ARM Code

```
        mystery3
LDR     R3, [R0, #8]      #R3 = *(R0+8). Loads a word from [R0+8,R0+0XC)
STR     R3,[R1]           #*R1 = R3
LDR     R3,[R0, #0xC]     #R3 = *(R0+0xC)
MOVS    R0, 0             #R0 = 0
STR     R3,[R1+#4]        #*R1+4 = R3
BX      LR                # return
        ;End of function mystery3
```

Mode

This is in thumb mode since all the instructions are 16 bits.

Types

R0 and R1 can be anything as well as R3. Therefore, just assume that they are void pointers.

When R0 returns, it returns 0 which makes R0 an int for return.

Function Prototype

int mystery3(void * arg1, void * arg2)

C Code

```
int mystery3(void *arg1, void * arg2){
        char * ptr; //char is one byte
        int count =8;
        ptr = arg1+8; //Start pointer from arg1+8
        while(count){
          *((char *) arg2) = *ptr;
          ptr++;
          arg2++;
          count--;
        }
        return 0;
```

Explanation

This code is copying the 8 bytes from one memory pointer to another memory pointer i.e. memcpy(). The most number of bits that the ARM code can copy and load at a time is 32 which means it takes two sets of load and store instructions to copy all 8 bytes.