

```
int x[10000], p;

void proc_b (int v[], int k)
{
int temp;

temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;

}

void proc_a (int v[], int n)
{
int i, j;

for (i = 0; i < n; i = i+1) {
for (j = i-1; j >= 0 && v[j] > v[j+1]; j = j-1) {
proc_b(v,j);
}
}

}

void main() {
p = 10000;
proc_a(x,p);
}
```

```

.size      proc_a,.Lfe2-proc_a      .file      "prog.c"
.align 4   .version      "01.01"
.type      main,@function          gcc2_compiled.:
main:      .text
           .align 4
           pushl   %ebp
           movl   %esp,%ebp
           movl   $10000,p
           movl   p,%eax
           pushl   %eax
           pushl   $x
           call  proc_a
           addl   $8,%esp
           leave
           ret

.size      main,.Lfe3-main
.comm     x,40000,4
.comm     p,4,4
.ident    "GCC: (GNU) 2.7.2.3"

           .globl proc_b
           .type   proc_b,@function
proc_b:
           pushl   %ebp
           movl   %esp,%ebp
           subl   $4,%esp
           pushl   %ebx
           movl   12(%ebp),%eax
           movl   %eax,%edx
           leal   0(,%edx,4),%eax
           movl   8(%ebp),%edx
           movl   (%edx,%eax),%eax
           movl   %eax,-4(%ebp)
           movl   12(%ebp),%eax
           movl   %eax,%edx
           leal   0(,%edx,4),%eax
           movl   8(%ebp),%edx
           movl   12(%ebp),%ecx
           movl   %ecx,%ebx
           leal   0(,%ebx,4),%ecx
           movl   %ecx,%ebx
           addl   8(%ebp),%ebx
           leal   4(%ebx),%ecx
           movl   (%ecx),%ebx
           movl   %ebx,(%edx,%eax)
           movl   12(%ebp),%eax
           movl   %eax,%edx
           leal   0(,%edx,4),%eax
           movl   %eax,%edx
           addl   8(%ebp),%edx
           leal   4(%edx),%eax
           movl   -4(%ebp),%edx
           movl   %edx,(%eax)

           .L1:
           movl   -8(%ebp),%ebx
           leave
           ret

```

```

.Lfe1:
    .size    proc_b,.Lfe1-proc_b
    .align 4
.globl proc_a
proc_a:
    .type    proc_a,@function
    pushl   %ebp
    movl    %esp,%ebp
    subl   $8,%esp
    pushl   %esi
    pushl   %ebx
    nop
    movl    $0,-4(%ebp)
.L3:
    movl    -4(%ebp),%eax
    cmpl   %eax,12(%ebp)
    jg     .L6
    jmp    .L4
    .align 4
.L6:
    movl    -4(%ebp),%esi
    decl   %esi
    movl    %esi,-8(%ebp)
.L7:
    cmpl   $0,-8(%ebp)
    jl    .L11
    movl   -8(%ebp),%eax
    movl   %eax,%edx
    leal  0(%edx,4),%eax
    movl   8(%ebp),%edx
    movl   -8(%ebp),%ecx
    movl   %ecx,%ebx
    leal  0(%ebx,4),%ecx
    movl   %ecx,%ebx
    addl   8(%ebp),%ebx
    leal  4(%ebx),%ecx
    movl   (%edx,%eax),%eax
    cmpl   %eax,(%ecx)
    jl    .L10
    jmp    .L11
    .align 4
.L11:
    jmp    .L8
    .align 4
.L10:
    movl   -8(%ebp),%eax
    pushl  %eax
    movl   8(%ebp),%eax
    pushl  %eax
    call   proc_b
    addl   $8,%esp
.L9:
    decl   -8(%ebp)
    jmp    .L7
    .align 4
.L8:
.L5:
    incl   -4(%ebp)
    jmp    .L3
    .align 4
.L4:
.L2:
    leal  -16(%ebp),%esp
    popl  %ebx
    popl  %esi
    leave
    ret

```

```

Lfe2:                                .file      "prog.c"
                                       .version  "01.01"
                                       .size     proc_a,Lfe2-proc_a
                                       .align 4
                                       gcc2_compiled.:
                                       .type     main,@function
main:                                .align 4
                                       .globl   proc_b
                                       .type     proc_b,@function
                                       .globl   proc_b:
push    bp
mov     sp,bp
mov     $10000,p
mov     p,ax
push   ax
push   $x
call   proc_a
add    $8,sp
L12:
leave
ret
Lfe3:                                .size     main,Lfe3-main
                                       .comm    x,40000,4
                                       .comm    p,4,4
                                       .ident   "GCC: (GNU) 2.7.2.3"
                                       .align 4
                                       .globl   proc_b
                                       .type     proc_b,@function
                                       .globl   proc_b:
push    bp
mov     sp,bp
sub    $4,sp
push   bx
mov    12[bp],ax
mov    ax,dx
lea   0[,dx,4],ax
mov    8[bp],dx
mov    [dx,ax],ax
mov    ax,-4[bp]
mov    12[bp],ax
mov    ax,dx
lea   0[,dx,4],ax
mov    8[bp],dx
mov    12[bp],cx
mov    cx,bx
lea   0[,bx,4],cx
mov    cx,bx
add   8[bp],bx
lea   4[bx],cx
mov   [cx],bx
mov   bx,[dx,ax]
mov   12[bp],ax
mov   ax,dx
lea   0[,dx,4],ax
mov   ax,dx
add   8[bp],dx
lea   4[dx],ax
mov   -4[bp],dx
mov   dx,[ax]
L1:
mov   -8[bp],bx
leave
ret

```

```

Lfe1:
    .size    proc_b,Lfe1-proc_b
    .align 4
.globl    proc_a
    .type    proc_a,@function
proc_a:
    push    bp
    mov     sp,bp
    sub     $8,sp
    push    si
    push    bx
    nop
    mov     $0,-4[bp]
L3:
    mov     -4[bp],ax
    cmp     ax,12[bp]
    jg     L6
    jmp     L4
    .align 4
L6:
    mov     -4[bp],si
    dec     si
    mov     si,-8[bp]
L7:
    cmp     $0,-8[bp]
    jl     L11
    mov     -8[bp],ax
    mov     ax,dx
    lea    0[,dx,4],ax
    mov     8[bp],dx
    mov     -8[bp],cx
    mov     cx,bx
    lea    0[,bx,4],cx
    mov     cx,bx
    add     8[bp],bx
    lea    4[bx],cx
    mov     [dx,ax],ax
    cmp     ax,[cx]
    jl     L10
    jmp     L11
    .align 4
L11:
    jmp     L8
    .align 4
L10:
    mov     -8[bp],ax
    push    ax
    mov     8[bp],ax
    push    ax
    call   proc_b
    add     $8,sp
L9:
    dec     -8[bp]
    jmp     L7
    .align 4
L8:
L5:
    inc     -4[bp]
    jmp     L3
    .align 4
L4:
L2:
    lea    -16[bp],sp
    pop     bx
    pop     si
    leave
    ret

```