

# Dominando Estrutura de Dados 1

## Os Temidos Ponteiros

prof. Samuel Martins (Samuka)  
@xavecoding @hisamuka



# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

```
int a = 10;  
int b, c;  
  
b = 20;  
c = a + b;
```

## Memória Ram

|      |      |
|------|------|
| 8888 | #### |
| ...  |      |
| 3028 | #### |
| 3024 | #### |
| 3020 | #### |

# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

```
int a = 10;  
int b, c;  
  
b = 20;  
c = a + b;
```

## Memória Ram

|      |      |     |
|------|------|-----|
| 8888 | #### |     |
| ...  |      |     |
| 3028 | #### |     |
| 3024 | #### |     |
| 3020 | 10   | a ← |

`&a = 3020; a = 10` ←

# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

```
int a = 10;  
int b, c;  
  
b = 20;  
c = a + b;
```

## Memória Ram

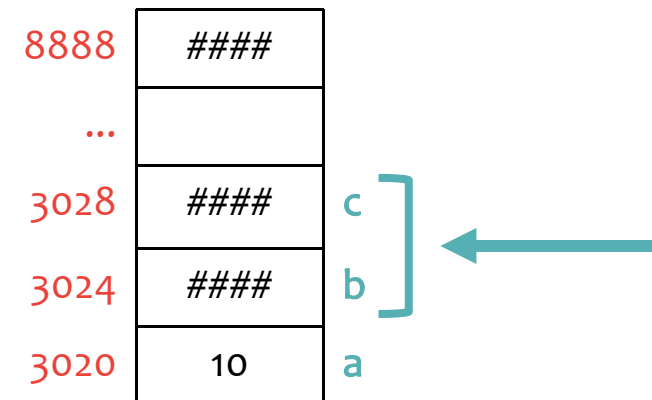


Diagram illustrating pointer addresses and current values for variables:

|                     |      |
|---------------------|------|
| &a = 3020; a = 10   |      |
| &b = 3024; b = #### | lixo |
| &c = 3028; c = #### | lixo |

A large teal arrow points from the right towards the memory blocks for variables b and c.

# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

```
int a = 10;  
int b, c;  
  
b = 20;  
c = a + b;
```

## Memória Ram

|      |      |     |
|------|------|-----|
| 8888 | #### |     |
| ...  |      |     |
| 3028 | 30   | c   |
| 3024 | 20   | b ← |
| 3020 | 10   | a   |

```
&a = 3020; a = 10  
&b = 3024; b = 20  
&c = 3028; c = ####
```

lixo ←

# Variáveis e a Memória

Em linguagem C, cada **variável** está associada a:

- 1) Um nome;
- 2) Um tipo;
- 3) Um valor;
- 4) Um endereço.

```
int a = 10;  
int b, c;  
  
b = 20;  
c = a + b;
```

## Memória Ram

|      |      |     |
|------|------|-----|
| 8888 | #### |     |
| ...  |      |     |
| 3028 | 30   | c ← |
| 3024 | 20   | b   |
| 3020 | 10   | a   |

```
&a = 3020; a = 10  
&b = 3024; b = 20  
&c = 3028; c = 30
```

# Let's code!

```
1  #include <stdio.h>
2
3  int main() {
4      int a = 10;
5      int b, c;
6
7      printf("&a = %p, a = %d\n", &a, a);
8      printf("&b = %p, b = %d\n", &b, b);
9      printf("&c = %p, c = %d\n\n", &c, c);
10
11     b = 20;
12     c = a + b;
13
14     printf("&a = %p, a = %d\n", &a, a);
15     printf("&b = %p, b = %d\n", &b, b);
16     printf("&c = %p, c = %d\n\n", &c, c);
17
18     return 0;
19 }
20
```

**&var:** endereço da variável **var**

**%p:** flag para imprimir **endereços**



# Let's code!

```
&a = 0x7ffee50a4978, a = 10  
&b = 0x7ffee50a4974, b = 32766  
&c = 0x7ffee50a4970, c = -452310640  
  
&a = 0x7ffee50a4978, a = 10  
&b = 0x7ffee50a4974, b = 20  
&c = 0x7ffee50a4970, c = 30
```

lixo: qualquer valor

Como **a**, **b**, **c** são **int**, seus endereços são separados por **4 bytes** na memória.

Dependendo do sistema operacional, a atribuição de endereços de memória pode ser em **ordem decrescente**

|      |      |   |
|------|------|---|
| 8888 | #### |   |
| ...  |      |   |
| 4970 | 30   | c |
| 4974 | 20   | b |
| 4978 | 10   | a |

# Ponteiros

# Ponteiros

*endereço*

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

*nome da variável*

10  
*valor da variável*

&messi = Avenida do Messi; messi = 10

# Ponteiros

*endereço*

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

*nome da variável*

10  
*valor da variável*

&messi = Avenida do Messi; messi = 10

*endereço*

Rua do Jaiminho

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona



Avenida do Messi

*valor da variável*

jaiminho

*nome da variável*

# Ponteiros

*endereço*

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

*nome da variável*

10  
*valor da variável*

&messi = Avenida do Messi; messi = 10

*endereço*

Rua do Jaiminho

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona



Avenida do Messi

*valor da variável*

jaiminho

*nome da variável*

jaiminho = &messi;

*endereço da variável* &jaiminho = Rua do Jaiminho

*valor da variável* jaiminho = Avenida do Messi

*conteúdo de um endereço* \*(jaiminho) = \*(Avenida do Messi) = 10

# Ponteiros

endereço

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

nome da variável

10  
valor da variável

&messi = Avenida do Messi; messi = 10

endereço

Rua do Zizu

Tipo: Jogador do  
Real Madrid



zidane

nome da variável

5.50  
valor da variável

&zidane = Rua do Zizu; zidane = 5.50

endereço

Rua do Jaiminho

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona



Avenida do Messi

valor da variável

jaiminho

nome da variável

jaiminho = &messi;

endereço da variável &jaiminho = Rua do Jaiminho

valor da variável jaiminho = Avenida do Messi

conteúdo de um endereço \*(jaiminho) = \*(Avenida do Messi) = 10

# Ponteiros

endereço

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

nome da variável

10  
valor da variável

&messi = Avenida do Messi; messi = 10

endereço

Rua do Zizu

Tipo: Jogador do  
Real Madrid



zidane

nome da variável

5.50  
valor da variável

&zidane = Rua do Zizu; zidane = 5.50

endereço

Rua do Jaiminho

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona



Avenida do Messi

valor da variável

jaiminho

nome da variável

jaiminho = &messi;

endereço da variável &jaiminho = Rua do Jaiminho

valor da variável jaiminho = Avenida do Messi

conteúdo de um endereço \*(jaiminho) = \*(Avenida do Messi) = 10

endereço

Rua do Mario

Tipo: Carteiro (ponteiro) de  
Jogador do Real Madrid



Rua do Zizu

valor da variável

mario\_carteiro

nome da variável

# Ponteiros

endereço

Avenida do Messi

Tipo: Jogador do  
Barcelona



messi

nome da variável

10  
valor da variável

&messi = Avenida do Messi; messi = 10

endereço

Rua do Zizu

Tipo: Jogador do  
Real Madrid



zidane

nome da variável

5.50  
valor da variável

&zidane = Rua do Zizu; zidane = 5.50

endereço

Rua do Jaiminho

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona



Avenida do Messi

valor da variável

jaiminho

nome da variável

jaiminho = &messi;  
endereço da variável &jaiminho = Rua do Jaiminho  
valor da variável jaiminho = Avenida do Messi  
conteúdo de um endereço \*(jaiminho) = \*(Avenida do Messi) = 10

endereço

Rua do Mario

Tipo: Carteiro (ponteiro) de  
Jogador do Real Madrid



Rua do Zizu

valor da variável

mario\_carteiro

nome da variável

mario\_carteiro = &zidane;  
endereço da variável &mario\_carteiro = Rua do Mario  
valor da variável mario\_carteiro = Rua do Zizu  
conteúdo de um endereço \*(mario\_carteiro) = \*(Rua do Zizu) = 5.50



# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;  
  
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

## Memória Ram

|      |  |
|------|--|
| 8888 |  |
| ...  |  |
| 5012 |  |
| 5004 |  |
| 5000 |  |

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;  
  
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

`&a = 5000; a = 10;`

## Memória Ram

|      |      |
|------|------|
| 8888 | #### |
| ...  |      |
| 5012 | #### |
| 5004 | #### |
| 5000 | 10   |

**a** ←

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;  
  
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

```
&a = 5000; a = 10;  
&p1 = 5004; p1 = NULL;  
*p1 = ---;
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | #### |      |
| 5004 | NULL | p1 ← |
| 5000 | 10   | a    |

Um ponteiro pode ter o valor especial **NULL** que **não é o endereço de lugar algum**.  
A constante **NULL** está definida em `stdlib.h` e seu valor é **0** na maioria dos computadores.

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;
```

```
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

```
&a = 5000; a = 10;  
  
&p1 = 5004; p1 = NULL;  
*p1 = ---;  
  
&p2 = 5012; p2 = ##;  
*p2 = ---;
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | #### | p2 ← |
| 5004 | NULL | p1   |
| 5000 | 10   | a    |

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;
int *p1 = NULL;
int *p2;

p1 = &a;
p2 = p1;
*p2 = 4;
```

```
&a = 5000; a = 10;

&p1 = 5004; p1 = 5000;
*p1 = *(5000) = a = 10;

&p2 = 5012; p2 = ####;
*p2 = ---;
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | #### | p2   |
| 5004 | 5000 | p1 ← |
| 5000 | 10   | a    |

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;  
  
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

```
&a = 5000; a = 10;  
  
&p1 = 5004; p1 = 5000;  
*p1 = *(5000) = a = 10;  
  
&p2 = 5012; p2 = 5000;  
*p2 = *(5000) = a = 10;
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | 5000 | p2 ← |
| 5004 | 5000 | p1   |
| 5000 | 10   | a    |

# Ponteiros

Um **ponteiro** nada mais é do que uma variável que guarda o **endereço** de uma **outra variável**;

```
int a = 10;  
int *p1 = NULL;  
int *p2;  
  
p1 = &a;  
p2 = p1;  
*p2 = 4;
```

```
&a = 5000; a = 4;  
  
&p1 = 5004; p1 = 5000;  
*p1 = *(5000) = a = 4;  
  
&p2 = 5012; p2 = 5000;  
*p2 = *(5000) = a = 4;
```

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | 5000 | p2 |
| 5004 | 5000 | p1 |
| 5000 | 4    | a  |

# Let's code!

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      int a = 10;
6      int *p1 = NULL;
7      int *p2;
8
9      p1 = &a;
10     p2 = p1;
11     *p2 = 4;
12
13     printf("&a = %p, a = %d\n", &a, a);
14     printf("&p1 = %p, p1 = %p, *p1 = %d\n", &p1, p1, *p1);
15     printf("&p2 = %p, p2 = %p, *p2 = %d\n", &p2, p2, *p2);
16
17     return 0;
18 }
```



# Let's code!

```
&a = 0x7ffe5b17c944, a = 4
&p1 = 0x7ffe5b17c948, p1 = 0x7ffe5b17c944, *p1 = 4
&p2 = 0x7ffe5b17c950, p2 = 0x7ffe5b17c944, *p2 = 4
```

&var: endereço da variável var

&var: conteúdo da variável cujo endereço está armazenado em var

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| c950 | c944 | p2 |
| c948 | c944 | p1 |
| c944 | 4    | a  |

```
1  #include <stdio.h>
2
3  int main() {
4      int a;
5
6      printf("sizeof(a) = %ld bytes\n", sizeof(a));
7      printf("sizeof(int) = %ld bytes\n", sizeof(int));
8      printf("sizeof(short) = %ld bytes\n", sizeof(short));
9      printf("sizeof(long) = %ld bytes\n", sizeof(long));
10     printf("sizeof(float) = %ld bytes\n", sizeof(float));
11     printf("sizeof(double) = %ld bytes\n\n", sizeof(double));
12
13     printf("sizeof(void *) = %ld bytes\n", sizeof(void *));
14     printf("sizeof(int *) = %ld bytes\n", sizeof(int *));
15     printf("sizeof(int **) = %ld bytes\n", sizeof(int **));
16     printf("sizeof(int ***) = %ld bytes\n", sizeof(int ***));
17     printf("sizeof(float *) = %ld bytes\n", sizeof(float *));
18
19     return 0;
20 }
```

# Ponteiro de Ponteiros

Tipo: Jogador do  
Barcelona

endereço

Avenida do Messi



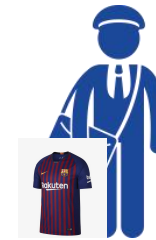
messi

nome da variável

10  
valor da variável

`&messi = Avenida do Messi; messi = 10`

endereço  
Rua do Jaiminho



jaiminho

nome da variável

Avenida do Messi

valor da variável

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona

`jaiminho = &messi;`  
*endereço da variável* `&jaiminho = Rua do Jaiminho`  
*valor da variável* `jaiminho = Avenida do Messi`  
*conteúdo de um endereço* `*(jaiminho) = *(Avenida do Messi) = 10`

# Ponteiro de Ponteiros

Tipo: Jogador do Barcelona

endereço  
Avenida do Messi



messi

nome da variável

10  
valor da variável

`&messi = Avenida do Messi; messi = 10`

endereço  
Rua do Sebastião



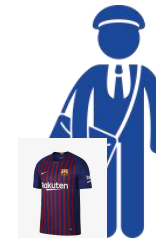
sebastiao

nome da variável

Rua do Jaiminho  
valor da variável

Tipo: Carteiro (ponteiro) de um  
Carteiro (ponteiro) de Jogador do Barcelona

endereço  
Rua do Jaiminho



jaiminho

nome da variável

Avenida do Messi

valor da variável

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona

`jaiminho = &messi;`  
*endereço da variável* `&jaiminho = Rua do Jaiminho`  
*valor da variável* `jaiminho = Avenida do Messi`  
*conteúdo de um endereço* `*(jaiminho) = *(Avenida do Messi) = 10`

# Ponteiro de Ponteiros

Tipo: Jogador do  
Barcelona

endereço  
Avenida do Messi



messi

nome da variável

10  
valor da variável

`&messi = Avenida do Messi; messi = 10`

endereço  
Rua do Sebastião



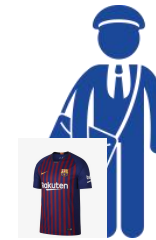
sebastiao

nome da variável

Rua do Jaiminho  
valor da variável

Tipo: Carteiro (ponteiro) de um  
Carteiro (ponteiro) de Jogador do Barcelona

endereço  
Rua do Jaiminho



jaiminho

nome da variável

Avenida do Messi

valor da variável

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona

`jaiminho = &messi;`  
endereço da variável `&jaiminho = Rua do Jaiminho`  
valor da variável `jaiminho = Avenida do Messi`  
conteúdo de um endereço `*(jaiminho) = *(Avenida do Messi) = 10`

`sebastiao = &jaiminho;`  
endereço da variável `&sebastiao = Rua do Sebastião;`  
valor da variável `sebastiao = Rua do Jaiminho;`  
conteúdo de um endereço `*(sebastiao) = *(Rua do Jaiminho) = Avenida do Messi`

# Ponteiro de Ponteiros

Tipo: Jogador do Barcelona

endereço  
Avenida do Messi



messi

nome da variável

10  
valor da variável

`&messi = Avenida do Messi; messi = 10`

endereço  
Rua do Sebastião



sebastiao

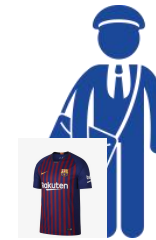
nome da variável

Rua do Jaiminho

valor da variável

Tipo: Carteiro (ponteiro) de um  
Carteiro (ponteiro) de Jogador do Barcelona

endereço  
Rua do Jaiminho



jaiminho

nome da variável

Avenida do Messi

valor da variável

Tipo: Carteiro (ponteiro) de  
Jogador do Barcelona

```
jaiminho = &messi;  
endereço da variável &jaiminho = Rua do Jaiminho  
valor da variável jaiminho = Avenida do Messi  
conteúdo de um endereço *(jaiminho) = *(Avenida do Messi) = 10
```

```
sebastiao = &jaiminho;  
endereço da variável &sebastiao = Rua do Sebastião;  
valor da variável sebastiao = Rua do Jaiminho;  
conteúdo de um endereço *(sebastiao) = *(Rua do Jaiminho) = Avenida do Messi  
conteúdo de um endereço **(sebastiao) = **(Rua do Jaiminho) = *(Avenida do Messi) = 10
```

# Ponteiro de Ponteiros

```
int a = 10;  
int *p1 = &a;  
int **p2 = &p1;  
  
**p2 = 99;
```

## Memória Ram

|      |  |
|------|--|
| 8888 |  |
| ...  |  |
| 5012 |  |
| 5004 |  |
| 5000 |  |

# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

## Memória Ram

|      |  |
|------|--|
| 8888 |  |
| ...  |  |
| 5012 |  |
| 5004 |  |
| 5000 |  |



# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("&fp = %f\n", *&fp);
printf("**&fp = %f\n", **&fp);
```

## Memória Ram

|      |      |     |
|------|------|-----|
| 8888 | #### |     |
| ...  |      |     |
| 5012 | #### |     |
| 5004 | #### |     |
| 5000 | 2.5  | z ← |

# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | #### |      |
| 5004 | #### | fp ← |
| 5000 | 2.5  | z    |

# Ponteiro de Ponteiros

```
float z = 2.5;  
float *fp;
```

```
fp = &z;
```

```
printf("&z = %f\n", *&z);  
printf("*fp = %f\n", *fp);  
printf("**&fp = %f\n", **&fp);
```

## Memória Ram

|      |      |      |
|------|------|------|
| 8888 | #### |      |
| ...  |      |      |
| 5012 | #### |      |
| 5004 | 5000 | fp ← |
| 5000 | 2.5  | z    |

# Ponteiro de Ponteiros

```
float z = 2.5;  
float *fp;
```

```
fp = &z;
```

```
printf("&z = %f\n", *&z);  
printf("*fp = %f\n", *fp);  
printf("**&fp = %f\n", **&fp);
```

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |

# Ponteiro de Ponteiros

```
float z = 2.5;  
float *fp;
```

```
fp = &z;
```

```
printf("&z = %f\n", *&z);  
printf("*fp = %f\n", *fp);  
printf("**&fp = %f\n", **&fp);
```

$*\&z = *(5000) = z = 2.5$

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |

# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

$*\&z = *(5000) = z = 2.5$

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |

# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

$*\&z = *(5000) = z = 2.5$

$*fp = *(5000) = z = 2.5$

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |

# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

$*\&z = *(5000) = z = 2.5$

$*fp = *(5000) = z = 2.5$

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |



# Ponteiro de Ponteiros

```
float z = 2.5;
float *fp;

fp = &z;

printf("&z = %f\n", *&z);
printf("*fp = %f\n", *fp);
printf("**&fp = %f\n", **&fp);
```

$*\&z = *(5000) = z = 2.5$

$*fp = *(5000) = z = 2.5$

$**\&fp = **(\&fp) = *(5004) = *(5000) = z = 2.5$

## Memória Ram

|      |      |    |
|------|------|----|
| 8888 | #### |    |
| ...  |      |    |
| 5012 | #### |    |
| 5004 | 5000 | fp |
| 5000 | 2.5  | z  |

# Exercício

1) Escreva os valores das variáveis para cada instrução do programa abaixo. Qual a saída do programa?

```
int main(int argc, char *argv[]) {  
    int a, b, *p1, *p2;  
  
    a = 4;  
    b = 3;  
    p1 = &a;  
    p2 = p1;  
    *p2 = *p1 + 3; b = b * (*p1); (*p2)++;  
    p1 = &b;  
  
    printf("%d %d\n", *p1, *p2);  
    printf("%d %d\n", a, b);  
}
```

Teste de Mesa

| a | b | p1 | *p1 | p2 | *p2 |
|---|---|----|-----|----|-----|
|---|---|----|-----|----|-----|

|  |
|--|
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

Sooo

# Dominando Estrutura de Dados 1

## Os Temidos Ponteiros

Prof. Samuel Martins (Samuka)  
@xavecoding @hisamuka

