Figure 3: C and assembly language code listings for Exercise D.

```c
int procC(int x)
{
    // POINT ONE

    return 8 * x + 2 * x;
}

void procB(int *p, int *q)
{
    while (p != q) {
        *p = procC(*p);
        p++;
    }
}

int procA(int s, int *a, int n)
{
    int k;
    k = n - 1;
    procB(a, a + n);
    while (k >= 0) {
        s += a[k];
        k--;
    }
    return s;
}

int gg[] = { 2, 3, 4 };

int main(void)
{
    int mv;
    mv = 1000;
    mv += procA(200, gg, 3);
    return 0;
}
```

```
    .text
    .globl  procC
procC:
    # POINT ONE

    slli    t0, a0, 3
    slli    t1, a0, 1
    add     a0, t0, t1
    jr      ra

    .text
    .globl  procB
procB:
    addi    sp, sp, -32
    sw      ra, 8(sp)
    sw      s1, 4(sp)
    sw      s0, 0(sp)
    add     s0, a0, zero
    add     s1, a1, zero

L1:
    beq     s0, s1, L2
    lw      a0, (s0)
    jal     procC
    sw      a0, (s0)
    addi    s0, s0, 4
    j       L1
L2:
    lw      s0, 0(sp)
    lw      s1, 4(sp)
    lw      ra, 8(sp)
    addi    sp, sp, 32
    jr      ra
```

```
    .text
    .globl  procA
procA:
    addi    sp, sp, -32
    sw      ra, 16(sp)
    sw      s3, 12(sp)
    sw      s2, 8(sp)
    sw      s1, 4(sp)
    sw      s0, 0(sp)
    add     s0, a0, zero
    add     s1, a1, zero
    add     s2, a2, zero

    addi    s3, s2, -1
    add     a0, s1, zero
    slli    t0, s2, 2
    add     a1, s1, t0
    jal     procB
L3: blt     s3, zero, L4
    slli    t1, s3, 2
    add     t2, s1, t1
    lw      t3, (t2)
    add     s0, s0, t3
    addi    s3, s3, -1
    j       L3
L4: add     a0, s0, zero

    lw      s0, 0(sp)
    lw      s1, 4(sp)
    lw      s2, 8(sp)
    lw      s3, 12(sp)
    lw      ra, 16(sp)
    addi    sp, sp, 32
    jr      ra

    .data
    .globl  gg
gg: .word   2, 3, 4

    .text
    .globl  main
main:
    addi    sp, sp, -32
    sw      ra, 4(sp)
    sw      s0, 0(sp)

    addi    s0, zero, 1000
    addi    a0, zero, 200
    la      a1, gg
    addi    a2, zero, 3
    jal     procA
    add     s0, s0, a0
    add     a0, zero, zero

    lw      s0, 0(sp)
    lw      ra, 4(sp)
    addi    sp, sp, 32
    jr      ra
```

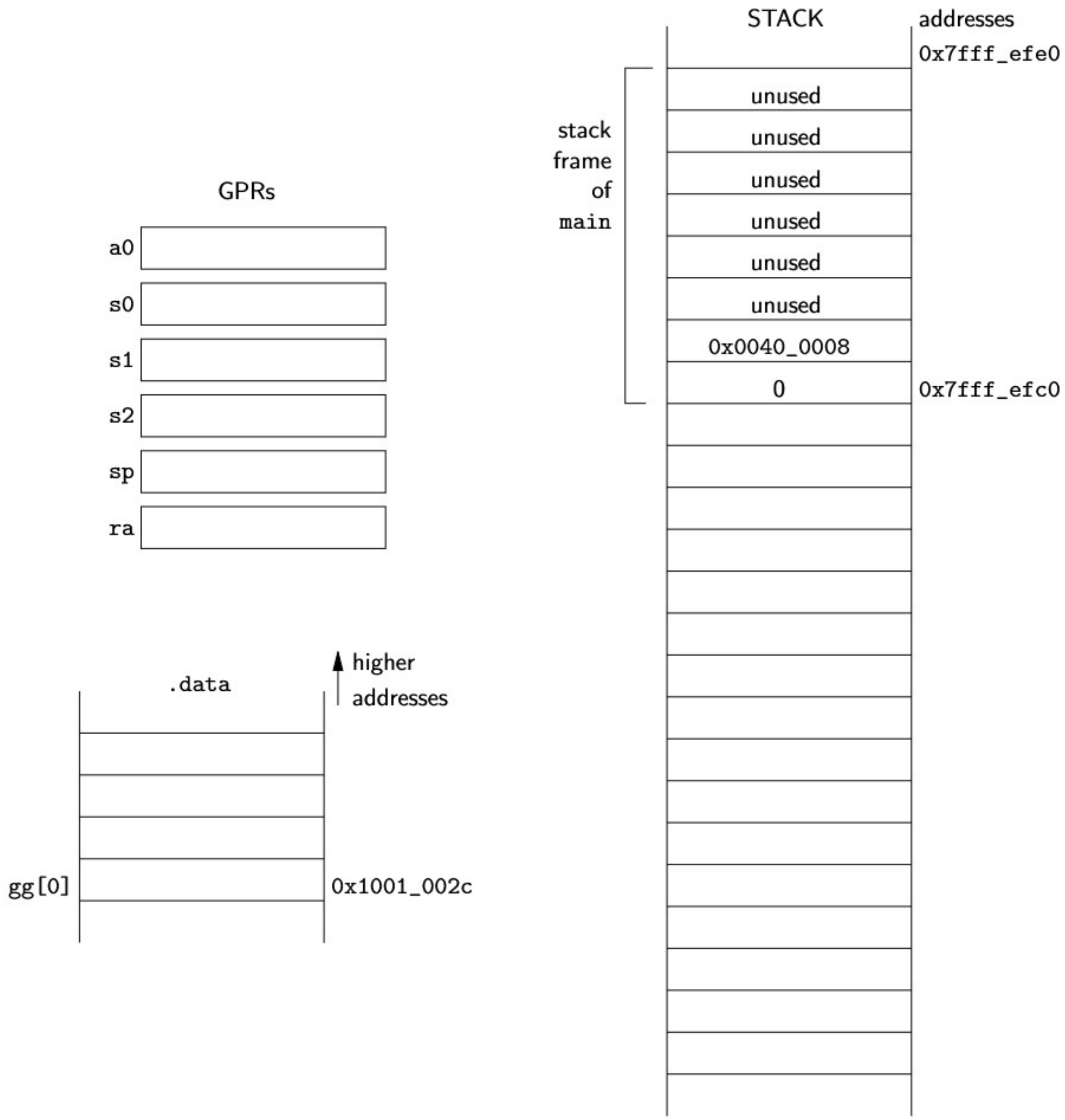
Figure 4: Worksheet for Exercise D.

Addresses of functions and data:

label	address
procC	0x0040_0050
procB	0x0040_0060
procA	0x0040_00a4
main	0x0040_0118
gg	0x1001_002c

Contents of some GPRs when main starts:

GPR	contents
s0-s11	all 0
sp	0x7fff_efe0
ra	0x0040_000c


STACK
addresses
0x7fff_efe0
stack frame of main
unused
unused
unused
unused
unused
unused
0x0040_0008
0
0x7fff_efc0
GPRs
a0
s0
s1
s2
sp
ra
.data
higher addresses
gg[0]
0x1001_002c