```
////////////////////////////////////////////////////////////////////////
//      var.c program to test memory allocation in C
//         M. Mizuno (c) 1995, 2004, 2005
//          modified for Learning Tree course 223P
//
// to compile var.c on Pentium,
//    $ gcc -O0 -S var.c
////////////////////////////////////////////////////////////////////////
long test(unsigned int ui, int i, short s, unsigned short us,
        char c, unsigned char uc, long l, unsigned long ul,
        int x, short y);

char ret;
int x=100;
static int si;
static int sj = 23;

int main(int argc, char **argv, char **envp)
{
        unsigned char uc;
        static short y = 99;
        short s;
        char c;
        unsigned short us;
        static int i;
        unsigned int ui;
        long l;
        unsigned long ul;

        if (i < 0) {
                ui = us + s - c;
        }
        else {
                ul = si - sj * 2;
        }

        while (sj > 0) {
                uc = y - 3;
                sj++;
        }

        ret = test(ui, i, s, us, c, uc, l, ul, x,  y);

        return 0;
}

long test(unsigned int ui, int i, short s, unsigned short us,
        char c, unsigned char uc, long l, unsigned long ul,
        int x, short y)
{
        char c1;
        int i1;
        char c2;

        ui = 1;
        i = 2;
        s = 3;
        us = 4;
        c = 5;
        uc = 6;
        l = 7;
        ul = 8;
        x = 9;
        y = 10;
        c1 = 11;
        c2 = 12;
        i1 = 13;
        return ui * 2 + l;
}

```



```
        .file   "var.c"
        .comm   _ret, 1, 0
.globl _x
        .data
        .align 4
_x:
        .long   100
.lcomm _si,4,4
        .align 4
_sj:
        .long   23
        .def    ___main;        .scl    2;      .type   32;     .endef
        .text
.globl _main
        .def    _main;  .scl    2;      .type   32;     .endef
_main:
        pushl   %ebp
        movl    %esp, %ebp
        andl    $-16, %esp
        pushl   %edi
        pushl   %esi
        pushl   %ebx
        subl    $116, %esp
        call    ___main
        movl    _i.1238, %eax
        testl   %eax, %eax
        jns     L2
        movzwl  98(%esp), %edx
        movswl  96(%esp), %eax
        addl    %eax, %edx
        movsbl  95(%esp), %eax
        movl    %edx, %ecx
        subl    %eax, %ecx
        movl    %ecx, %eax
        movl    %eax, 104(%esp)
        jmp     L4
L2:
        movl    _sj, %edx
        movl    $0, %eax
        subl    %edx, %eax
        addl    %eax, %eax
        movl    %eax, %edx
        movl    _si, %eax
        leal    (%edx,%eax), %eax
        movl    %eax, 100(%esp)
        jmp     L4
L5:
        movzwl  _y.1234, %eax
        subl    $3, %eax
        movb    %al, 111(%esp)
        movl    _sj, %eax
        addl    $1, %eax
        movl    %eax, _sj
L4:
        movl    _sj, %eax
        testl   %eax, %eax
        jg      L5
        movzwl  _y.1234, %eax
        cwtl
        movl    %eax, 76(%esp)
        movl    _x, %eax
        movl    %eax, 60(%esp)
        movzbl  111(%esp), %edi
        movsbl  95(%esp), %esi
        movzwl  98(%esp), %ebx
        movswl  96(%esp), %ecx
        movl    _i.1238, %edx
        movl    76(%esp), %eax
        movl    %eax, 36(%esp)
        movl    60(%esp), %eax
        movl    %eax, 32(%esp)
        movl    100(%esp), %eax
        movl    %eax, 28(%esp)
```

```
144         movl    88(%esp), %eax
145         movl    %eax, 24(%esp)
146         movl    %edi, 20(%esp)
147         movl    %esi, 16(%esp)
148         movl    %ebx, 12(%esp)
149         movl    %ecx, 8(%esp)
150         movl    %edx, 4(%esp)
151         movl    104(%esp), %eax
152         movl    %eax, (%esp)
153         call    _test
154         movb    %al, _ret
155         movl    $0, %eax
156         addl    $116, %esp
157         popl    %ebx
158         popl    %esi
159         popl    %edi
160         movl    %ebp, %esp
161         popl    %ebp
162         ret
163 .globl _test
164         .def    _test;  .scl    2;      .type   32;     .endef
165 _test:
166         pushl   %ebp
167         movl    %esp, %ebp
168         pushl   %esi
169         pushl   %ebx
170         subl    $68, %esp
171         movl    16(%ebp), %esi
172         movl    20(%ebp), %ebx
173         movl    24(%ebp), %ecx
174         movl    28(%ebp), %edx
175         movl    44(%ebp), %eax
176         movw    %si, -60(%ebp)
177         movw    %bx, -64(%ebp)
178         movb    %cl, -68(%ebp)
179         movb    %dl, -72(%ebp)
180         movw    %ax, -76(%ebp)
181         movl    $1, -12(%ebp)
182         movl    $2, -16(%ebp)
183         movw    $3, -18(%ebp)
184         movw    $4, -20(%ebp)
185         movb    $5, -21(%ebp)
186         movb    $6, -22(%ebp)
187         movl    $7, -28(%ebp)
188         movl    $8, -32(%ebp)
189         movl    $9, -36(%ebp)
190         movw    $10, -38(%ebp)
191         movb    $11, -39(%ebp)
192         movb    $12, -40(%ebp)
193         movl    $13, -44(%ebp)
194         movl    -12(%ebp), %eax
195         leal    (%eax,%eax), %edx
196         movl    -28(%ebp), %eax
197         leal    (%edx,%eax), %eax
198         addl    $68, %esp
199         popl    %ebx
200         popl    %esi
201         popl    %ebp
202         ret
203 .lcomm _i.1238,4,4
204         .data
205         .align 2
206 _y.1234:
207         .word   99
208
```



```
209         .file   "var.c"
210         .comm   _ret, 1, 0      ; allocation 1 byte and name the area _ret
211                                 ; the name is exported
212                                 ; the last "0" specify the alignment
213 .globl _x
214         .data
215         .align 4
216 _x:
217         .long   100
218 .lcomm _si,4,4                  ; allocate 4 bytes at 4-byte boundary
219                                 ; and name the area _si
220                                 ; the name is not exported
221         .align 4
222 _sj:
223         .long   23
224         .def    ___main;        .scl    2;      .type   32;     .endef
225                                 ; ___main is a special function to do necessary
226                                 ; initialization
227                                 ; https://gcc.gnu.org/onlinedocs/gccint/Collect2
    .html
228         .text
229 .globl _main
230         .def    _main;  .scl    2;      .type   32;     .endef
231 ;       int main(int argc, char **argv, char **envp) {
232 ;               unsigned char uc;
233 ;               short s;
234 ;               char c;
235 ;               unsigned short us;
236 ;               unsigned int ui;
237 ;               long l;
238 ;               unsigned long ul;
239
240 _main:
241         pushl   %ebp
242         movl    %esp, %ebp
243         andl    $-16, %esp      ; -16: 0XFFF0   set the lowest 4 bits to 0
244                                 ; align at 8-byte boundary
245                                 ; $-16 : immediate addressing (constant 16)
246         pushl   %edi            ; save the non-destructive registers used
247         pushl   %esi            ; in the function
248         pushl   %ebx
249         subl    $116, %esp      ; allocate areas for the local variables and
250                                 ; the parameters for the functions to call
251                                 ; 116 = 29 * 4
252         call    ___main         ; do necessary initialization
253 ;               if (i < 0) {
254
255         movl    _i.1238, %eax   ; _i.1238: internal static variable i
256         testl   %eax, %eax      ; testl: performs logican AND, but does not
257                                 ; the result in the destination register
258         jns     L2              ; jns: jump if sign bit is not set
259                                 ;   == jge  (L2 is at the beginning of ELSE)
260 ;               ui = us + s - c;
261         movzwl  98(%esp), %edx  ; 98(%esp): us
262         movswl  96(%esp), %eax  ; 96(%esp): s
263         addl    %eax, %edx
264         movsbl  95(%esp), %eax  ; 95(%esp): c
265         movl    %edx, %ecx
266         subl    %eax, %ecx
267         movl    %ecx, %eax
268         movl    %eax, 104(%esp) ; 104(%esp): ui
269         jmp     L4              ; L4: right after the IF statement
270 ;               } else {
271 L2:                             ; at the beginning of ELSE
272 ;               ul = si - sj * 2;
273         movl    _sj, %edx
274         movl    $0, %eax
275         subl    %edx, %eax      ; eax <- eax - edx (eax <- -sj)
276         addl    %eax, %eax      ; eax <- (-sj) * 2
277         movl    %eax, %edx
278         movl    _si, %eax
279         leal    (%edx,%eax), %eax ; eax <- (edx + eax)
280         movl    %eax, 100(%esp) ; 100(%esp) : ul
```

```
281         jmp     L4              ; L4: right after the IF statement
282 L5:
283 ;       {                       ; the body of the WHILE statement
284 ;               uc = y - 3;
285         movzwl  _y.1234, %eax
286         subl    $3, %eax
287         movb    %al, 111(%esp)  ; 111(%esp) : uc
288 ;               sj++;
289         movl    _sj, %eax
290         addl    $1, %eax
291         movl    %eax, _sj
292 ;       }
293 L4:                             ; right after the IF statement
294 ;               while(sj > 0) goto L5
295         movl    _sj, %eax
296         testl   %eax, %eax
297         jg      L5              ; jg: jump if greater than (if true)
298 ;               ret = test(ui, i, s, us, c, uc, l, ul, x, y);
299         movzwl  _y.1234, %eax
300         cwtl                            ; cwtl: convert word to long
301                                         ;  eax <- sign extended ax
302                                         ; eax <- sign extended y
303         movl    %eax, 76(%esp)          ; 76(%esp): temporary area to store
304                                         ; sign extended y
305         movl    _x, %eax
306         movl    %eax, 60(%esp)          ; 60(%esp): temporary area to store x
307         movzbl  111(%esp), %edi         ; 111(%esp): uc
308         movsbl  95(%esp), %esi          ; 95(%esp): c
309         movzwl  98(%esp), %ebx          ; 98(%esp): us
310         movswl  96(%esp), %ecx          ; 96(%esp): s
311         movl    _i.1238, %edx
312         movl    76(%esp), %eax          ; 76(%esp): temporary area to store y
313         movl    %eax, 36(%esp)          ; (equivalent to) push sign extended y
314         movl    60(%esp), %eax          ; 60(%esp): temporary area to store x
315         movl    %eax, 32(%esp)          ; push x
316         movl    100(%esp), %eax         ; 100(%esp): ul
317         movl    %eax, 28(%esp)          ; push ul
318         movl    88(%esp), %eax          ; 88(%esp): l
319         movl    %eax, 24(%esp)          ; push l
320         movl    %edi, 20(%esp)          ; push zero extended uc
321         movl    %esi, 16(%esp)          ; push sign extended c
322         movl    %ebx, 12(%esp)          ; push zero extended us
323         movl    %ecx, 8(%esp)           ; push sign extended s
324         movl    %edx, 4(%esp)           ; push i
325         movl    104(%esp), %eax         ; 104(%esp): ui
326         movl    %eax, (%esp)            ; push ui
327         call    _test
328         movb    %al, _ret
329 ;               return 0;
330         movl    $0, %eax
331 ;       }
332         addl    $116, %esp
333         popl    %ebx
334         popl    %esi
335         popl    %edi
336         movl    %ebp, %esp
337         popl    %ebp
338         ret
339 .globl _test
340         .def    _test;  .scl    2;      .type   32;     .endef
341 ;       void test(unsigned int ui, int i, short s, unsigned short us,
342 ;                         char c, unsigned char uc, long l, unsigned long ul,
343 ;                         int x, short y) {
344 _test:
345         pushl   %ebp
346         movl    %esp, %ebp
347         pushl   %esi
348         pushl   %ebx
349         subl    $68, %esp
350         movl    16(%ebp), %esi  ; 16(%ebp): sign extended s
351         movl    20(%ebp), %ebx  ; 20(%ebp): zero extended us
352         movl    24(%ebp), %ecx  ; 24(%ebp): signe extended c
353         movl    28(%ebp), %edx  ; 28(%ebp): zero extended uc
```



```
354         movl    44(%ebp), %eax  ; 44(%ebp): y
355         movw    %si, -60(%ebp)  ; -60(%ebp): 2-byte temporary area to store s
356         movw    %bx, -64(%ebp)  ; -64(%ebp): 2-byte temporary area to store us
357         movb    %cl, -68(%ebp)  ; -68(%ebp): 1-byte temporary area to store c
358         movb    %dl, -72(%ebp)  ; -72(%ebp): 1-byte temporary area to store uc
359         movw    %ax, -76(%ebp)  ; -76(%ebp): 2-byte temporary area to store y
360 ;               ui = 1;
361         movl    $1, -12(%ebp)   ; -12(%ebp): temporary area for ui
362 ;               i = 2;
363         movl    $2, -16(%ebp)   ; -16(%ebp): temporary area for i
364 ;               s = 3;
365         movw    $3, -18(%ebp)   ; -18(%ebp): temporary area for s
366 ;               us = 4;
367         movw    $4, -20(%ebp)   ; -20(%ebp): temporary area for us
368 ;               c = 5;
369         movb    $5, -21(%ebp)   ; -21(%ebp): temporary area for c
370 ;               uc = 6;
371         movb    $6, -22(%ebp)   ; -22(%ebp): temporary area for uc
372 ;               l = 7;
373         movl    $7, -28(%ebp)   ; -28(%ebp): temporary area for l
374 ;               ul = 8;
375         movl    $8, -32(%ebp)   ; -32(%ebp): temporary area for ul
376 ;               x = 9;
377         movl    $9, -36(%ebp)   ; -34(%ebp): temporary area for x
378 ;               y = 10;
379         movw    $10, -38(%ebp)  ; -38(%ebp): temporary area for y
380 ;               c1 = 11;
381         movb    $11, -39(%ebp)  ; -39(%ebp): temporary area for c1
382 ;               c2 = 12;
383         movb    $12, -40(%ebp)  ; -40(%ebp): temporary area for c2
384 ;               i1 = 13;
385         movl    $13, -44(%ebp)  ; -44(%ebp): temporary area for i1
386 ;               return ui * 2 + l;
387         movl    -12(%ebp), %eax ; -12(%ebp): temporary area for ui
388         leal    (%eax,%eax), %edx
389         movl    -28(%ebp), %eax ; -28(%ebp): temporary area for l
390         leal    (%edx,%eax), %eax
391 ;       }
392         addl    $68, %esp
393         popl    %ebx
394         popl    %esi
395         popl    %ebp
396         ret
397 .lcomm _i.1238,4,4              ; in bss
398         .data
399         .align 2
400 _y.1234:
401         .word   99              ; in data
```