Arhitektura računara 1

ASEMBLER - 6. termin

Zadatak 1: Formirati Fibonacijev niz od 40 clanova, a zatim u petlji stampati one clanove niza cije indekse korisnik unosi sa tastature, sve dok ne unese 0.

```
; Formirati Fibonacijev niz od 40 clanova, a zatim u petlji stampati one clanove niza
; cije indekse korisnik unosi sa tastature, sve dok ne unese 0.
;
;#include <stdio.h>
;
;int main()
;{
;      unsigned int fib[30];
;      unsigned int index, i;
;
;      fib[0]=1;
;      fib[1]=1;
;
;      for (i=2; i<40; i++)
;            fib[i] = fib[i-2]+fib[i-1];
;
;      printf("Unesite indeks fibonacijevog niza: ");
;      scanf("%u", &index);
;
;      while (index!=0)
;      {
;            printf("Clan Fibonacijevog niza sa unetim indeksom je %d\n", fib[index]);
;            printf("Unesite indeks fibonacijevog niza: ");
;            scanf("%u", &index);
;      }
;
;      return 0;
;}
;
%include "../asm_io.inc"

segment .data
       poruka1             db      "Unesite indeks fibonacijevog niza: ", 0
       poruka2             db     "Clan Fibonacijevog niza sa unetim indeksom je ", 0


segment .bss
       fib          resd    40   ; rezervacija niza od 40 double word clanova, svaki je 4
bajta


segment .text
        global  asm_main
asm_main:
       enter   0,0                     ; rutina za inicijalizaciju
       pusha

       ; Ovde pocinje koristan kod
       mov    dword [fib], 1            ; fib[0]=1
       mov    dword [fib+4], 1    ; fib[1]=1
       mov    ecx, 38                   ; u ecx se nalazi brojac, postavi ga na 38
       mov    esi, 8              ; indeksni registar esi postavi na 8, tj. fib[2]

petlja1:
```

```
	mov	eax, [fib+esi-8]	; eax je jednak fib[i-2], tj. 8 bajtova pre trenutnog clana
	add	eax, [fib+esi-4]	; dodaj na eax vrednost fib[i-1], tj. 4 bajta pre trenutnog
clana
	mov	[fib+esi], eax		; fib[i] = eax (izracunati zbir)
	add	esi, 4			; pomeri indeksni registar za 4 bajta
	loop	petlja1
kraj_petlja1:

	mov	eax, poruka1		; stampanje poruke da se unese indeks
	call	print_string
	call	read_int		; unos indeksa u eax
petlja2:
	cmp	eax, 0			; while (index!=0)
	je	kraj_petlja2		; izadji iz petlje ako je index==0
	mov	esi, eax		; esi=eax
	shl	esi, 2			; pomeri esi za 2 mesta ulevo, tj. pomnozi sa 4, da dobijes
mem. lokaciju clana
	mov	eax, poruka2		; stampaj poruku
	call	print_string
	mov	eax, [fib+esi]		; stampaj odgovarajuci clan niza
	call	print_int
	call	print_nl

	mov	eax, poruka1		; stampanje poruke da se unese novi indeks
	call	print_string
	call	read_int		; ucitavanje indeksa u eax
	jmp	petlja2			; skok na pocetak petlje

kraj_petlja2:

	popa
	mov     eax, 0			; vrati se nazad u C
	leave
	ret
```