Ime:	Luka		Broj indeksa:	18587	
Prezime:	Veličković				
LV po redu:	IV	Termin:		3	
Datum i vreme početka izrade		22.12.2022 15:52			

#### Zadatak:

U segmentu podataka su dve skalarne 16b promenljive i niz 32b elemenata, čiji broj elemenata nije nigde zapamćen, već je poznato da se niz završava nulom. Svi podaci se smatraju označenim brojevima.

Treba napisati program na asemblerskom jeziku koji sve elemente jednake drugom elementu uvećava za vrednost druge skalarne promenljive.

a) Formirati primere početnih vrednosti koji demonstriraju sve osobine zadatog problema, posebno u pogledu graničnih slučajeva, parnosti broja elemenata niza međuprenosa, i za eventualne slučajeve kada nije moguće doći do tačnog rezultata.

Za svaki primer početnih vrednosti:

- navesti po čemu je karakterističan koju osobinu demonstrira
- ustanoviti vrednosti rezultata, ukoliko je to moguće.
- b) **Zadatak 1**: Napisati program koji rešava dati problem. Za pristup elementima niza koristiti indeksno adresiranje. Primere početnih vrednosti iz a) uneti u segmentu podataka kao komplete direktiva za alokaciju memorije, pri čemu su svi kompleti, osim jednog, podešeni kao komentar.

**Zadatak 2**: Napisati program koji rešava dati problem, i ima identičan segment podataka kao Zadatak 1. Za pristup elementima niza u memoriji koristiti isključivo instrukcije za rad s nizovima.

U dati šablon izveštaja, deo b) uneti izveštaj za oba zadatka, u redosledu u kome su zadaci dati ovde.

#### Rešenje:

#### a) Primeri vrednosti, šeme operacija

1) Početne vrednosti u kojima se isti element pojavljuje više puta, uključujući i drugu poziciju u nizu

```
skal1 = 15h

skal2 = 255h

niz = [44h, 55h, 12h, 55h, 55h, -18h, 55h, 67h, 0h]

Očekivana vrednost niza nakon transformacije

niz = [44h, 2AAh, 12h, 2AAh, 2AAh, -18h, 2AAh, 67h, 0h]
```

2) Početne vrednosti pri čemu je nula jedini element niza

```
skal1 = 15h
skal2 = 255h
niz = [0h]
Očekivana vrednost niza nakon transformacije
niz = [0h]
```

3) Početne vrednosti pri čemu je element na drugoj poziciji jedinstven u nizu, a druga skalarna promenljiva negativna

```
skal1 = 15h

skal2 = -255h

niz = [44h, 55h, 12h, 42h, 94h, -18h, 6h, 67h, 0h]

Očekivana vrednost niza nakon transformacije

niz = [44h, -200h, 12h, 42h, 94h, -18h, 6h, 67h, 0h]

Vrednost -200h se u potpunom komplementu predstavlja kao 0FFFFFE00h
```

4) Početne vrednosti za koje se ne dobija tačno rešenje

Vrednosti za koje nije moguće dobiti tačno rešenje predstavljaju one slučajeve kada se pri sabiranju druge skalarne vrednosti i drugog elementa niza bit najveće težine menja, što dovodi do promene znaka, i kada se ona ne treba da se javi. Kako bi se ovo sprečilo potrebno je proširiti vrednosti na 64b, međutim kako je niz 32b ovo ostaje kao situacija u kojoj nije moguće dobiti tačno rešenje.

```
skal1 = 15h

skal2 = EBCDh = (-5171)<sub>10</sub>

niz = [432h, FFFF4582h, 0] // FFFF4582h = (-47742)<sub>10</sub>

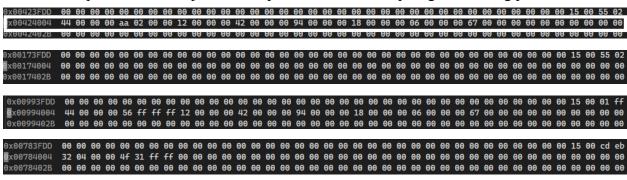
Očekivana vrednost niza nakon transformacije

niz = [432h, -FFFF314h, 0] // -FFFFF314h = (-52913)<sub>10</sub>
```

#### b) Kod rešenja, izveštaj o testiranju

```
1) Korišćenje indeksnog adresiranja za pristup elementima niza
   .386
   .model flat, c
   .data
      skal1 dw 15h
      skal2 dw 255h
      niz dd 44h, 55h, 12h, 55h, 55h, -18h, 55h, 67h, 0h
      ;niz dd 0h
      ;skal2 dw -255
      ;niz dd 44h, 55h, 12h, 42h, 94h, 18h, 6h, 67h, 0h
      ;skal2 dw 0EBCDh
      ;niz dd 432h, 0FFFF4582h, 0
   . code
      main proc
                                   Uvećavanje svih elemenata jednakih drugom
            mov eax, 0
                                   elementu za vrednost druge skalarne promenljive
            mov ax, skal2
                                   ekvivalentno je zameni tih elemenata vrednošću koja
            cwd
                                   se dobija sabiranjem druge vrednosti i druge skalarne
                                   promenljive. Stoga se druga skalarna promenljiva
            shl edx, 16
            or eax, edx
                                               (16b) prvo smešta u AX, proširuje u
                                               DX:AX, a nakon toga vrednost iz DX
                                               smešta u viši deo registra EAX. U
            mov edx, niz[type niz]
                                               registar EDX se potom smešta drugi
            mov ebx, edx
                                               element niza, a nakon toga se vrši
            add ebx, eax
                                               sabiranje (pri čemu se ignoriše
            mov esi, 0
                                               eventualni izlazni prenos). Sada se u
                                               EBX nalazi element koji će biti umetan
            uslov:
            cmp niz[type niz * esi], 0 na odgovarajuća mesta, a u EDX
                                               element sa kojim će se upoređivati.
             ie krai
            cmp niz[type niz * esi], edx
             ine inkrement
            mov niz[type niz * esi], ebx
            inkrement:
            inc esi
                               Osatak problema se rešava tako što se za svaki element
                               proveri da li je terminalni, i da li je jednak drugom
             jmp uslov
                               element niza. Ukoliko je jednak drugom element niza na
            kraj:
                               njegovo mesto se umeće odgovarajuća vrednost, a
            nop
      ret
                               ukoliko nije ne dešava se ništa.
      main endp
      end main
```

Za primere početnih vrednosti pod 1, 2 i 3 dobija se očekivani rezultat, dok se, kao što je navedeno za primer 4 ne dobija. Slike su prikazane redom od prvog do četvrtog primera.



2) Korišćenje instrukcija za rad sa nizovima (.data segment isti kao u prvom delu)

```
.code
  cmp niz, 0
  je kraj
  mov eax, 0
  mov ax, skal2
  cwd
  shl edx, 16
  or eax, edx
  mov edx, eax
  lea esi, niz + type niz
  lodsd
  add edx, eax
  mov ecx, 0
  lea edi, niz
  cld
  uslov:
  xchg eax, ecx
  scasd
  je kraj
  xchg eax, ecx
  sub edi, type niz
  scasd
  ine skok
  sub edi, type niz
  xchg eax, edx
```

stosd

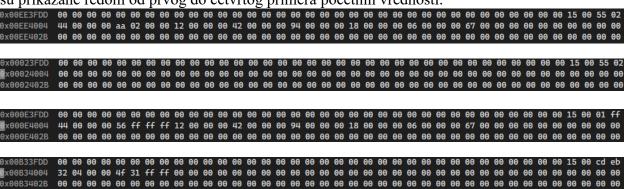
xchg eax, edx

Na početku proveravamo da li je prvi element niza 0, što predstavlja specijalni slučaj. Nakon toga se druga skalarna vrednost pribavlja u EDX. Drugi element niza se pribavlja u akumulator EAX, i ta vrednost se dodaje na vrednost registra EDX. Sada će svi elementi niza koji su jednaki akumulatorom biti zamenjeni vrednošću iz EDX.

Kao što je rečeno, naredbom scasd se vrednost akumulatora (u kome se nalazi originalna vrednost drugog elementa niza) upoređuje sa tekućim elementom odredišnog niza. Ukoliko su jednaki, kako bi se na to mesto ubacila vrednost iz EDX, potrebno je vratiti adresu iz EDI za jedan element unazad, što omogućava zamenu uz pomoć naredbe stosd.

skok: jmp uslov kraj: nop

Izvršavanje ovog programa dovodi do istih ishoda kao i izvršavanje prethodnog programa. Slike su prikazane redom od prvog do četvrtog primera početnih vrednosti.



#### Samoevaluacija

Na skali 0-5 (0 - "nikako", "nimalo"; 5 - "potpuno"), u kom stepenu smatrate da ste:

1)	bili savladali gradivo PRE početka rada na vežbi	4
2)	razumeli zadatak	5
3)	ispunili zahteve zadatka a)	4
4)	ispunili zahteve zadatka b)	4
5)	istestirali i opisali funkcionisanje svog rešenja	5
6)	razumeli ponašanje svog rešenja i pojedinih instrukcija i mehanizama	5
7)	imali dovoljno vremena za vežbu	5
8)	unapredili svoje znanje u toku vežbe	4

<Luka Veličković, 24.12.2022 11:22 >