本解答僅為正解範例，同學不用完全跟上面一樣

1.

a. i. 錯，要先移到register上面
   ii. 錯，x0無法被寫入

b.

addi是register + immediate，add是register + register
如果沒有addi，我們就要先把值移到register上，才能加

c.

```
        addi    t0, sp, 0
        addi    t1, sp, 40
        addi    t2, zero, 100
        j       .L0
.L1:
        sd      t2, 0(t0)
        addi    t0, t0, 8
.L0:
        bne     t0, t1, .L1
```

[註: 有些同學在作業上會發現助教寫"勉強給分"的情況，是因為for-loop必須要先經過條件判斷才是正常的寫法，但由於同學的執行結果是對的，所以才不扣分]

d.

0 000000 00000 00110 000 10000 1100011
1 1111111010 1 11111111 00000 1101111

e.

(to put -107062541 into x19)

```
lui     x19, 26138          (107062541 / 4096取整數)
addi    x19, x19, 1293      (107062541 % 4096)
xori    x19, x19, -1        (invert the bits in x19)
addi    x19, x19, 1         (+1 becoming 2’s complement -107062541)
```

(注意：addi immediate的範圍是介於-2的11次方~2的11次方減1)
(注意：lui不能放入負數，範圍必須是[0, 2^20 - 1])

2.

a.

```
.L3:
        ld      a5, -24(s0)
        sll     a5, a5, 3
        addi    t0, a5, 0       (mv t0, a5)
        ld      a4, -40(s0)
        add     a5, a4, t0
        ld      a3, 0(a5)
        ; delete two lines
        ld      a4, -48(s0)
        add     a5, a4, t0
        ld      a4, 0(a5)
        ; delete two lines
        ld      a2, -32(s0)
        add     a5, a2, t0
```

b.

yes, cpu time = instruction/program * cycle/instruction * second/cycle
已知instruction數下降，且在各種instruction的CPI不變的情況下，總cycle數自然下降
-> 效能較佳

c.

```
element_wise_product_n:
        add     sp,sp,-56
        sd      ra,48(sp)
        sd      s0,40(sp)
        add     s0,sp,56
        sd      a0,-40(s0)
        sd      a1,-48(s0)
        sd      a2,-56(s0)
        li      a0,800
        call    malloc
        mv      a5,a0
        sd      a5,-32(s0)
        sd      zero,-24(s0)
        j       .L2
        .align  2
.L3:
        ld      a5,-24(s0)
        sll     a5,a5,3
        ld      a4,-40(s0)
        add     a5,a4,a5
        ld      a3,0(a5)
        ld      a5,-24(s0)
        sll     a5,a5,3
        ld      a4,-48(s0)
        add     a5,a4,a5
        ld      a4,0(a5)
        ld      a5,-24(s0)
        sll     a5,a5,3
        ld      a2,-32(s0)
        add     a5,a2,a5
        mul     a4,a3,a4
        sd      a4,0(a5)
        ld      a5,-24(s0)
        add     a5,a5,1
        sd      a5,-24(s0)
        .align  2
.L2:
        ld      a4,-24(s0)
        ld      a5,-56(s0)
```

```
addi    a5,a5,-1
; li    a5,3    (deleted)
ble     a4,a5,.L3
ld      a5,-32(s0)
mv      a0,a5
ld      ra,48(sp)
ld      s0,40(sp)
add     sp,sp,56
jr      ra
```

d.

進function時

i. 減stack，return前加回

ii. 存入ra，return前放回 (在該function有call其它function的情況下)

iii. 存入s0，return前放回

iv. use a0~a7 as arguments, a0 as return value

...etc

[You need to point out any example in the assembly program that is relative to RISC-V calling convention to get the full credit.]

e.

A在8(sp)

B在0(sp)

C在16(sp)

i在24(sp)