**Part I:**

**Part II:**

Brief Description:

The way I implemented my MIPS program was I first prompt the user to enter the 32-bit signed/magnitude binary number as a string. For entering the signed binary number, the user must enter all 32 bits with spaces between for every 4 bits (ex: 0000 0000 0000 0000 0000 0000 0000 0000). After input, the program stores the entered string in a variable in .data (strBinNum). Then the program loads the string and reads each character of the string to check if it’s valid signed binary number. A valid signed binary number can have 0’s, 1’s, and spaces. If the string is invalid, then the program will print out their invalid input and print out how it should look like with a small explanation, then re-prompts the user to enter their signed binary number. After the user entered a valid signed binary number, then the program will start converting the signed binary number to its two’s complement format. The way the program does this is by visiting each character of the string. Before the program iterates through the string, it sets up counters for calculating the powers of 2 for the respective place value in the binary number using logical shift.