Homework #2

- Chapter 2：2.1, 2.3, 2.6, 2.12, 2.16
- Due on Mar. 12

11713020
张集晨

2.1 [5] <§2.2> For the following C statement, what is the corresponding MIPS assembly code? Assume that the variables `f`, `g`, `h`, and `i` are given and could be considered 32-bit integers as declared in a C program. Use a minimal number of MIPS assembly instructions.

```
f = g + (h - 5);
```

```
lw $t0, h          # register $t0 contains h
subi $t0, $t0, -5  # register $t0 contains h-5
lw $t1, g          # register $t1 contains g
add $t0, $t0, $t1  # register $t0 contains g+(h-5)
sw $t0, f          # save register $t0 to variable f
```

2.3 [5] <§§2.2, 2.3> For the following C statement, what is the corresponding MIPS assembly code? Assume that the variables `f`, `g`, `h`, `i`, and `j` are assigned to registers `$s0`, `$s1`, `$s2`, `$s3`, and `$s4`, respectively. Assume that the base address of the arrays `A` and `B` are in registers `$s6` and `$s7`, respectively.

```
B[8] = A[i-j];
```

```
sub $t0, $s3, $s4  # register $t0 contains i-j
sll $t0, $t0, 2    # register $t0 contains 4(i-j)
add $t0, $t0, $s6  # register $t0 contains $s6+4(i-j)
lw $t0, ($t0)      # reg $t0 = A[i-j]
addi $t1, $s7, 32  # reg $t1 = $s7 + 32
sw $t0, ($t1)      # save A[i-j] to B[8]
```

2.6 The table below shows 32-bit values of an array stored in memory.

Address	Data
24	2
28	4
32	3
36	6
40	1

2.6.1 [5] <§§2.2, 2.3> For the memory locations in the table above, write C code to sort the data from lowest to highest, placing the lowest value in the smallest memory location shown in the figure. Assume that the data shown represents the C variable called `Array`, which is an array of type `int`, and that the first number in the array shown is the first element in the array. Assume that this particular machine is a byte-addressable machine and a word consists of four bytes.

```
int Array[5] = {2, 4, 3, 6, 1};
int temp = Array[0];
Array[0] = Array[4];
Array[4] = temp; // {1, 4, 3, 6, 2}
temp = Array[1];
Array[1] = Array[4];
Array[4] = temp; // {1, 2, 3, 6, 4}
temp = Array[3];
Array[3] = Array[4];
Array[4] = temp; // {1, 2, 3, 4, 6}
```

2.6.2 [5] <§§2.2, 2.3> For the memory locations in the table above, write MIPS code to sort the data from lowest to highest, placing the lowest value in the smallest memory location. Use a minimum number of MIPS instructions. Assume the base address of `Array` is stored in register `$s6`.

```
lw $t0, ($s6)  # reg $t0 = 2
lw $t1, 16($s6) # reg $t1 = 1
sw $t1, ($s6)  # Array[0] = 1
sw $t0, 16($s6) # Array[4] = 2
lw $t0, 4($s6) # reg $t0 = 4
lw $t1, 16($s6) # reg $t1 = 2
sw $t1, 4($s6) # Array[1] = 2
sw $t0, 16($s6) # Array[4] = 4
```

```
lw $t0, 12($s6) # reg $t0 = 6
lw $t1, 16($s6) # reg $t1 = 4
sw $t1, 12($s6) # Array[3] = 4
sw $t0, 16($s6) # Array[4] = 6
```

2.12 Assume that registers `$s0` and `$s1` hold the values `0x80000000` and `0xD0000000`, respectively.

2.12.1 [5] <§2.4> What is the value of `$t0` for the following assembly code?

```
add $t0, $s0, $s1
```

```
# $s0 = 0x80000000 = 1000...
# $s1 = 0xD0000000 = 1101...
# $s0 + $s1 = 0101...
result: $t0 = 0x50000000
```

2.12.2 [5] <§2.4> Is the result in `$t0` the desired result, or has there been overflow?

The result has been overflow.

2.12.3 [5] <§2.4> For the contents of registers `$s0` and `$s1` as specified above, what is the value of `$t0` for the following assembly code?

```
sub $t0, $s0, $s1
```

```
# $s0 = 0x80000000 = 1000...
# $s1 = 0xD0000000 = 1101...
# $s0 - $s1 = 1011...
result: $t0 = 0xB0000000
```

2.12.4 [5] <§2.4> Is the result in `$t0` the desired result, or has there been overflow?

The result in $t0 is the desired result.

2.12.5 [5] <§2.4> For the contents of registers `$s0` and `$s1` as specified above, what is the value of `$t0` for the following assembly code?

```
add $t0, $s0, $s1
add $t0, $t0, $s0
```

```
# $s0 = 0x80000000 = 1000...
# $s1 = 0xD0000000 = 1101...
add $t0, $s0, $s1  # $t0 = 1011... = 0xB0000000
add $t0, $t0, $s0  # $t0 = 0011... = 0x30000000
result: $t0 = 0xD0000000
```

2.12.6 [5] <§2.4> Is the result in `$t0` the desired result, or has there been overflow?

The result has been overflow.

2.16 [5] <§2.5> Provide the type, assembly language instruction, and binary representation of instruction described by the following MIPS fields:

`op=0, rs=3, rt=2, rd=3, shamt=0, funct=34` 0x22

Type: R-type Instruction

Assembly Language: sub $v1, $v1, $v0

Binary Representation: 000000 00011 00010 00011 00000 100010