

AULA 02

ESTRUTURA DE DADOS

Criação de uma primeira estrutura

Norton T. Roman & Luciano A. Digiampietri

Criando uma estrutura simples

Criando uma estrutura simples

Queremos criar uma estrutura para guardar algumas informações

Criando uma estrutura simples

Queremos criar uma estrutura para guardar algumas informações

Como modelamos essa estrutura?

Criando uma estrutura simples

Queremos criar uma estrutura para guardar algumas informações

Como modelamos essa estrutura?

Como instanciamos essa estrutura?

Criando uma estrutura simples

Queremos criar uma estrutura para guardar algumas informações

Como modelamos essa estrutura?

Como instanciamos essa estrutura?

Como acessamos seus campos?

Código em Java

Código em Java

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

Código em Java

```
public class PesoAltura {
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
}

public class EstruturaSimples {
    public static final int alturaMaxima = 225;
    public static void main(String[] args) {
        PesoAltura pessoa1 = new PesoAltura();
        pessoa1.peso = 80;
        pessoa1.altura = 185;
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");
        else System.out.println("Altura abaixo da maxima.");
    }
}
```

Criando uma estrutura em C

Em C, a seguinte sintaxe é utilizada para definir estruturas:

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

A sintaxe **struct { ... }** define uma estrutura com os campos definidos dentro das chaves

A sintaxe **typedef ... *PesoAltura*** define o nome *PesoAltura* para o que foi definido nas *reticências*

Criando uma estrutura em C

Em C, a seguinte sintaxe é utilizada para definir estruturas:

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

```
typedef int CHAVE;
```

A sintaxe ***struct { ... }*** define uma estrutura com os campos definidos dentro das chaves

A sintaxe ***typedef ... PesoAltura*** define o nome *PesoAltura* para o que foi definido nas *reticências*

Definimos que a palavra **CHAVE** poderá ser usada no código para representar o tipo *int* (inteiro), isto é, damos um novo nome para o tipo *int*

Comparando Java e C

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

Comparando Java e C

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
typedef struct{  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
} PesoAltura;
```

Comparando Java e C

```
public class PesoAltura {
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
}

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

public static final int alturaMaxima = 225;
```

Comparando Java e C

```
public class PesoAltura {
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
}

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

public static final int alturaMaxima = 225;

#define alturaMaxima 225
```

Comparando Java e C

```
public static void main(String[] args) {
```

Comparando Java e C

```
public static void main(String[] args) {  
  
    int main() {  
        //...  
        return 0;  
    }  
}
```

Comparando Java e C

```
public static void main(String[] args) {
```

```
    int main() {  
        //...  
        return 0;  
    }
```

```
    PesoAltura pessoa1 = new PesoAltura();  
    pessoa1.peso = 80;  
    pessoa1.altura = 185;
```

Comparando Java e C

```
public static void main(String[] args) {
```

```
    int main() {  
        //...  
        return 0;  
    }
```

```
    PesoAltura pessoa1 = new PesoAltura();  
    pessoa1.peso = 80;  
    pessoa1.altura = 185;
```

```
    PesoAltura pessoa1;  
    pessoa1.peso = 80;  
    pessoa1.altura = 185;
```

Comparando Java e C

```
System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");
```

Comparando Java e C

```
System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
```

Comparando Java e C

```
System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");
```

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
```

```
if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
else System.out.println("Altura abaixo da maxima.");
```

Comparando Java e C

```
System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");
```

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
```

```
if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
else System.out.println("Altura abaixo da maxima.");
```

```
if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");  
else printf("Altura abaixo da maxima.\n");
```

Código em C

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

Código em C

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

```
$ Peso: 80, Altura 185. Altura abaixo da maxima.
```

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```



alturaMaxima 225

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```

alturaMaxima

225

9742

args

9742

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```

alturaMaxima	225	9742		
args	9742	3408	peso	0
pessoa1	3408		altura	0

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```

alturaMaxima	225	9742		
args	9742	3408	peso	80
pessoa1	3408		altura	185

Uso de memória

Uso de memória em Java:

```
public class PesoAltura {  
    int peso; // peso em quilogramas  
    int altura; // altura em centímetros  
}
```

```
public class EstruturaSimples {  
    public static final int alturaMaxima = 225;  
    public static void main(String[] args) {  
        PesoAltura pessoa1 = new PesoAltura();  
        pessoa1.peso = 80;  
        pessoa1.altura = 185;  
        System.out.print("Peso: " + pessoa1.peso + ", Altura "+pessoa1.altura+". ");  
        if (pessoa1.altura>alturaMaxima) System.out.println("Altura acima da maxima.");  
        else System.out.println("Altura abaixo da maxima.");  
    }  
}
```

alturaMaxima	225	9742		
args	9742	3408	peso	80
pessoa1	3408		altura	185

\$ Peso: 80, Altura 185. Altura abaixo da maxima.

Uso de memória

Uso de memória em C:

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

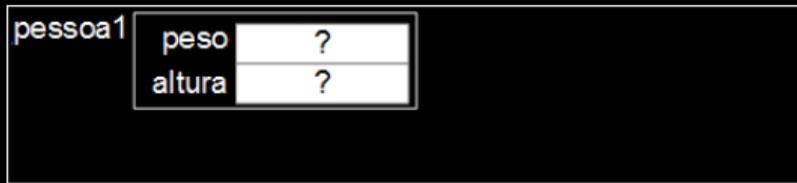
Uso de memória

Uso de memória em C:

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```



Uso de memória

Uso de memória em C:

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

pessoa1	peso	80
	altura	185

Uso de memória

Uso de memória em C:

```
#include <stdio.h>
#define alturaMaxima 225
```

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

```
int main() {
    PesoAltura pessoa1;
    pessoa1.peso = 80;
    pessoa1.altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
    if (pessoa1.altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

pessoa1	peso	80
	altura	185

\$ Peso: 80, Altura 185. Altura abaixo da maxima.

Uso de memória

Por que o uso de memória foi tão diferente nas duas soluções?

Uso de memória

Por que o uso de memória foi tão diferente nas duas soluções?

alturaMaxima	225	9742		
args	9742	3408	peso	80
peessoa1	3408		altura	185

Uso de memória

Por que o uso de memória foi tão diferente nas duas soluções?

alturaMaxima	225	9742	
args	9742	3408	peso 80
pessoa1	3408		altura 185

pessoa1	peso 80
	altura 185

Uso de memória

Por que o uso de memória foi tão diferente nas duas soluções?

alturaMaxima	225	9742	
args	9742	3408	peso 80
pessoa1	3408		altura 185

pessoa1	peso 80
	altura 185

Um dos motivos é que nossa implementação em C **não corresponde totalmente** à implementação em Java

Ponteiros e alocação de memória

Em C há uma distinção bastante explícita entre um tipo (ou uma estrutura) e um endereço:

int x; significa que *x* é uma variável do tipo **inteiro**

int y;* significa que *y* é uma variável do tipo **endereço para inteiro**

O asterisco (*) após a palavra *int* indica que estamos falando de um endereço para inteiro e não mais de um inteiro

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```



A variável x é inicializada com valor 25.

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```

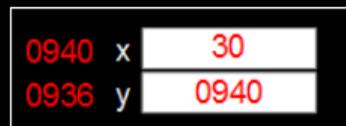
0940	x	25
0936	y	0940

A variável x é inicializada com valor 25.

A variável y recebe o endereço onde está a variável x.

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```



A variável x é inicializada com valor 25.

A variável y recebe o endereço onde está a variável x.

Coloca-se o valor 30 na posição de memória referenciada (apontada) pelo endereço armazenado em y.

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```

0940	x	30
0936	y	0940

\$ Valor atual de x: 30

A variável x é inicializada com valor 25.

A variável y recebe o endereço onde está a variável x.

Coloca-se o valor 30 na posição de memória referenciada (apontada) pelo endereço armazenado em y.

Ponteiros e alocação de memória

```
#include <stdio.h>
int main(){
    int x = 25;
    int* y = &x;
    *y = 30;
    printf("Valor atual de x: %i\n", x);
    return 0;
}
```

0940	x	30
0936	y	0940

\$ Valor atual de x: 30

A variável x é inicializada com valor 25.

A variável y recebe o endereço onde está a variável x.

Coloca-se o valor 30 na posição de memória referenciada (apontada) pelo endereço armazenado em y.

Ponteiros e alocação de memória

Em C há uma função para alocação de memória: **malloc** (*memory allocation*)

- recebe como parâmetro o **número de bytes** que se deseja alocar
- retorna o endereço inicial do bloco de bytes que foi alocado, porém esse retorno tem o tipo *void** (ponteiro para void)
- há um operador chamado **sizeof** que recebe como parâmetro um tipo (simples ou composto) e retorna a quantidade de bytes ocupada por esse tipo

Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```

Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```

Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```



Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```



Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```



Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```



Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```

0940	y	2200	2200	20
0936	z	4		

Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```

0940	y	2200	2200	20
0936	z	4		

Função malloc

```
#include <stdio.h>
#include <malloc.h>
int main(){
    int* y = (int*) malloc(sizeof(int));
    *y = 20;
    int z = sizeof(int);
    printf("*y=%i z=%i\n", *y, z);
    return 0;
}
```

Saída:

```
$ *y=20 z=4
```



Nossa segunda implementação

```
#include <stdio.h>  
#define alturaMaxima 225
```

Nossa segunda implementação

```
#include <stdio.h>  
#define alturaMaxima 225
```

```
#include <stdio.h>  
#include <malloc.h>  
#define alturaMaxima 225
```

Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225
```

```
#include <stdio.h>
#include <malloc.h>
#define alturaMaxima 225
```

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225
```

```
#include <stdio.h>
#include <malloc.h>
#define alturaMaxima 225
```

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

```
typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;
```

Nossa segunda implementação

```
int main() {
```

Nossa segunda implementação

```
int main() {
```

```
int main() {
```

Nossa segunda implementação

```
int main() {
```

```
int main() {
```

```
    PesoAltura pessoa1;  
    pessoa1.peso = 80;  
    pessoa1.altura = 185;
```

Nossa segunda implementação

```
int main() {
```

```
int main() {
```

```
    PesoAltura pessoa1;  
    pessoa1.peso = 80;  
    pessoa1.altura = 185;
```

```
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));  
    pessoa1->peso = 80;  
    pessoa1->altura = 185;
```

Nossa segunda implementação

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
```

Nossa segunda implementação

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);
```

```
printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
```

Nossa segunda implementação

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);  
printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);  
  
if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");  
else printf("Altura abaixo da maxima.\n");
```

Nossa segunda implementação

```
printf("Peso: %i, Altura %i. ", pessoa1.peso, pessoa1.altura);  
printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);  
  
if (pessoa1.altura>alturaMaxima) printf("Altura acima da maxima.\n");  
else printf("Altura abaixo da maxima.\n");  
  
if (pessoa1->altura>alturaMaxima) printf("Altura acima da maxima.\n");  
else printf("Altura abaixo da maxima.\n");
```

Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));
    pessoa1->peso = 80;
    pessoa1->altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
    if (pessoa1->altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));
    pessoa1->peso = 80;
    pessoa1->altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
    if (pessoa1->altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

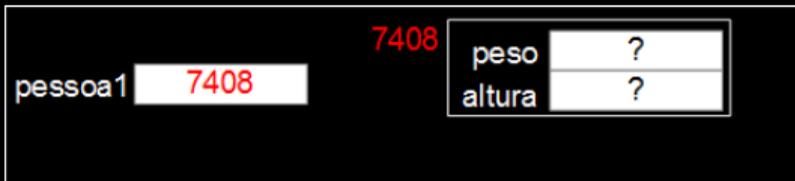
pessoa1

Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));
    pessoa1->peso = 80;
    pessoa1->altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
    if (pessoa1->altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

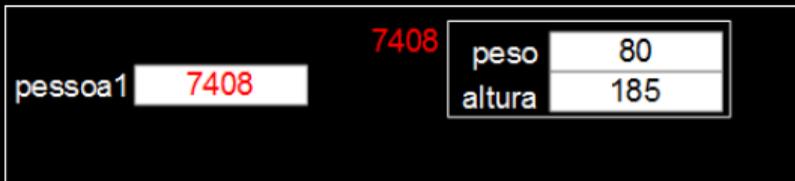


Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));
    pessoa1->peso = 80;
    pessoa1->altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
    if (pessoa1->altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```

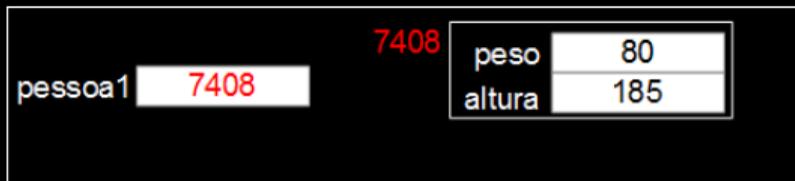


Nossa segunda implementação

```
#include <stdio.h>
#define alturaMaxima 225

typedef struct{
    int peso; // peso em quilogramas
    int altura; // altura em centímetros
} PesoAltura;

int main() {
    PesoAltura* pessoa1 = (PesoAltura*) malloc(sizeof(PesoAltura));
    pessoa1->peso = 80;
    pessoa1->altura = 185;
    printf("Peso: %i, Altura %i. ", pessoa1->peso, pessoa1->altura);
    if (pessoa1->altura > alturaMaxima) printf("Altura acima da maxima.\n");
    else printf("Altura abaixo da maxima.\n");
    return 0;
}
```



\$ Peso: 80, Altura 185. Altura abaixo da maxima.

AULA 02

ESTRUTURA DE DADOS

Criação de uma primeira estrutura

Norton T. Roman & Luciano A. Digiampietri