**JEAN ALVES ROCHA**

**RA: 2313057**

**EXERCÍCIOS**

Para os próximos exercícios, considere que int e float ocupam 4 bytes; Double ocupa 8 bytes e char ocupa 1 byte. Ainda, suponha que o programa será compilado e executado em um sistema de 32 bits, portanto as variáveis ponteiros sempre ocupam 4 bytes (em um sistema 64 bits, os ponteiros ocupam 8 bytes). Considere, também, que a alocação de memória é feita sequencialmente, em ordem decrescente.

1. Preencha a tabela abaixo, conforme o programa for executado e escreva quais saídas serão apresentadas na tela. Considere que a primeira variável será endereçada em 0055FF10.

Float a,b;

    float \*c,\*d,\*e;

    a=40;

    b=20;

    c=&a;

    d=&b;

    printf("\n%p%f",&a,a);

    printf("\n%p%f",&b,b);

    printf("\n%p %p %f",&c,c,\*c);

    printf("\n%p%p %f",&d,d,\*d);

    a = \*c + \*d + 1;

    e=c;

    c=d;

    d=e;

    \*c = \*d + \*c + 2;

    printf("\n%p%f",&a,a);

    printf("\n%p%f",&b,b);

    printf("\n%p %p %f",&c,c,\*c);

    printf("\n%p %p %f",&d,d,\*d);

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Endereço** | **Valor** |
| a | 0055FF10 | ~~40~~ 61 |
| b | 0055FF0C | ~~20~~ = 61 + 20 + 2 = 83 |
| c | 0055FF08 | ~~0055FF10~~ 0055FF0C |
| d | 0055FF04 | ~~0055FF0C~~ 0055FF10 |
| e | 0055FF00 | 0055FF10 |
|  |  |  |
|  |  |  |

1. Preencha a tabela da próxima página, conforme o programa for executado e escreva quais saídas serão apresentadas na tela. Considere que a primeira variável será endereçada em 005F0010.

floata,b;

    float \*c,\*d,\*e;

    double f,\*g;

    a=10;

    b=20;

    c=&a;

    d=&b;

    g=&f;

    \*g=10;

    printf("\n%p %f",&a,a);

    printf("\n%p %f",&b,b);

    printf("\n%p %p %f",&c,c,\*c);

    printf("\n%p %p %f",&d,d,\*d);

    printf("\n%p %p %f",&g,g,\*g);

 e=c;

    c=d;

    d=e;

    \*g = \*c + \*d + \*e + a + f+1;

    printf("\n%p %f",&a,a);

    printf("\n%p %f",&b,b);

    printf("\n%p %p %f",&c,c,\*c);

    printf("\n%p %p %f",&d,d,\*d);

    printf("\n%p %p %f",&g,g,\*g + 1);

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Endereço** | **Valor** |
| a | 005F0010 | 10 |
| b | 005F000C | 20 |
| c | 005F0008 | ~~005F0010~~ 005F000C |
| d | 005F0004 | ~~005F000C~~ 005F0010 |
| e | 005F0000 | 005F0010 |
| f | 005EFFF8 | ~~10~~  61 |
| g | 005EFFF4 | 005EFFF8 |

1. Preencha a tabela da próxima página, conforme o programa for executado e escreva quais saídas serão apresentadas na tela. Considere que a primeira variável será endereçada em 00552100.

Float x,d;

    float \*v,\*w,\*t;

    x=10;

    d=20;

    w=&x;

    t=&d;

    printf("\n%p%f",&x,x);

    printf("\n%p%f",&d,d);

    printf("\n%p %p %f",&w,w,\*w);

    printf("\n%p %p %f",&t,t,\*t);

    d = \*t + x;

    v=w;

    w=t;

    t=v;

    \*t = \*v + 5;

    printf("\n%p%f",&x,x);

    printf("\n%p%f",&d,d);

    printf("\n%p %p %f",&w,w,\*w);

    printf("\n%p %p %f",&t,t,\*t);

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Endereço** | **Valor** |
| x | 00552100 | ~~10~~  15 |
| d | 005520FC | ~~20~~  30 |
| \*v | 005520F8 | 00552100 |
| \*w | 005520F4 | ~~00552100~~ 005520FC |
| \*t | 005520F0 | ~~005520FC~~  00552100 |
|  |  |  |
|  |  |  |

1. Considere o trecho de código abaixo:

Float a,b;

float \*c, \*d,\*e;

double f;

double \*g;

a=1;b=2;f=3;

Identifique quais atribuições abaixo não são recomendadas (ou mesmo, que não podem ser realizadas).

f=10; Pode.

a=b; Pode.

a=&b; Não pode.

a=\*b; Não pode.

a=c; Não pode.

a=&c; Não pode.

a=\*c; Pode.

a=g; Não pode.

&a=c; Não pode.

c=a; Não pode.

c=&a; Pode.

c=\*a; Não pode.

c=d; pode

c=&d; Não pode.

c=\*d; Não pode.

c=g; Não pode.

\*c=a; Pode.

\*c=&a; Não pode.

\*c=\*a; Não pode.

\*c=10; Pode.

\*c=d; Não pode.

\*c=\*d; Pode.

\*c=&d; Não pode.

\*c=\*g; Não pode.

\*g=\*c; Pode.

g=c; Não pode.

Obs: caso não saiba a resposta, tente usar a tabela para encontrar a explicação. Lembre-se de perguntar: o valor a ser atribuído é do mesmo tipo? Se a atribuição for feita a uma variável ponteiro, então você deve fazer duas perguntas: o valor atribuído é um endereço de memória? Este endereço de memória permite armazenar valores do tipo declarado no ponteiro?

Obs2: caso não consiga responder, teste no DevC ou CodeBlock para encontrar a resposta. Uma vez que obtenha a resposta, tente achar a explicação. Mas não se esqueça de anotar qual atribuição você teve dificuldade (e precisou da ajuda do DevC ou Codeblock) e envie email para pedir a explicação e confirmar sua resposta.