```
int C (int *x1, int x2, int x3, int x4, int x5) {
        int i ;
        i = x2 + x3 + x4 + x5 ;
        *x1 = i ;
}

void B (int a, int *b) {
        int z ;
        *b = C (&z, a+1, a+2, a+3, a+4) ;
}

void A () {
        int x=2, y ;
        B (3 * x, &y) ;
        x = y  ;
}

main () { A() ; }

/* ------------------------------------------------------------------ */
        .file   "abc2.c"
        .text
        .align  2
        .global C
        .type   C,function
C:
        @ args = 4, pretend = 0, frame = 20
        @ frame_needed = 1, uses_anonymous_args = 0
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        sub     sp, sp, #20
        str     r0, [fp, #-16]
        str     r1, [fp, #-20]
        str     r2, [fp, #-24]
        str     r3, [fp, #-28]
        ldr     r2, [fp, #-20]
        ldr     r3, [fp, #-24]
        add     r2, r2, r3
        ldr     r3, [fp, #-28]
        add     r2, r2, r3
        ldr     r3, [fp, #4]
        add     r3, r2, r3
        str     r3, [fp, #-32]
        ldr     r2, [fp, #-16]
        ldr     r3, [fp, #-32]
        str     r3, [r2, #0]
        mov     r0, r3
        ldmea   fp, {fp, sp, pc}
.Lfe1:
        .size   C,.Lfe1-C
        .align  2
        .global B
        .type   B,function
B:
        @ args = 0, pretend = 0, frame = 12
        @ frame_needed = 1, uses_anonymous_args = 0
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        sub     sp, sp, #16
        str     r0, [fp, #-16]
```



```
        str     r1, [fp, #-20]
        sub     r2, fp, #24
        ldr     r3, [fp, #-16]
        add     r1, r3, #1
        ldr     r3, [fp, #-16]
        add     ip, r3, #2
        ldr     r3, [fp, #-16]
        add     lr, r3, #3
        ldr     r3, [fp, #-16]
        add     r3, r3, #4
        str     r3, [sp, #0]
        mov     r0, r2
        mov     r2, ip
        mov     r3, lr
        bl      C
        mov     r2, r0
        ldr     r3, [fp, #-20]
        str     r2, [r3, #0]
        ldmea   fp, {fp, sp, pc}
.Lfe2:
        .size   B,.Lfe2-B
        .align  2
        .global A
        .type   A,function
A:
        @ args = 0, pretend = 0, frame = 8
        @ frame_needed = 1, uses_anonymous_args = 0
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        sub     sp, sp, #8
        mov     r3, #2
        str     r3, [fp, #-16]
        ldr     r2, [fp, #-16]
        mov     r3, r2
        mov     r3, r3, asl #1
        add     r3, r3, r2
        sub     r2, fp, #20
        mov     r0, r3
        mov     r1, r2
        bl      B
        ldr     r3, [fp, #-20]
        str     r3, [fp, #-16]
        ldmea   fp, {fp, sp, pc}
.Lfe3:
        .size   A,.Lfe3-A
        .align  2
        .global main
        .type   main,function
main:
        @ args = 0, pretend = 0, frame = 0
        @ frame_needed = 1, uses_anonymous_args = 0
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        bl      A
        mov     r0, r3
        ldmea   fp, {fp, sp, pc}
.Lfe4:
        .size   main,.Lfe4-main
        .ident  "GCC: (GNU) 3.1"
```