Question 5 (30 points):

Write the RISC-V assembly code for function `flipBits`. The parameters for `flipBits` are as follows:

- `a0`: the address of the first position of a null-terminated string S.

For each character `flipBits` flips a single bit in the character. Bit 0 of character 0 is flipped, bit 1 of character 1, ..., bit 7 of character 7, bit 0 of character 8, bit 1 of character 9, and so far.

`flipBits` must work for any string length, including empty strings.

`flipBits` must invoke the `flipBitInByte` from Question 5 to flip a bit of a character.

`flipBits` does not have any return values.

Your RISC-V code must follow all the register saving/restoring convention of RISC-V.

```
1  # flipBits
2  # a0: address of first position of a null-terminated string S.
3  # Pseudo code:
4  # p <- a0
5  # for(i=0 ; i<n ; i++)
6  #    bitpos <- i & 0x07
7  #    flipBitInByte(p,i)
8  #    p++
9  flipBits:
10     addi    sp, sp, -16
11     sw      s0, 0(sp)
12     sw      s1, 4(sp)
13     sw      s2, 8(sp)
14     sw      ra, 12(sp)
15     mv      s0, a0          # p <- Address{B[0]}
16     mv      s1, zero        # i <- 0
17     lbu     t0, 0(s0)       # c <- *p
18     beq     t0, zero, Done  # if c=null, done
19 NextByte:
20     andi    a1, s1, 0x07    # a1 <- i&0x07
21     mv      a0, s0          # a0 <- p
22     jal     flipBitInByte
23     addi    s1, s1, 1       # i <- i+1
24     addi    s0, s0, 1       # p <- p+1
25     lbu     t0, 0(s0)
26     bne     t0, zero, NextByte
27 Done:
28     sw      s0, 0(sp)
29     sw      s1, 4(sp)
30     sw      s2, 8(sp)
31     sw      ra, 12(sp)
32     addi    sp, sp, -16
33     jalr    zero, ra, 0
```

Figure 1: A solution for `flipBits`.

A simpler version of the solution appears in Figure **??**. A slightly more complicated solution — for programmers that did not realized that a XOR 0 = a and that a XOR 1 = /a — is shown in Figure Figure **??**

Figure 2: Another solution for `flipBits`.