```
#define M 5

#define N 9

int arr1[M][N];
int arr2[N][M];

void swap(int k, int j){
  arr1[k][j] = arr2[j][k];
}

int main(){
  int j = 0; int k = 0;
  for (k = 0; k < M; k++){
    for (j = 0; j < N; j++){
      swap(k,j);
    }
  }
  return 0;
}

ASSEMBLY:
( arr1 = 0x8049540, arr2 = 0x8049600 )

08048308 <swap>:
 8048308:       push   %ebp
 8048309:       mov    %esp,%ebp
 804830b:       mov    0x8(%ebp),%edx             # %edx = k
 804830e:       mov    %edx,%eax                  # %eax = k
 8048310:       shl    $0x3,%eax                  # %eax = 8*k
 8048313:       add    %edx,%eax                  # %eax = 8*k+k
 8048315:       mov    %eax,%ecx                  # %ecx = 9*k
 8048317:       add    0xc(%ebp),%ecx             # %ecx = 9*k + j
 804831a:       mov    0xc(%ebp),%edx             # %edx = j
 804831d:       mov    %edx,%eax                  # %eax = j
 804831f:       shl    $0x2,%eax                  # %eax = 4*j
 8048322:       add    %edx,%eax                  # %eax = 4*j+j
 8048324:       add    0x8(%ebp),%eax             # %eax = 5*j + k
 8048327:       mov    0x8049600(,%eax,4),%eax    # %eax = *((int *)arr2 + 5*j+k)  (the 4
is sizeof(int))
 804832e:       mov    %eax,0x8049540(,%ecx,4)    # *((int *)arr1 + 9*k+j) = %eax
 8048335:       pop    %ebp
 8048336:       ret

------------------------
int scrat(int val, int n)
{
        int result = 0;
        if(n > 0)
              result = val + scrat(val, n-1);
        return result;
}
08048390 <scrat>:
  8048390:      push %ebp
->8048391:      mov %esp,%ebp
  8048393:      push %ebx
  8048394:      sub $0x8,%esp
  8048397:      mov 0x8(%ebp),%ebx
  804839a:      mov 0xc(%ebp),%eax
  804839d:      mov $0x0,%edx
  80483a2:      test %eax,%eax
  80483a4:      jle 80483b6 <scrat+0x26>
  80483a6:      dec %eax
  80483a7:      mov %eax,0x4(%esp,1)
```

```
  80483ab:      mov %ebx,(%esp,1)
  80483ae:      call 8048390 <scrat>
  80483b3:      lea (%eax,%ebx,1),%edx
  80483b6:      mov %edx,%eax
  80483b8:      add $0x8,%esp
  80483bb:      pop %ebx
  80483bc:      pop %ebp
  80483bd:      ret
```

ANSWER:

```
0xbffff9a4      0x3             n
0xbffff9a0      0x21            val
0xbffff99c      0x80483db       return address
0xbffff998      0xbffff9a8      saved ebp
0xbffff994      n/a             saved ebx
0xbffff990      0x2             n-1
0xbffff98c      0x21            val
0xbffff988      0x80483b3       return address
0xbffff984      0xbffff998      saved ebp
0xbffff980      0x21            saved ebx (haven't done this yet..)
0xbffff97c
```

Alignment problem:

1. 0
2. 0
3. 4
4. 8
5. 16
6. 16
7. 64
8. 32
9. 32
10. 2
11. 20
12. 64

Floating Point

Recommended Book Practice Problems: 2.33, 2.34, 2.37