1. **（书中习题4第15题）以下是对矩阵进行转置的程序段：**

**typedef int array[4][4];**

**void transpose(array dst, array src)**

**{**

**int i, j;**

**for (i = 0; i < 4; i++)**

**for (j = 0; j < 4; j++)**

**dst[j][i] = src[i][j];**

**}**

**假设该段程序运行的计算机中sizeof(int)=4，且只有一级cache，其中L1 data cache的数据区大小为64B，采用直接映射、写回方式，块大小为16B，初始为空。数组dst从地址0000C000H开始存放，数组src从地址0000C040H开始存放。填写下表，说明数组元素src[row][col]和dst[row][col]映射到cache的哪一行，其访问是命中（hit）还是失效（miss）。**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **src数组** | | | | **dst数组** | | | |
| **32B** | **col=0** | **col=1** | **col=2** | **col=3** | **col=0** | **col=1** | **col=2** | **col=3** |
| **row=0** |  |  |  |  |  |  |  |  |
| **row=1** |  |  |  |  |  |  |  |  |
| **row=2** |  |  |  |  |  |  |  |  |
| **row=3** |  |  |  |  |  |  |  |  |

**2. 对于书中习题4第24题，假定采用小端方式， x为short型变量，其虚拟地址为02EAH，则x能否在cache中命中，若能，则x为多少？若不能，为什么？若x的虚拟地址为0E58H，则结果如何？（要求写出过程）**

**3. 主存按字节编址，数组在内存按行优先方式存放。Cache数据区为1024B，块大小为16B，直接映射，gd的地址从0开始，sizeof(int)=4。求以下代码段A、B和C对数组gd的访问中，各自共有多少次读操作？三段代码各有多少次不命中？若Cache数据区扩大为2048B，则各有多少次不命中？**

**struct algae\_pos{**

**int x;**

**int y;**

**} ;**

**struct algae\_pos gd [16][16];**

**int total\_x=0, total\_y=0;**

**int i, j;**

**…**

**for (i = 0; i < 16; i++) {**

**for (j = 0; j<16; j++) {**

**total\_x+= gd[i][j].x;**

**total\_y+= gd[i][j].y;**

**}**

**}**

**…**

**struct algae\_pos{**

**int x;**

**int y;**

**} ;**

**struct algae\_pos gd [16][16];**

**int total\_x=0, total\_y=0;**

**int i, j;**

**…**

**for (i = 0; i < 16; i++) {**

**for (j = 0; j<16; j++) {**

**total\_x+= gd[i][j].x;**

**}**

**}**

**for (i = 0; i < 16; i++) {**

**for (j = 0; j<16; j++) {**

**total\_y+= gd[i][j].y;**

**}**

**}**

**…**

**struct algae\_pos{**

**int x;**

**int y;**

**} ;**

**struct algae\_pos gd [16][16];**

**int total\_x=0, total\_y=0;**

**int i, j;**

**…**

**for (i = 0; i < 16; i++) {**

**for (j = 0; j<16; j++) {**

**total\_x+= gd[j][i].x;**

**total\_y+= gd[j][i].y;**

**}**

**}**

**…**

**代码段A 代码段B 代码段C**