Question 4 (12 points):

In each item of this question you are asked to write the sequence of RISC-V instructions that implements a single C-language statement.

a. (**4 points**) The statement appears in the following C function. Assume that all the RISC-V conventions about register usage are followed by the assembly code generator for this function. Use a minimum number o instructions to generate code for the statement.

```
foo(char *p, char *q){
     *p = *q
     ...
     }
```

```
     lb   t0, 0(a1)    # t0 <- *q
     sb   t0, 0(a0)    # *p <- t0
```

b. (**4 points**) `p` is in the stack frame of a function at position `fp-4` and `q` is in the stack frame at position `fp-12`. This program runs in a machine where memory addresses are 32 bits and double values are stored in 64 bits. Use a minimum number o instructions to generate code for the statement. The relevant declaration in the code and the statement are :

```
double **p, **q;
*p = *q
```

```
     lw   t0, -4(fp)     # t0 <- p
     lw   t2, -12(fp)    # t2 <- q
     lw   t1, 0(t2)      # t1 <- *q
     sw   t1, 0(t0)      # *p <- *q
```

c. (**4 points**) `p` is in the stack frame of a function at position `sp+8` and `q` is in the stack frame at position `sp+4`. This program runs in a machine where memory addresses are 32 bits and integer values are stored in 32 bits. Use a minimum number o instructions to generate code for the statement. The relevant declaration in the code and the statement are :

```
double **p, **q;
*(p+5) = *(q+3)
```

```
     lw   t0, 8(sp)     # t0 <- p
     lw   t2, 4(sp)     # t2 <- q
     lw   t1, 12(t2)    # t1 <- *(q+3)
     sw   t1, 20(t0)    # *(p+5) <- *(q+3)
```