

## Apontadores

1. Considere as seguintes declarações de variáveis:

**int \*\*V, W[8], N, B;**

Considere também que **sizeof(int) = 4** e **sizeof(\*int) = 8**.

**a)** Preencha o esquema dado à esquerda considerando a execução do bloco de instruções em linguagem C dado à direita.

(esquema de um bloco de memória)		
100500	<div><div>.</div><div>.</div><div>.</div><div>110160</div><div>.</div><div>.</div><div>.</div></div>	W
100700	<div><div>.</div><div>.</div><div>.</div><div>110172</div><div>.</div><div>.</div><div>.</div></div>	
100800		
		N
	<div><div>.</div><div>.</div><div>.</div></div>	
110160	<div><div>-100</div></div>	
110164		B
110168		
110172		
110176		
110180		
110184		
110188		
110192	<div><div>-200</div></div>	
	<div><div>.</div><div>.</div><div>.</div></div>	
110500	<div><div>100700</div></div>	
	<div><div>.</div><div>.</div><div>.</div></div>	

Bloco de instruções em linguagem C:	
<div><div>...</div><div>**V = 900;</div><div>for (N = 2; N &lt; 5; N++)</div><div>    (*V)[N-1] = N * 100;</div><div>*(W + 1) = 600;</div><div>*(V - 1) = 700;</div><div>W[7] = 100;</div><div>...</div></div>	
<div><div>b)</div><div>Com os valores obtidos com a execução do bloco de instruções dado, determine os valores das seguintes expressões:</div></div>	
EXPRESSÃO	VALOR
**V + 3	
*V + 3	
V + 3	
(*V)[4]	
*(V - 3)	
**V + 4	
&(*V)[2]	
W - 3	
W[3]	
&(W[2])	
W[8]	
*(W + 5)	
*W - 2	

2. Considere as seguintes declarações de variáveis:

**int V[7], \*\*W, N;**

Considere também que **sizeof(int) = 4** e **sizeof(\*int) = 8**.

a) Preencha o esquema dado à esquerda, considerando a execução do bloco de instruções em linguagem C dado à direita.

**Esquema de um bloco de memória:**

	.	
	.	
100500	<b>110180</b>	
	.	
	.	
100700	<b>110164</b>	<b>V</b>
	.	
	.	
100800		<b>N</b>
	.	
	.	
110160	<b>100</b>	
110164		
110168		
110172		
110176		
110180		
110184		
110188		
110192	<b>200</b>	
	.	
	.	
110500	<b>100500</b>	<b>W</b>
	.	
	.	
	.	

**Bloco de instruções em linguagem C:**

```

...
*V = 900;
N = 1;
do {
    V[N] = N * 500;
    N = N + 1;
} while (N < 4);
>(*W + 1) = 600;
*(V + 6) = 400;
(*W)[0] = 50;
...

```

b) Usando os valores obtidos com a execução do bloco de instruções dado, determine os valores das seguintes expressões:

EXPRESSÃO	VALOR
V + 4	
*(V + 4)	
*V + 4	
V[4]	
&V[4]	
&(*V + 4)	
*V - 4	
**W	
**W - 2	
(*W)[2]	
&(*W)[2]	
__(*W + 3)	
W - 2	

**3.** Um ponteiro pode ser manipulado como sendo um vetor (array de 1 dimensão).

```
#include <stdio.h>

main ( ) {
    int v[5] = { 10, 20, 30, 40, 50 };
    int p, i;
    p = v;
    for (i = 1; i < 5; i++)
        printf ("%d ", p[i]);
}
```

- a)** Escreva o programa anterior no computador e, identifique e corrija os erros no código.
- b)** Acrescente ao código anterior uma instrução para escrever o endereço do 1º elemento do vetor, usando as seguintes 4 formas diferentes para o fazer: `&v[0]`, `&p[0]`, `v`, `p`.
- c)** Acrescente ao código anterior uma instrução para escrever o endereço do 2º elemento do vetor, usando as seguintes 4 formas diferentes para o fazer: `&v[1]`, `&p[1]`, `v+1`, `p+1`. Quantos bytes ocupa cada valor do vetor?
- d)** Acrescente ao código anterior uma instrução para escrever o valor do 1º elemento do vetor, usando as seguintes 4 formas diferentes para o fazer: `v[0]`, `p[0]`, `*v`, `*p`.
- e)** Acrescente ao código anterior uma instrução para escrever o valor do 2º elemento do vetor, usando as seguintes 4 formas diferentes para o fazer: `v[1]`, `p[1]`, `*(v+1)`, `*(p+1)`.
- f)** Acrescente ao código anterior a instrução que se segue e compare os resultados com os obtidos na alínea anterior: **`printf("%d %d %d %d\n", v[1], p[1], *v+1, *p+1);`**

**4.** Faça o mesmo com o programa anterior mas considerando que o vetor usado é de valores reais e efetue as operações que constam nas alíneas b) a f) para este caso.

**5.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){
    int y, *p, x;
    y = 0;
    p = &y;
    x = *p;
    x = 4;
    (*p)++;
    x--;
    (*p) += x;
    printf ("y = %d\n", y);
}
```

**6.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){  
    int  *pont, cont, val;  
    cont = 100;  
    pont = &cont;  
    val = *pont;  
    printf ("%d", val);  
}
```

**7.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){  
    int  x, y, *px, *py;  
    x = 5;  
    px = &x;  
    py = px;  
    y = *py;  
    printf ("%d %d", x, y);  
}
```

**8.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){  
    int  x, y, *px, **py;  
    x = 5;  
    px = &x;  
    py = &px;  
    y = **py;  
    printf ("%d %d", x, y);  
}
```

**9.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){  
    char  a, b, *p;  
    b = 'c';  
    p = &a;  
    *p = b;  
    printf ("%c", a);  
}
```

10. Considere as seguintes declarações de variáveis:

```
int W[5], **V, M, k;
```

e que `sizeof(int) = 2`.

Considere o seguinte bloco de instruções em linguagem C:

```
M = 5;
for (k = 1; k < M; k++)
    W[k] = 5*k + 10;
*W = 40;
*(W-1) = 50;
```

- a) Preencha o quadro à direita tendo em conta a execução do bloco anterior.
- b) Usando os valores obtidos com a execução do bloco de instruções anterior, determine os valores de cada uma das expressões que constam na tabela seguinte:

Expressão	Valor
<code>*W - 2</code>	
<code>W - 2</code>	
<code>&amp;W[2]</code>	
<code>&amp;(*W)</code>	
<code>**V - 2</code>	
<code>*(**V - 2)</code>	

(esquema de um bloco de memória)

	...	
2000	4000	V
	...	
3000		M
	...	
4000	4090	
	...	
4080		
4082		
4084		
4086		
4088		
4090		
4092		
	...	
5000	4084	W
	...	

11. Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
main (){
    int x, y, *px, *py;
    printf ("Digite um valor: ");
    scanf ("%d", &x);
    px = &x;
    y = *px;
    printf ("digitou = %d e y = %d\n", x, y);
    *px = 8;
    printf ("valor mudou para %d\n", x);
}
```

**12.** Indicar e justificar (através de simulação) o que faz o seguinte programa:

```
#include <stdio.h>

main (){
    int i, k, *pi, *pk;
    char a;
    i = 2;
    k = 0;
    puts ("Qual será o valor de k? ");
    pk = &k;
    pi = &i;
    *pk = i;
    printf ("para *pk = I, temos k = %d\n", k);
    k = *pi;
    printf ("para k = *pi, temos k = %d\n", k);
    scanf ("%c", &a);
}
```

**13.** Escreva um programa que some dois valores inteiros. Para tal, deverá implementar e utilizar os subprogramas seguintes:

- a)** int \*soma4 (int a, int b);
- b)** int \*soma5 (int \*a, int \*b).

**14.** Verificar que o subprograma **soma3**, implementada da forma que segue, está incorreta. Explicar o comportamento indesejado utilizando, se necessário, exemplos de execução.

```
int *soma3 (int a, int b){
    int temp ;
    temp = a+b;
    return (&temp);
}
```

**15.** Escreva um programa que leia dois valores para duas variáveis de tipo **real** e troque os seus conteúdos. Deverá implementar e utilizar um subprograma com o seguinte cabeçalho:

```
void trocar_float (float *x1, float *x2)
```

**16.** Escreva um programa que calcule o maior e o menor valor de um array de 1 dimensão de 10 valores inteiros. Deverá implementar e utilizar um subprograma com o seguinte cabeçalho:

```
void maior_menor (int X[], int *maior, int *menor)
```

**17.** Escreva um programa que calcule o maior elemento, e o respetivo índice/posição, de um vetor de 10 valores reais. Implementar e utilizar subprograma com o seguinte cabeçalho:

```
int maior_indicemaior (float X[], float *maior)
```

- 18.** Escreva um programa que calcule a soma e a média de um array de 1 dimensão de inteiros. Deverá implementar e utilizar um subprograma com o seguinte cabeçalho:

**void soma\_media (int X[], int \*soma, float \*media)**

- 19.** Construa um programa que:

- leia 1 array de 1 dimensão de números inteiros, X, de tamanho N
- crie um segundo array Y apenas com os números positivos do vetor X.
- escreva os arrays X e Y.

Para tal, use subprogramas para as seguintes ações:

- a)** ler um array V de números inteiros de tamanho N (usando memória estática)
- b)** escrever um array V de números inteiros de tamanho N
- c)** dado um array V de números inteiros de tamanho N, construa um array W com todos os números positivos do array V.

- 20.** Escreva um programa que calcule o maior e o menor elementos de um array de 2 dimensões de reais. Deverá implementar e utilizar um subprograma com o seguinte cabeçalho:

**void maiorMenor (float X[][], int \*maior, int \*menor)**