**GABRIEL GREGÓRIO SANTOS MARQUES**

**SEGUNDO PERÍODO**

**ATIVIDADE DO PROFESSOR PAULO RICARDO NOÉ**

**Exercício 1:**

|  |  |  |
| --- | --- | --- |
| a) x= 10 | i) &x = 6DFEEC | q) &ppx = 6DFEE4 |
| b) \*py = 5,9 | j) py+1 = 6DFEE0 | r) &ppy = 6DFED8 |
| c) px = 6DFEEC | k) \*px-1 = 9 | s) &px = 6DFEE8 |
| d) &y = 6DFEE0 | l) \*\*ppy = 5,9 | t) &py = 6DFEDC |
| e) \*px = 10 | m) &ppy = 6DFED8 | u) ppx = 6DFEE8 |
| f) y = 5,9 | n) \*&px = 6DFEEC | v) ppy = 6DFEDC |
| g)\*ppx = 6DFEEC | o) (\*\*ppx)+1 = 11 |  |
| h) py = 6DFEE0 | p) px+1 = 6DFEF0 |  |

**Exercício 2:**

* Um vetor vet é alocado com espaço para 10 inteiros.
* Se a alocação falhar, exibe uma mensagem de erro e termina o programa.
* O vetor vet é preenchido com números ímpares: {1, 3, 5, 7, 9, 11, 13, 15, 17, 19}.
* Esses valores são exibidos na tela.
* ptr1 aponta para o início de vet. O primeiro valor de vet é alterado de 1 para 445.
* ptr1 avança 5 posições e aponta para vet[5]. O valor de vet[5] é alterado para 854.
* ptr2 aponta para vet[8]. Ele recua uma posição e decrementa o valor de vet[7], mudando de 15 para 14.
* O valor de vet[3] é alterado para 186.
* ptr3 aponta para ptr2. \*ptr3 avança para vet[8] e altera seu valor para 203.
* ptr2 recua duas posições e altera o valor de vet[6] para 204.
* \*ptr3 recua duas posições e o valor de vet[6] é incrementado de 204 para 205.
* Os valores atualizados do vetor vet são exibidos:
* 445 3 5 186 9 854 205 14 203 19

**Exercício 3:**

**1º Código fonte:**

char x;  
int i;  
x = 'c'; // x = 99 ('c' em ASCII)  
i = 5;  
x = x + i; // x = 99 + 5 = 104 ('h' em ASCII)  
printf("%c", x); // Imprime 'h'

Resultado: O caractere 'h' será impresso.

**2º Código fonte:**

int a, \*b, \*c;  
a = 20;  
b = &a; // b aponta para a  
c = b; // c aponta para a (mesmo endereço de b)  
a = (\*b + \*c) % 7; // a = (20 + 20) % 7 = 40 % 7 = 5  
printf("%d", a); // Imprime 5

Resultado: O valor 5 será impresso.

**3º Código fonte:**

int a, b, c, \*d;  
a = 2;  
b = 3;  
d = &a; // d aponta para a  
c = a \* (\*d) + b; // c = 2 \* 2 + 3 = 4 + 3 = 7  
printf("%d", c); // Imprime 7

Resultado: O valor 7 será impresso.

**4º Código fonte:**

int x, \*p, \*\*pp;  
x = 1;  
p = &x; // p aponta para x  
pp = &p; // pp aponta para p  
for(int i=0; i<3; i++)  
 x = \*p + \*\*pp; // x = 1 + 1 = 2, depois x = 2 + 2 = 4, e depois x = 4 + 4 = 8  
printf ("%d", \*\*pp); // \*\*pp é o valor de x, que agora é 8

Resultado: O valor 8 será impresso.

**5º Código fonte:**

int x, \*p, \*\*pp;  
x = 5;  
p = &x; // p aponta para x  
pp = &p; // pp aponta para p  
\*\*pp = \*p + 1; // \*\*pp é x, logo x = \*p + 1 = 5 + 1 = 6  
printf("%X", pp); // Imprime o valor hexadecimal do endereço de pp

Resultado: O valor hexadecimal do endereço de pp será impresso (o endereço de p, que é 6356744). Portanto, será impresso 6356744 em hexadecimal.

**6º Código fonte:**

int a[5] = {10, 20, 30, 40, 50};  
int \*b, \*\*c;  
b = a; // b aponta para o primeiro elemento de a  
c = &b; // c aponta para b  
\*c = \*c + 3; // \*c = b, e \*c + 3 faz b apontar para a[3], ou seja, b agora aponta para 40  
printf("%d", \*b); // Imprime o valor apontado por b, que é 40

Resultado: O valor 40 será impresso.

**7º Código fonte:**

int a[5] = {10, 20, 30, 40, 50};  
int \*b, \*\*c;  
b = a; // b aponta para o primeiro elemento de a  
c = &b; // c aponta para b  
\*c = \*c + 3; // b passa a apontar para a[3], ou seja, 40  
printf("%d", \*(b + 1)); // b aponta para a[3], logo \*(b + 1) será a[4], que é 50

Resultado: O valor 50 será impresso.

**8º Código fonte:**

int a[5] = {10, 20, 30, 40, 50};  
int \*b, \*\*c;  
b = (a+3); // b aponta para a[3], ou seja, 40  
b--; // b agora aponta para a[2], ou seja, 30  
printf("%d", \*b); // Imprime o valor apontado por b, que é 30

Resultado: O valor 30 será impresso.

**9º Código fonte:**

int i, \*vet, \*ptr;  
vet = (int\*) malloc(sizeof(int)\*10); // Aloca espaço para 10 inteiros  
for(i = 0; i < 10; i++)  
 vet[i] = i \* 2 + 1; // Preenche vet com valores ímpares: {1, 3, 5, 7, 9, 11, 13, 15, 17, 19}  
ptr = vet; // ptr aponta para o início de vet  
printf("%d", \*(ptr + 3)); // \*(ptr + 3) será vet[3], que é 7

Resultado: O valor 7 será impresso.