Name: Solution Key

Note: There are three questions for a total of 100 points. Please write your answers in the space provided.

Question I (35 points total)

(a) In C, what is the significance of the static attribute for a function? (5 points)

Answer: The "static" attribute for a function means that the function can be called only by functions within the same file (i.e., the scope of the function is limited to the file containing the function).

(b) Show the output produced by the following C program. There are no syntax errors in the program. Mark your final answers clearly. (30 points)

```c
#include <stdio.h>
float q = 2.5;

void mystery(void);

int main(void) {
    int p = 2, j;
    for (j = 0; j < p; j++) {
        printf("%f\n", q); mystery();
    }
    return 0;
} /* End of main */

void mystery(void) {
    float q = -1.0; static int r = 23;
    q = 2 * q;
    printf("%f %d\n", q, --r);
} /* End of mystery */
```

Output:

```
2.5
-2.0 22
2.5
-2.0 21
```
Question II (25 points total)

(b) Show the sequence of TAL instructions needed to implement the MAL instruction shown below.
You may assume that the product fits in 32 bits. (8 points)

\[
mul \quad $11, \quad $7, \quad $8
\]

Answer: The sequence of TAL instructions is as follows:

\[
mult \quad $7, \quad $8 \quad \# \quad \text{Product in HI & LO registers.}
mflo \quad $11 \quad \# \quad \text{Copies LO into $11.}
\]

(b) Assume that the data type \texttt{int} uses 4 bytes. A two-dimensional \texttt{int} array \texttt{Y} with 4 rows and 5 columns is stored in column-major order. The address of the element \texttt{Y[3][2]} is 864 (decimal). Find the starting address of the array. Your answer must be in decimal. Work must be shown to receive partial credit. (17 points)

Answer: Let \( s \) denote the starting address of the array \texttt{Y}.

By the formula for column-major order and using the fact that each integer uses 4 bytes, the address of element \texttt{Y[3][2]} is given by

\[
s + 4 \times 2 \times 4 + 4 \times 3 = s + 44
\]

Since the address of \texttt{Y[3][2]} is given as 864, we have

\[
s + 44 = 864 \quad \text{or} \quad s = 820
\]

Thus, the starting address of \texttt{Y} = 820.
Question III (40 points)
You are required to write a MAL function named CountOnes (i.e., a MAL program segment whose first executable statement has the label CountOnes). This function should assume that register $6$ contains a 32-bit integer. Imagine that the 32 bits are numbered from right to left using integers 0 to 31. (Thus, the rightmost bit is numbered 0 and the leftmost bit is numbered 31.) The function must determine how many of the bits in positions 8 through 15 of the value in $6$ are 1 and return that count in $7$. (Thus, the value returned in $7$ is an integer in the range 0 through 8.) The function must return to its caller using the MAL instruction jr $31$.

You may use other registers in the function. There is no need to save/restore those registers.

Your answer needs to show only the MAL code for the CountOnes function. There is no need to include comments in your code.

Note: If you need additional space for your answer, please use the back of this sheet.

Idea: We start with the mask 0x100 (which has 1 in bit position 8 and 0’s everywhere else). Suppose this mask is bitwise-anded with the value in $6$. If the result is non-zero (0), then bit in position 8 of $6$ is 1 (0). Now, we can shift the mask one position to the left and check bit 9, and so on. Thus, a simple loop can be used to check the bits in positions 8 through 15 and obtain the required count.

In the following MAL code, registers $9$, $10$, $11$ and $12$ are used as temporaries.

MAL Code:
---------

.text

# Initialization steps.

CountOnes: move $7, $0 # Init. required count to 0.
li $9, 0x100 # Mask in $9.
li $10, 8 # No. of bit positions to be checked.
li $11, 0 # No. of bit positions already checked.

# Loop to check bit positions 8 through 15 and obtain the required count.

loop: and $12, $6, $9 # Bitwise-and with current mask.
beqz $12, skip # If result is 0, don't increment.
addi $7, $7, 1 # Increment the no. of 1's.

skip: sll $9, $9, 1 # Left shift mask by 1 position.
addi $11, $11, 1 # Increment no. of positions checked.
bne $11, $10, loop # If more positions are to be checked,
# go back to the top of the loop.

# Here, all the eight positions have been checked and the count is in $7$. So,

jr $31