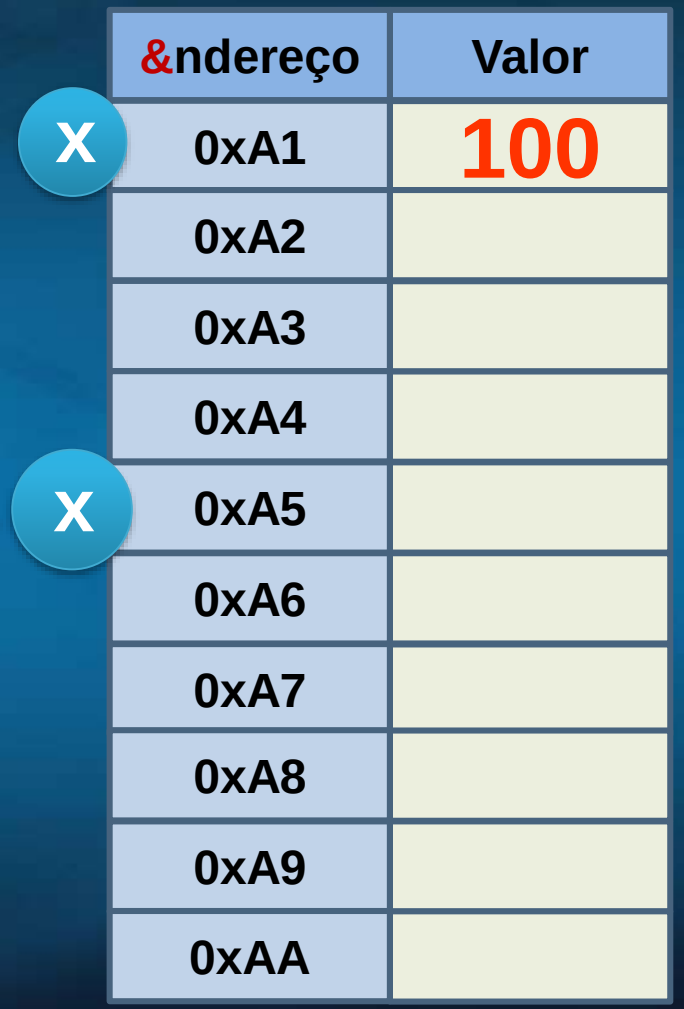
```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portugues");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```

&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros e funções

- Passagem por **VALOR**:

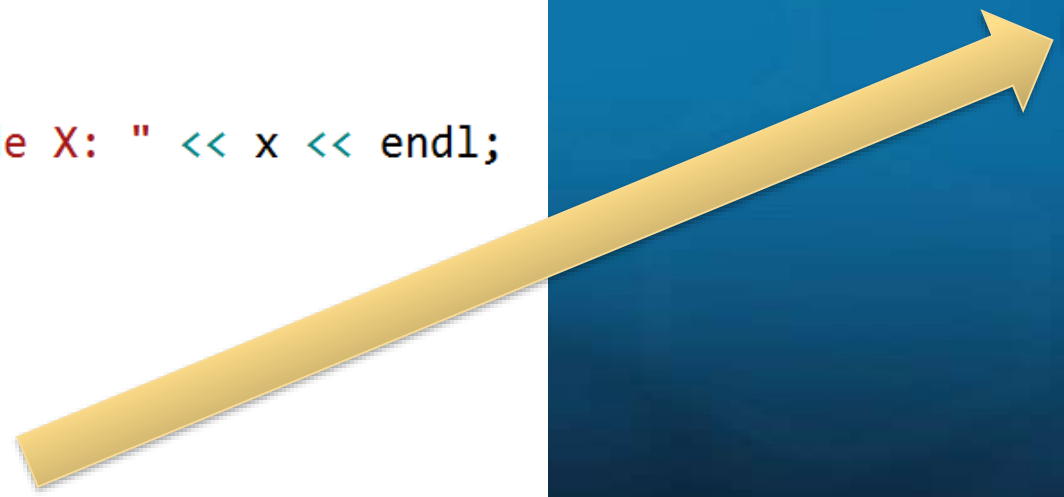
```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```



Ponteiros e funções

- Passagem por **VALOR**:

```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```

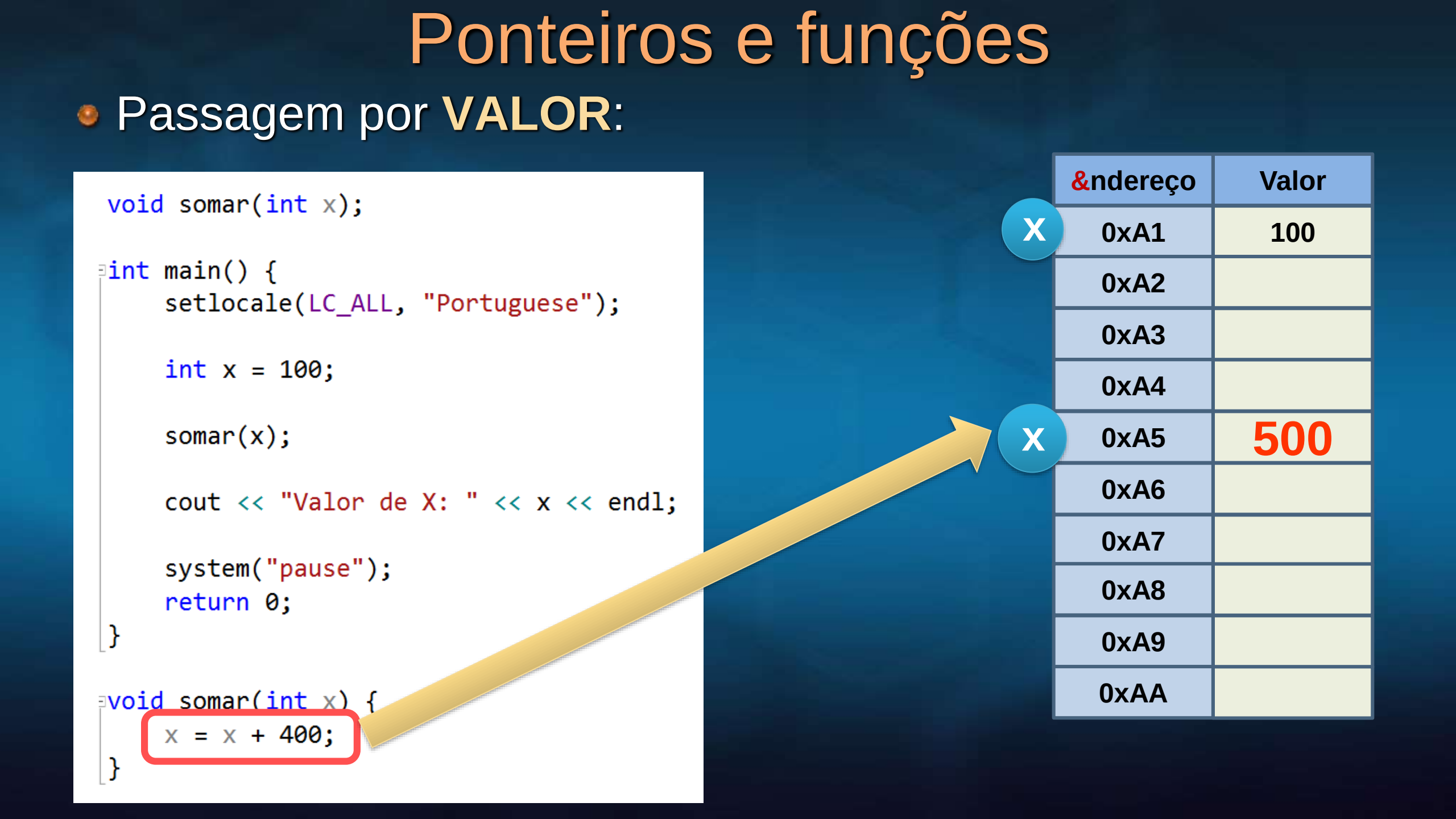


	&ndereço	Valor
X	0xA1	100
	0xA2	
	0xA3	
	0xA4	
X	0xA5	100
	0xA6	
	0xA7	
	0xA8	
	0xA9	
	0xAA	

Ponteiros e funções

- Passagem por **VALOR**:

```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```

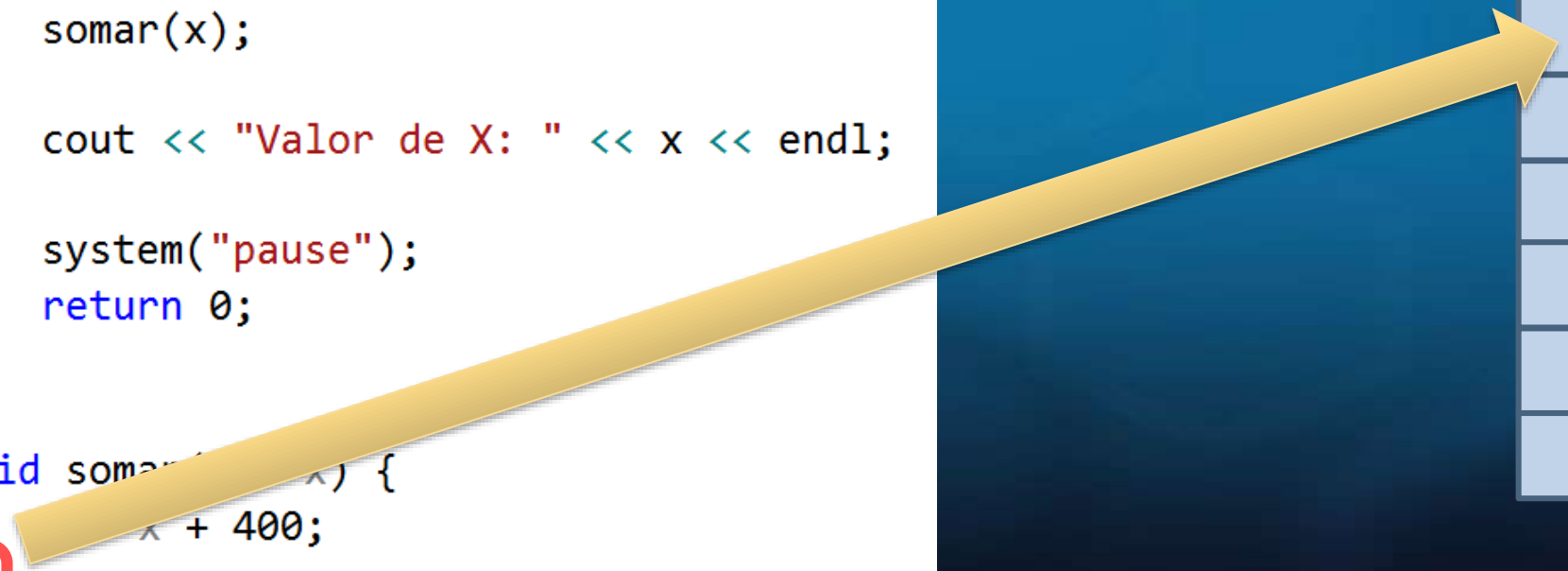


&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	500
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros e funções

- Passagem por **VALOR**:

```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar( x) {
     x + 400;
}
```



&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

X

Ponteiros e funções

- Passagem por **VALOR**:

```
void somar(int x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int x) {
    x = x + 400;
}
```

&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

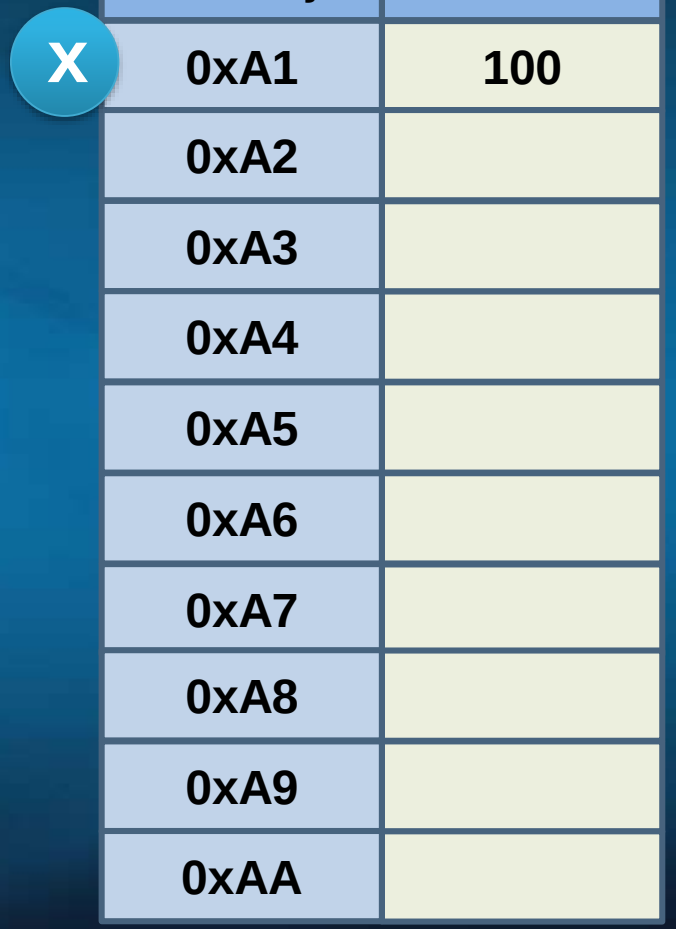
Funções
Passagem de parâmetros por referência

Ponteiros

Ponteiros e funções

- Passagem por **REFERÊNCIA**:

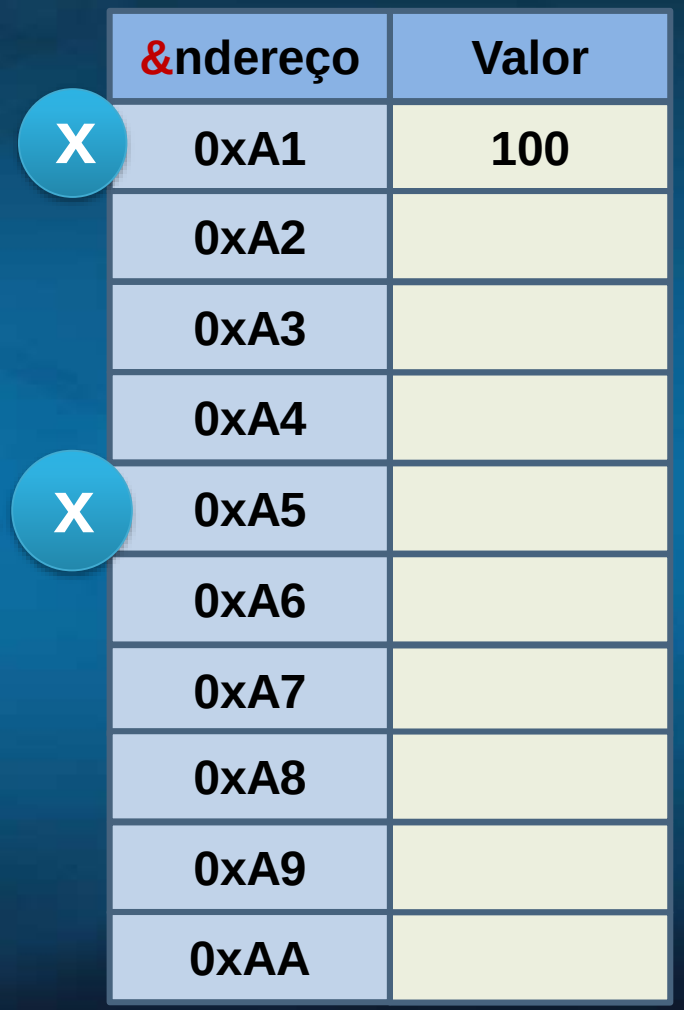
```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portugues");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



Ponteiros e funções

- Passagem por **REFERÊNCIA**:

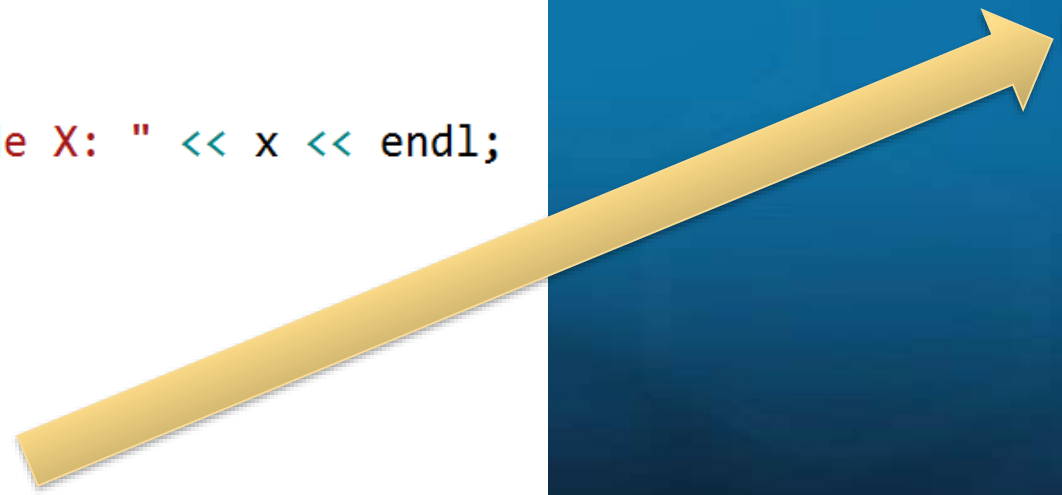
```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



Ponteiros e funções

- Passagem por **REFERÊNCIA**:

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```

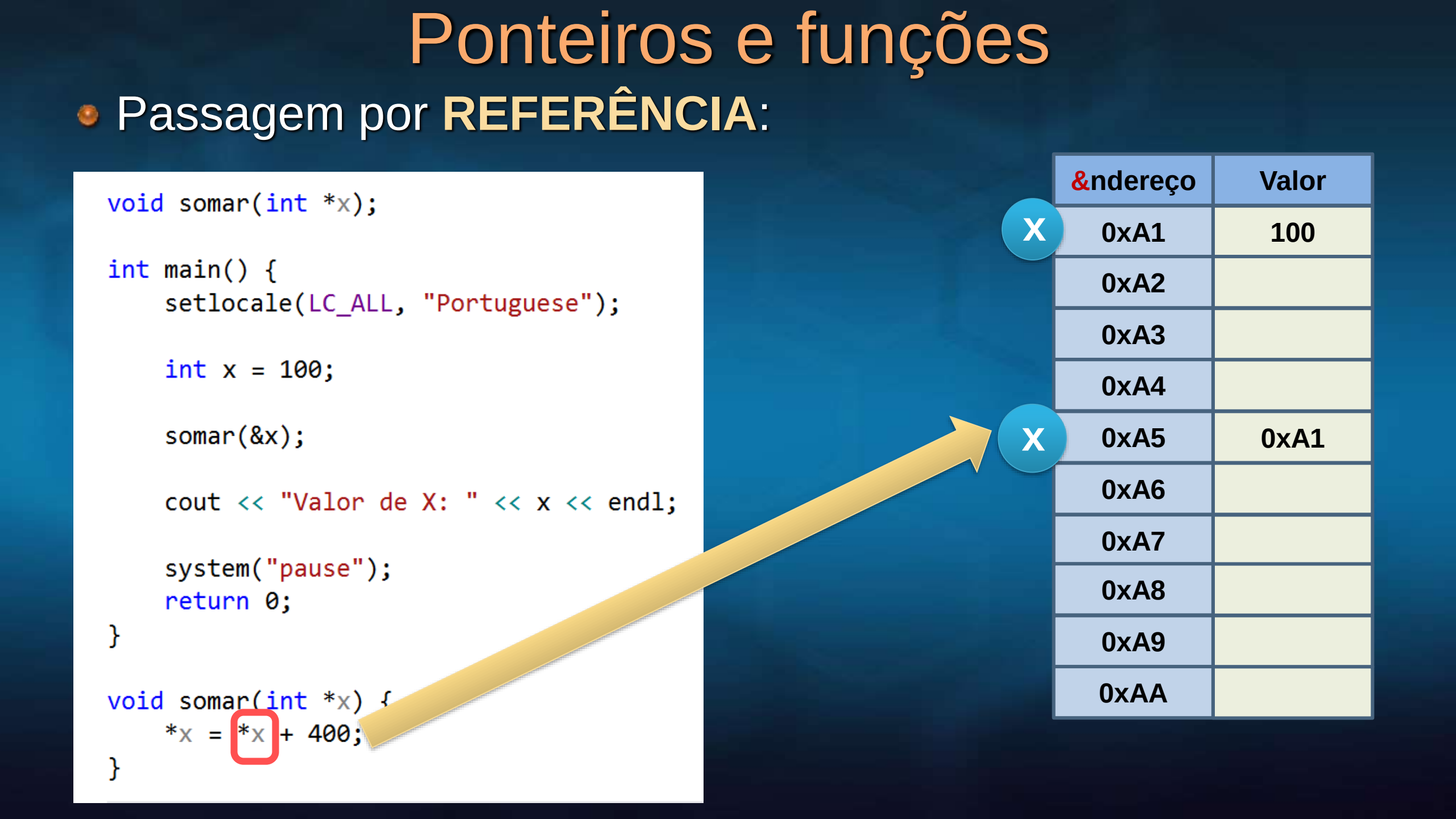


	&ndereço	Valor
x	0xA1	100
	0xA2	
	0xA3	
	0xA4	
x	0xA5	0xA1
	0xA6	
	0xA7	
	0xA8	
	0xA9	
	0xAA	

Ponteiros e funções

- Passagem por **REFERÊNCIA**:

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```

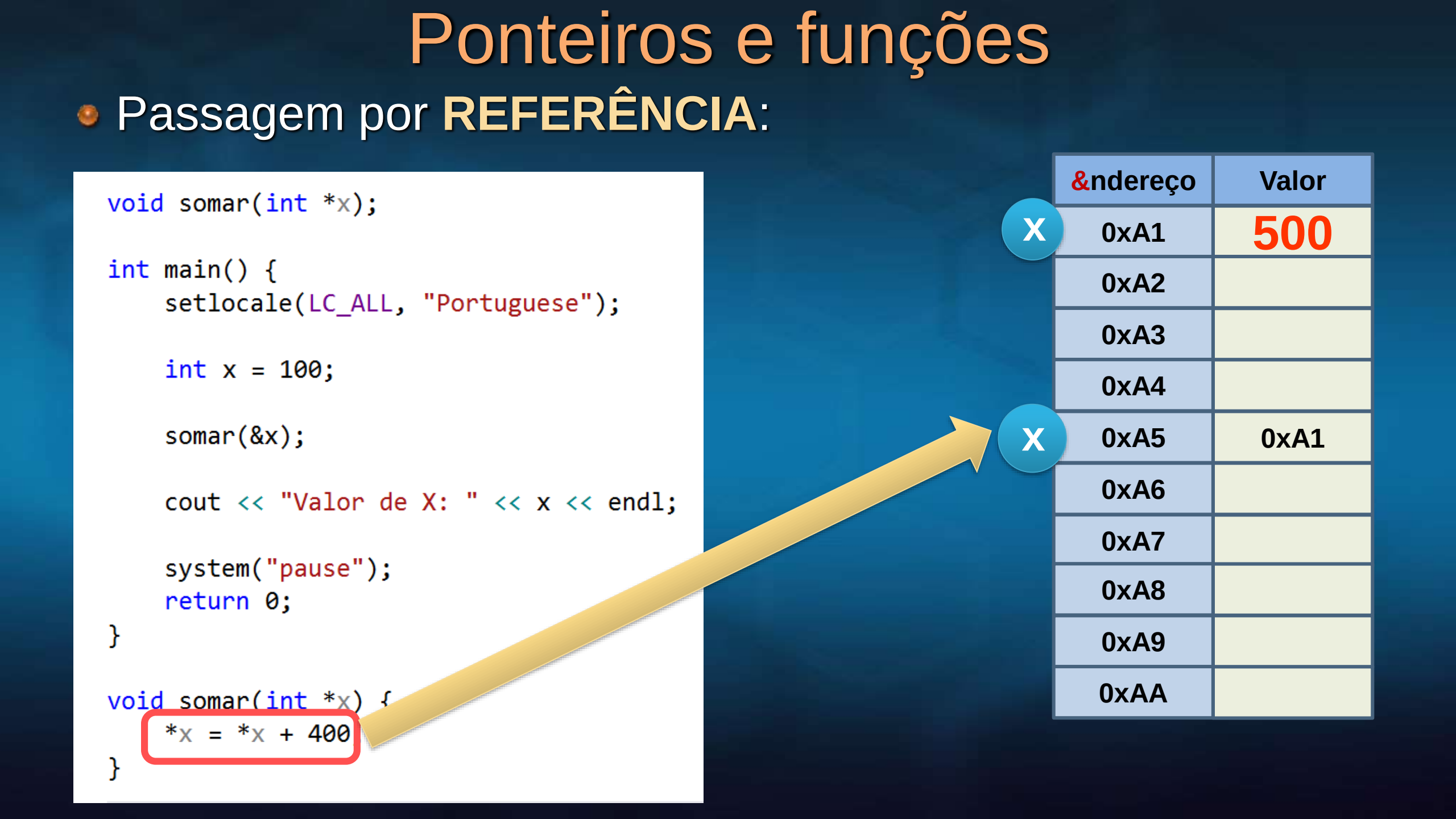


&ndereço	Valor
0xA1	100
0xA2	
0xA3	
0xA4	
0xA5	0xA1
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros e funções

- Passagem por **REFERÊNCIA**:

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400
}
```

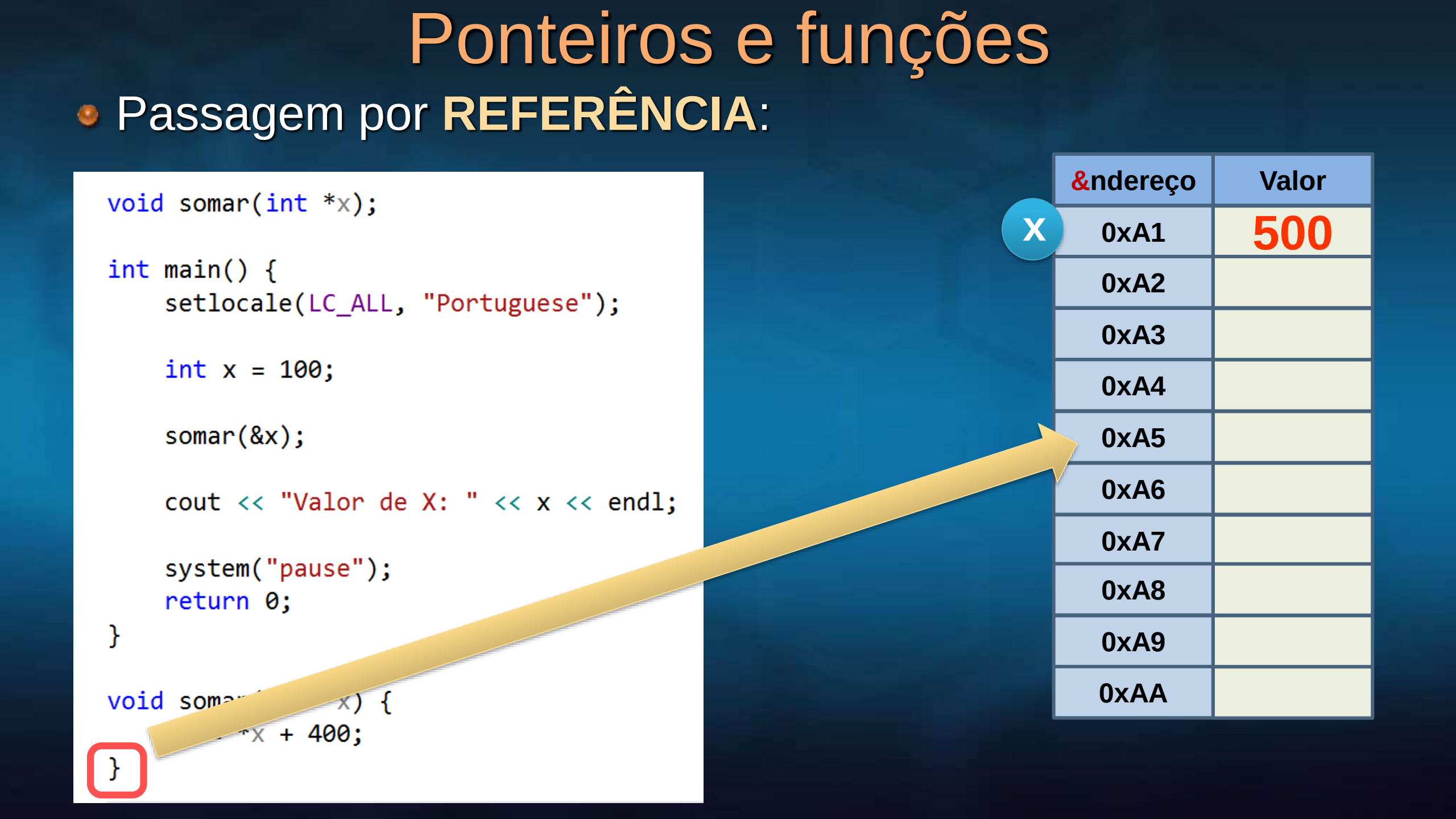


&ndereço	Valor
0xA1	500
0xA2	
0xA3	
0xA4	
0xA5	0xA1
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Ponteiros e funções

- Passagem por **REFERÊNCIA**:

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



Ponteiros e funções

- Passagem por **REFERÊNCIA**:

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;

    somar(&x);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```

Problema - Exercício

- Como obter o retorno das três variáveis utilizando **uma única função**?

```
void obterXYZ(int x, int y, int z);

int main() {
    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(x, y, z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int x, int y, int z) {
    x = 100;
    y = 200;
    z = 300;
}
```

Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```

&ndereço	Valor
0xA1	
0xA2	
0xA3	
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```

	&ndereço	Valor
x	0xA1	
y	0xA2	
z	0xA3	
	0xA4	
	0xA5	
	0xA6	
	0xA7	
	0xA8	
	0xA9	
	0xAA	

Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```

&ndereço	Valor
x 0xA1	0
y 0xA2	0
z 0xA3	0
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

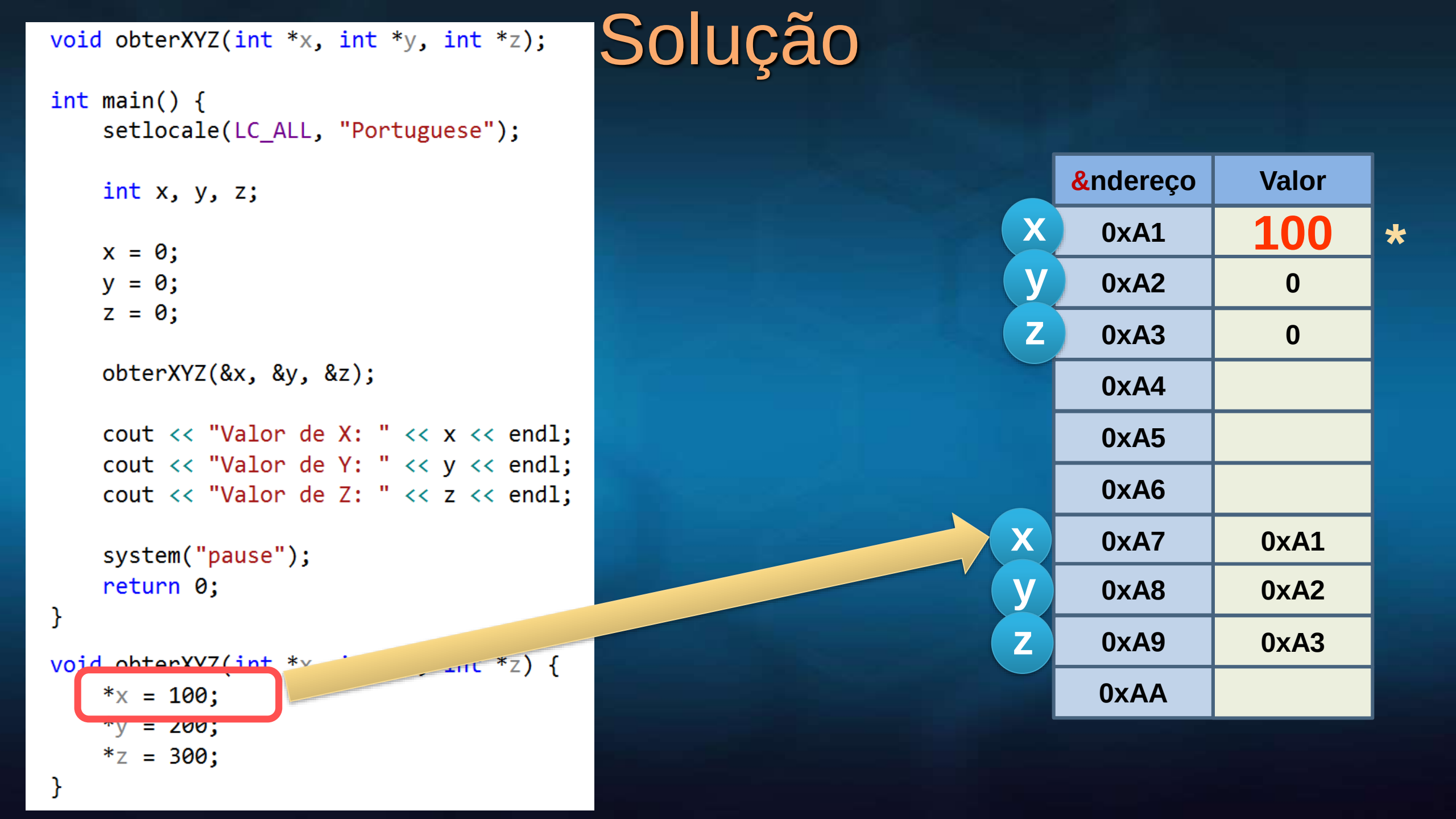
Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;


void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```

	&ndereço	Valor
x	0xA1	0
y	0xA2	0
z	0xA3	0
	0xA4	
	0xA5	
	0xA6	
x	0xA7	0xA1
y	0xA8	0xA2
z	0xA9	0xA3
	0xAA	

Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```



Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z)
    *x = 100;
    *y = 200;
    *z = 300;
}
```

	&ndereço	Valor	
x	0xA1	100	
y	0xA2	200	*
z	0xA3	0	
	0xA4		
	0xA5		
	0xA6		
x	0xA7	0xA1	
y	0xA8	0xA2	
z	0xA9	0xA3	
	0xAA		

Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```

	&ndereço	Valor	
x	0xA1	100	
y	0xA2	200	
z	0xA3	300	*
	0xA4		
	0xA5		
	0xA6		
x	0xA7	0xA1	
y	0xA8	0xA2	
z	0xA9	0xA3	
	0xAA		

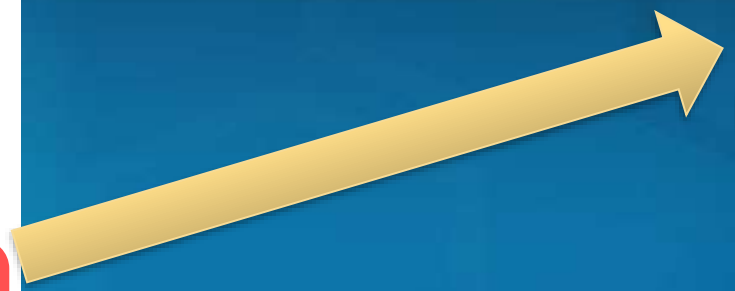
Solução

```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z
}
```

	&ndereço	Valor
x	0xA1	100
y	0xA2	200
z	0xA3	300
	0xA4	
	0xA5	
	0xA6	
	0xA7	
	0xA8	
	0xA9	
	0xAA	

Solução

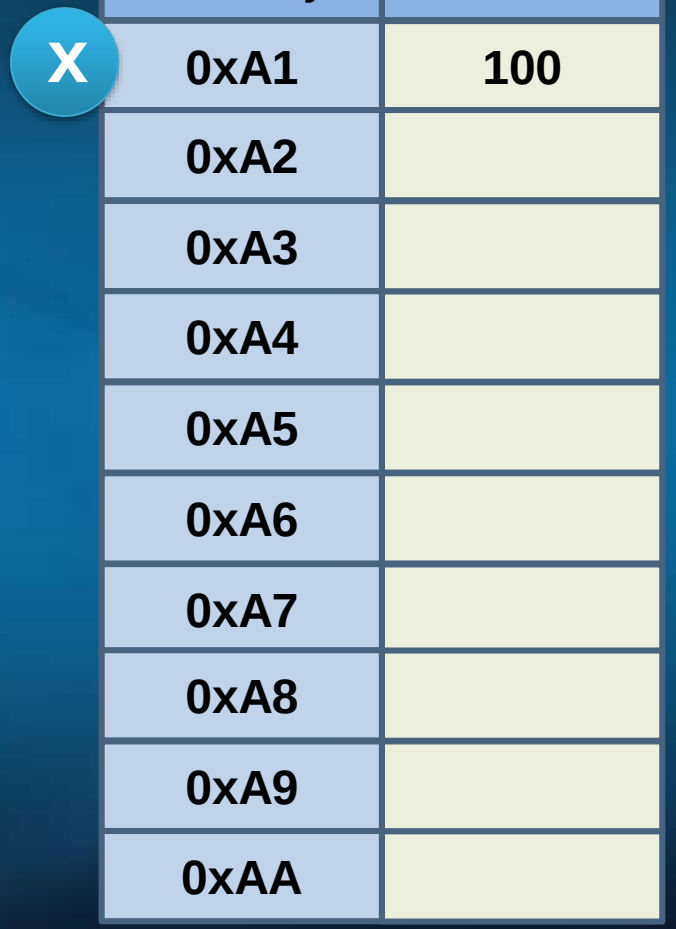
```
void obterXYZ(int *x, int *y, int *z);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x, y, z;

    x = 0;
    y = 0;
    z = 0;

    obterXYZ(&x, &y, &z);

    cout << "Valor de X: " << x << endl;
    cout << "Valor de Y: " << y << endl;
    cout << "Valor de Z: " << z << endl;

    system("pause");
    return 0;
}

void obterXYZ(int *x, int *y, int *z) {
    *x = 100;
    *y = 200;
    *z = 300;
}
```



&ndereço	Valor
x 0xA1	100
y 0xA2	200
z 0xA3	300
0xA4	
0xA5	
0xA6	
0xA7	
0xA8	
0xA9	
0xAA	

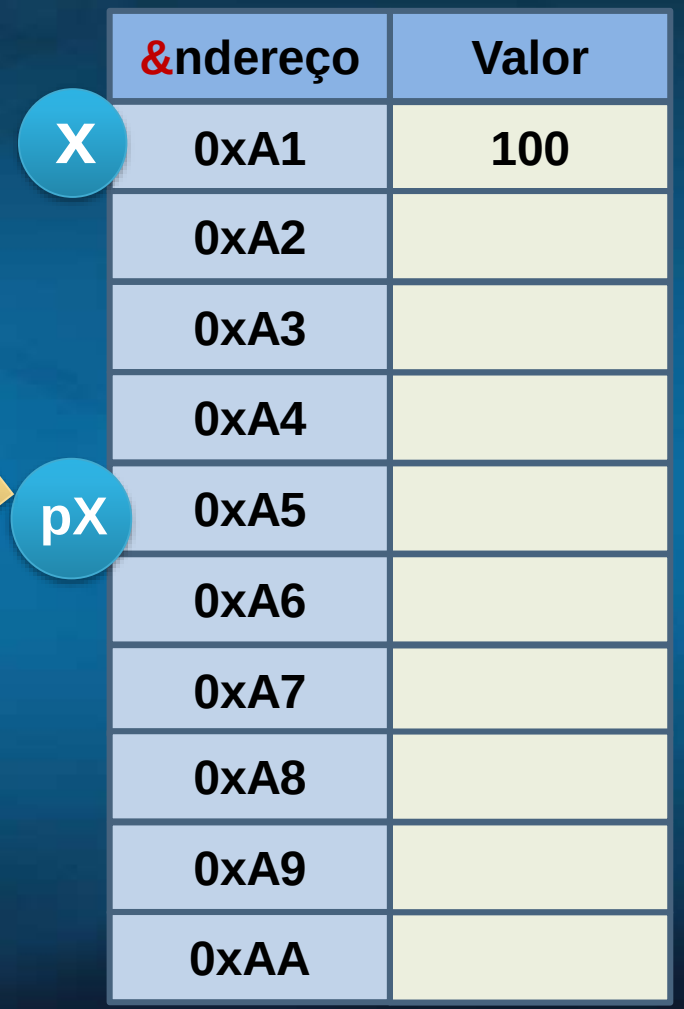
Passando ponteiros para funções

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



Passando ponteiros para funções

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



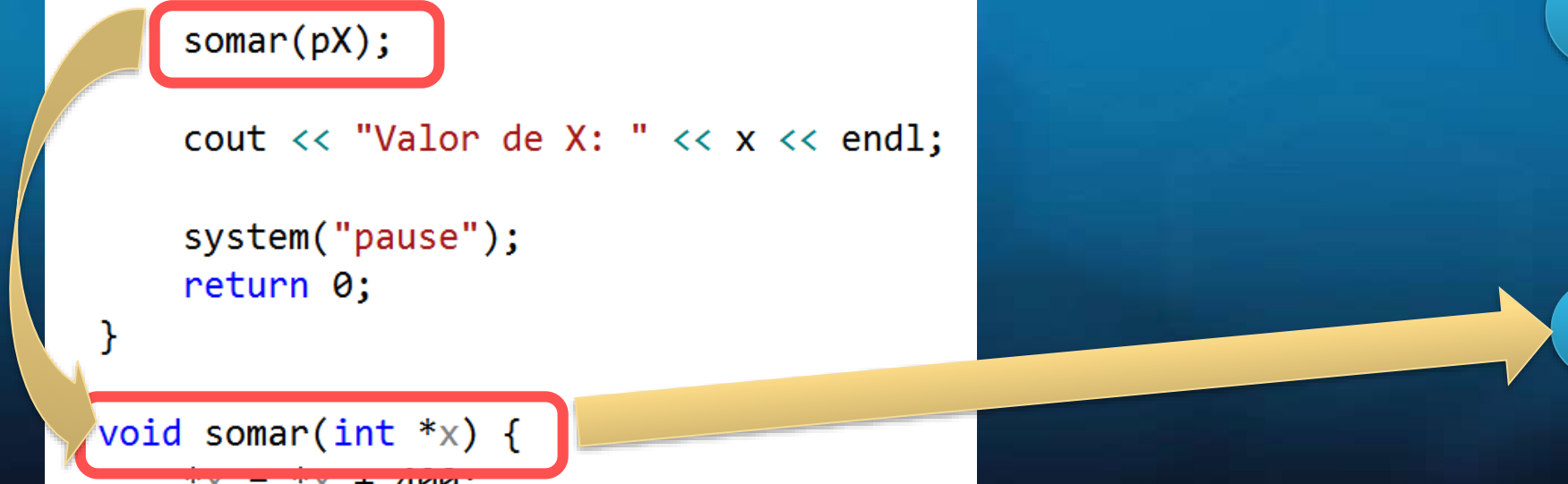
Passando ponteiros para funções

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```

	&ndereço	Valor
x	0xA1	100
	0xA2	
	0xA3	
	0xA4	
pX	0xA5	0xA1
	0xA6	
	0xA7	
	0xA8	
	0xA9	
	0xAA	

Passando ponteiros para funções

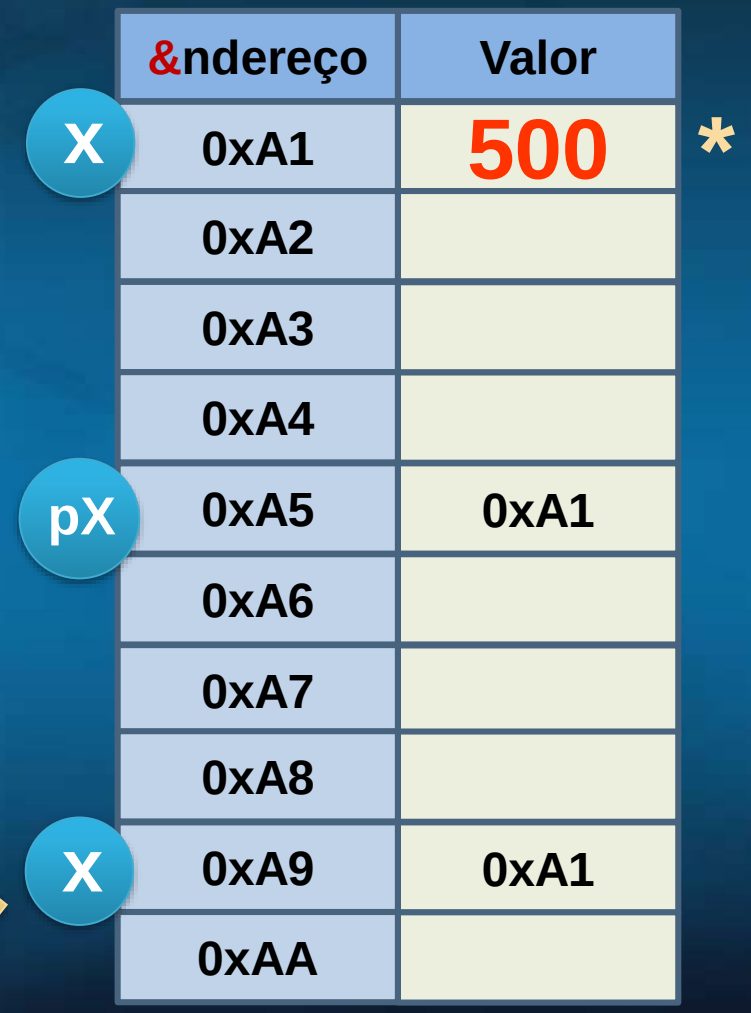
```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



	&ndereço	Valor
X	0xA1	100
	0xA2	
	0xA3	
	0xA4	
pX	0xA5	0xA1
	0xA6	
	0xA7	
	0xA8	
X	0xA9	0xA1
	0xAA	

Passando ponteiros para funções

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *x + 400;
}
```



Passando ponteiros para funções

```
void somar(int *x);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int x = 100;
    int *pX;

    pX = &x;

    somar(pX);

    cout << "Valor de X: " << x << endl;

    system("pause");
    return 0;
}

void somar(int *x) {
    *x = *
}
```

	&ndereço	Valor	
X	0xA1	100	*
	0xA2		
	0xA3		
	0xA4		
pX	0xA5	0xA1	
	0xA6		
	0xA7		
	0xA8		
	0xA9		
	0xAA		

Exercício

O método `misterio(&i,&j)` tem um problema. Qual é?

Antes da chamada do método,

temos a seguinte linha de comando: `int i=6, j=10;`

```
void misterio(int *p, int *q){
        int *temp;
        *temp = *p;
        *p = *q;
        *q = *temp;
}
```

Exercício

1) Fazer uma função que retorna a soma e a subtração entre dois números. Após a função, imprima os dados na tela.

Ex: void calculo(int nro1, int nro2, int *soma, int *sub)

2) Fazer uma função para ler e retornar o valor das 4 notas de um aluno e sua média. Após a função, imprima os dados na tela.

3) Fazer uma função que recebe um número, calcule e retorne se o número é positivo ou negativo e se é múltiplo ou não de três. Após a função, imprima os dados na tela.

Ex: void validarNro(int nro1, bool *positivo, bool *multiploTres)

Exercício 1

```
void calculadora(int nro1, int nro2, int *sm, int *sb);

int main() {
    setlocale(LC_ALL, "Portuguese");
    int soma, subtracao;

    calculadora(20, 10, &soma, &subtracao);

    cout << "Soma: " << soma << endl;
    cout << "Subtração: " << subtracao << endl;

    system("pause");
    return 0;
}

void calculadora(int nro1, int nro2, int *sm, int *sb) {
    *sm = nro1 + nro2;
    *sb = nro1 - nro2;
}
```

Exercício 2 – Parte 1

```
void notas(float *n1, float *n2, float *n3, float *n4, float *media);

int main() {
    setlocale(LC_ALL, "Portuguese");
    float nota1, nota2, nota3, nota4, media;

    notas(&nota1, &nota2, &nota3, &nota4, &media);

    cout << "Nota 1: " << nota1 << endl;
    cout << "Nota 2: " << nota2 << endl;
    cout << "Nota 3: " << nota3 << endl;
    cout << "Nota 4: " << nota4 << endl;
    cout << "Média: " << media << endl;

    system("pause");
    return 0;
}
```

Exercício 2 – Parte 2

```
void notas(float *n1, float *n2, float *n3, float *n4, float *media) {

    cout << "Informe a nota 1: ";
    cin >> *n1;

    cout << "Informe a nota 2: ";
    cin >> *n2;

    cout << "Informe a nota 3: ";
    cin >> *n3;

    cout << "Informe a nota 4: ";
    cin >> *n4;

    *media = (*n1 + *n2 + *n3 + *n4) / 4;
}
```

Exercício 3 – Parte 1

```
void validarNro(int nro1, bool *positivo, bool *multiploTres);

int main() {
    setlocale(LC_ALL, "Portuguese");

    int nro1;
    bool pos, multTres;

    nro1 = 9;

    validarNro(nro1, &pos, &multTres);

    if (pos) {
        cout << "O número é positivo." << endl;
    }
    else {
        cout << "O número é negativo." << endl;
    }

    if (multTres) {
        cout << "O número é múltiplo de três." << endl;
    }
    else {
        cout << "O número não é múltiplo de três." << endl;
    }

    system("pause");
    return 0;
}
```

Exercício 3 – Parte 2

```
void validarNro(int nro1, bool *positivo, bool *multiploTres)
{
    if (nro1 >= 0) {
        *positivo = true;
    }
    else {
        *positivo = false;
    }

    if ((nro1 % 3) == 0) {
        *multiploTres = true;
    }
    else {
        *multiploTres = false;
    }
}
```

F I M

“Nossa história, depositada nas mãos de Deus, pode ser reescrita a qualquer momento”.

(Pe. Luís Erlin - 9 Meses com Maria)

Prof. Dr. Ricardo Luis Balieiro
ricbalieiro@gmail.com