Homework 1: MIPS and Instruction Encoding

Problem 1 - Implement the following C code in MIPS assembly. Assume your own variables to register mapping and document it as a register map comment in your assembly code. Using the `la` instruction, load the base address of `arr` into $s2. You may only use the instructions given in the MIPS Instruction Sheet and the la instruction to load the array address. Verify the operation in QtSPIM.

```
int main() {

int arr[]={10, 9, 8, 7, 6, 5, 4, 3, 2, 1},n=10,i,j,temp;

for (i = 0; i < n - 1; i++){
   for (j = 0; j < n - i - 1; j++){
     if (arr[j] > arr[j + 1]){
        temp=arr[j];
        arr[j]=arr[j+1];
        arr[j+1]=temp;
     }
  }
}
```

Hint: Declare the array in .data section as follows:
arr: .word 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

See attached *.asm file

Problem 2 - Determine the MIPS instruction and instruction type for the binary entries:

a. 100011 01000 01000 0000000001000000
   op rs rt address/imm

lw $8, $8
I-type Instruction: lw rt, imm(rs)

b. 000000 11111 00000 00000 00000 001000
   op rs rt rd shamt funct

jr $31
R-type Instruction: jr rs

Problem 3 - Determine the binary representation for the MIPS instructions:

a. nor $t2, $s3, $zero
   nor rd, rs, rt (R-type Instruction)

000000 10011 00000 01010 00000 100111
op rs rt rd shamt funct

b. lw $t2, 8($s3)
lw rt, imm(rs) (I-type Instruction)

100011 10011 01010 0000000000001000
op rs rt address/imm

c. add $s0, $s1, $s2
add rd, rs, rt (R-type Instruction)

000000 10001 10010 10000 00000 100000
op rs rt rd shamt funct

Problem 4 - Using your solution for Problem 1, provide the binary for the MIPS instructions only pertaining to the code snippet below:

```
temp=arr[j];
arr[j]=arr[j+1];
arr[j+1]=temp;
```

op rd , rs , rt
add $t2, $s0, $t2 # temp = t2 = arr address + offset = &arr[j]
op rs rt rd shamt funct
000000 10000 01010 01010 00000 100000

op rt, imm(rs)
sw $s5, 4($t2) # arr[j] = arr[j+1]
op rs rt add/imm
101011 01010 10101 0000000000000100

op rt, imm(rs)
sw $s6, 0($t2) # arr[j+1] = temp
op rs rt add/imm
101011 01010 10110 0000000000000000