CS 281 HW W1A

2.1

C Code:
f = g + (h - 5)

MIPS Code:
addi $t0, $s2, -5

- Add the value stored in the variable s2, which is associated with variable h in C code, by -5, thus subtracting the two values
- Store the subtracted value in the temporary variable t0

add $s0, $s1, $t0

- Add the value stored in variable s1, which is associated with the variable g in C code, to the value stored in the temporary variable t0
- Store the added value in the variable s0, which is associated with the variable f in C code

2.3

C Code:
B[8] = A[i - j]

MIPS Code:
sub $t0, $s3, $s4

- Subtract the value stored in the variable s3, which is associated with the variable i in C code, by the value stored in the variable s4, referenced by variable j in C code
- Store the subtracted value in the temporary variable t0

sll $t0, $t0, 2

- Shift the array index over the correct number of spots to scale the array properly, since array index values are multiplied by 4 to get the correct index in MIPS
- Store the shifted index in temporary variable t0

add $t0, $t0, $s6

- Change the index of the array stored in s6, which is associated with the array A in C code, to the value stored in t0 by adding them
- Store the array with the shifted index in the temporary variable t0

lw $t0, 0($t0)

- Load the value of the array with the correct index offset, referenced by the variable t0, into the temporary variable t0
- 0 is used because the array was properly indexed in the previous step

sw $t0, 32($s7)

- Store the value of the temporary variable t0 into index 8 of the array referenced by variable s7, which is associated with the array B in C code
- 32 is used because indexes are in multiples of 4 in MIPS code

2.11

- addi $t0, $s6, 4

OP RS RT

8 22 8 Immediate Field = 4

OP code equals 8 based on a given chart.
In the MIPS code, for I-type format, the RS code is the second position, the RT code is the first position, and the Immediate Field code is the given integer.
RS equals 22 and RT equals 8 based on a given chart, and the Immediate Field equals 4 based on the given integer in the MIPS code

- add $t1, $s6, $0

OP RS RT

0 22 0 RD = 9

OP equals 0 based on a given chart.
In the MIPS code, for R-type format, the RS code is the second position, the RT code is the third position, and the RD code is the first position.
RS equals 22, RT equals 0, and RD equals 9 based on a register chart.

- sw $t1, 0($t0)

OP RS RT
43 8 9 Immediate Field = 0

OP equals 43 based on a given chart.
In the MIPS code, for I-type format, the RS code is the second position, the RT code is the first position, and the Immediate Field code is the given integer.
RS equals 8 and RT equals 9 based on a given chart, Immediate Field equals 0 based on the given integer in the MIPS code.

- lw $t0, 0($t0)

OP	RS	RT	
35	8	8	Immediate Field = 0

OP code equals 35 based on a given chart.
In the MIPS code, for I-type format, the RS code is the second position, the RT code is the first position, and the Immediate Field code is the given integer.
RS equals 8 and RT equals 8 based on a given chart, Immediate Field equals 0 based on the given integer in the MIPS code.

- add $s0, $t1, $t0

OP	RS	RT	
0	9	8	RD = 16

OP equals 0 based on a given chart.
In the MIPS code, for R-type format, the RS code is the second position, the RT code is the third position, and the RD code is the first position.
RS equals 9, RT equals 8, and RD equals 16 based on a register chart.

2.15

sw $t1, 32($t2)

OP	RS	RT	
43	10	9	RD = 32

OP equals 43 based on a given chart.
In the MIPS code, for I-type format, the RS code is the second position, the RT code is the first position, and the Immediate Field code is the given integer.
RS equals 10 and RT equals 9 based on a given chart, Immediate Field equals 32 based on the given integer in the MIPS code.

Binary code

OP	RS	RT	
101011	01010	01001	RD = 0000000000100000

Combined binary code: 10101101010010010000000000100000

Hex code: AD490020

2.39

0010 0000 0000 0001 0100 1001 0010 0100

Binary	Decimal
0010000000000001	8193
0100100100100100	18724

lui $t1, 8193

- The first 16 bits
- Load upper 16 bits into the temporary variable t1

ori $t1, $t1, 18724

- The last 16 bits
- Load lower 16 bits into the temporary variable t1