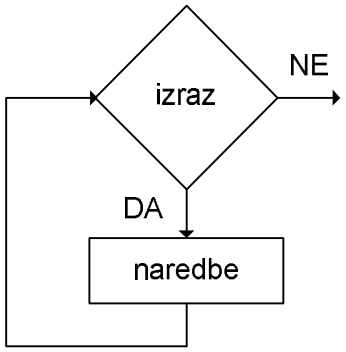
1. Dat je dio koda u programskom jeziku C kojim se vrši sabiranje prvih 100 prirodnih brojeva:

```
s=0;
i=1;
while (i<101)
{
    s=s+i;
    i=i+1;
}
```

Napisati ovaj dio koda u MIPS asemblerskom jeziku pod pretpostavkom da su promjenljivim **s** i **i** dodijeljeni registri $15 i $16, respektivno. Kao privremeni registar koristiti $8.

While petlja principijelno:

```
while(izraz)
{
     naredbe
}
```




```
     add  $15, $0, $0      # s=0;
     addi $16, $0, 1       # i=1;
Loop:slti $8, $16, 101     # Ako je i<101 onda $8=1
     beq  $8, $0, Exit     # Ako je $8=0 izađi iz petlje
     add  $15, $15, $16    # s=s+i;
     addi $16, $16, 1      # i=i+1;
     j  Loop               # go to na Loop
Exit:
```

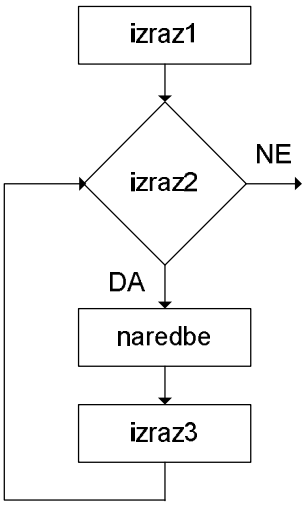
2. Dat je dio koda u programskom jeziku C koji kopira niz **A** u niz **B** i koji broji koliko ima elemenata u nizu A koji su različiti od 0. Dužina niza **A** je 20 elemenata.

```
br=0;
for (i=0; i<20; i++)
{
    B[i]=A[i];
    if (A[i]!=0)
        br=br+1;
}
```

Napisati ovaj dio koda u MIPS asemblerskom jeziku pod pretpostavkom da su početne adrese nizova **A** i **B** - 1000 i 1400, respektivno, kao i da registar $14 sadrži promjenljivu **br**, a registar $15 promjenljivu **i**. Kao privremene registre koristiti $8, $25 i $26.

For petlja principijelno:

```
for(izraz1; izraz2; izraz3)
{
     naredbe
}
```




```
      add  $14, $0, $0      # br=0;
      add  $15, $0, $0      # i=0;
Loop:slti $8, $15, 20       # Ako je i<20 onda $8=1
      beq  $8, $0, Exit     # Ako je $8=0 ($8≠1) izađi iz petlje
      muli $25, $15, 4      # U $25 smještamo 4*i; memorija byte-adresibilna
      lw   $26, 1000($25)   # U $26 smještamo A[i]
      sw   $26, 1400($25)   # $26 smještamo u B[i], B[i]= A[i]
      beq  $26, $0, L1      # Ako je A[i]=0 skačemo na labelu L1
      addi $14, $14, 1      # br=br+1
L1:   addi $15, $15, 1      # i=i+1
      j Loop                # go to na Loop
Exit:
```