```
;
; Ucitava se niz od N elemenata. Postojece clanove zanemariti.
Sortirati niz tako da bude u opadajucem poretku.
;
;for(int i=0; i<N-1; i++)
;    {
;        for(int j=i+1; j<N; j++)
;        {
;            if(A[ j ] < A [ i ] )
;            {
;                int b=A[ i ];
;                A[ i ]=A[ j ];
;                A[ j ]=b;
;            }
;        }
;    }

%include "asm_io.inc"
segment .data

poruka1 db "Unesite broj N: ",0
poruka2 db "Unesite clan niza ",0
const dd 4
segment .bss

niz resd 100
N resd 1
x resd 1
granica resd 1
temp resd 1

segment .text
        global  _asm_main
_asm_main:
        enter   0,0               ; setup routine
        pusha

           mov eax, poruka1
           call print_string
           call read_int
           mov [N], eax ; ukupan broj clanova
           mov ecx, 0 ; brojac trenutnih clanova niza

           mov eax, poruka2
           call print_string
           call read_int
           mov [niz], eax ; unosimo prvi element
           inc ecx
           mov esi, 4
```

```
     formiranje_niza:
          cmp ecx, [N]
          je sortiranje_niza  ; popunili smo ceo niz, prelazimo na
sortiranje
          mov eax, poruka2
          call print_string
          call read_int
          mov [x], eax ; potencijalni element

          mov edi, 0
          mov ebx, 0 ; novi brojac koji ide od 0 do vrednosti
registra ecx
          ;proveravamo da li je potencijalni element vec deo niza
     postojanje_clana:
          cmp ebx, ecx
          je uvrsti ; potencijalni element se ne nalazi u nizu
          mov eax, [x]
          sub eax, [niz+edi]
          cmp eax, 0

          ; Ukoliko naidjemo na podudarnost nekog postojeceg
elementa niza i potencijalnog,
          ; vracamo se na pocetak petlje formiranje_niza gde se
zahteva unos novog elementa
          je formiranje_niza
          add edi, 4
          inc ebx
          jmp postojanje_clana

     uvrsti:
          mov eax, [x]
          mov [niz + esi], eax
          add esi, 4
          inc ecx
          jmp formiranje_niza

     sortiranje_niza:
          mov eax, [N]
          dec eax
          mov [granica], eax ; granica za prvu for petlju je N-1
          mov ecx, 0 ; ovo je parametar i prve for petlje
     for_petlja1:
          cmp ecx, [granica]
          je kraj
          mov ebx, ecx
          inc ebx ; ovo je parametar j druge for petlje
     for_petlja2:
          cmp ebx, [N]
          je kraj_for_petlja2
          mov eax, ecx ; uzimamo i
          mul dword [const] ; mnozimo sa 4
```

```
            mov esi, eax
            mov eax, ebx ; uzimamo j
            mul dword [const] ; mnozimo sa 4
            mov edi, eax
            mov eax, [niz+esi] ; A [ i ]
            cmp eax, [niz+edi] ; poredimo A [ i ] i A [ j ]
            jb zameni_mesta
            inc ebx ; j++
            jmp for_petlja2
      zameni_mesta:
            mov eax, [niz+esi]
            mov [temp], eax
            mov eax, [niz + edi]
            mov [niz+esi], eax
            mov eax, [temp]
            mov [niz+edi], eax
            inc ebx ; j++
            jmp for_petlja2
      kraj_for_petlja2:
            inc ecx ; i++
            jmp for_petlja1

      kraj:
            mov esi, 0
            mov ebx, 0
            inc ecx ; uvecavamo za 1 jer je ostala vrednost granice,
a to je N-1
      stampanjeniza:
            cmp ebx, ecx
            je krajzadatka
            mov eax, [niz+esi]
            call print_int
            call print_nl
            inc ebx
            add esi, 4
            jmp stampanjeniza

      krajzadatka:


        popa
        mov     eax, 0            ; return back to C
        leave
        ret
```