**Ime Prezime:**

1. Za idući kôd odrediti T(N) i O(N) (15 bodova):

int f(int \*niz, int n) {

int i = n, sum = 0;

while(i > 0) {

int j = 1;

while(j < n) {

sum += niz[j];

j = j\*2;

}

i--;

}

return sum;

}

1. Procijeniti klasu složenosti O() algoritma koji u nizu pozitivnih cijelih brojeva (dužine N) broji (i ispisuje) koliko puta se pojavljuje broj *b* za svaki *b* od 1 do M. Algoritam se može napisati na dva načina:
2. Niz nije sortiran. Linearno pretražujemo za svaki b.
3. Sortiramo niz uzlazno. Brojanje za svaki idući broj b nastavljamo gdje je prethodno brojanje stalo. U vremensku složenost ulazi i vrijeme sortiranja.

Izraziti složenost u terminima N i M. Koja je opcija bolja ako je M vrlo mali broj u odnosu na N? Koja je opcija bolja ako su N i M približno jednaki?

(15 bodova)

1. Napisati funkciju koja prima jednostruko vezanu listu (svaki element sadrži broj b). Funkcija vraća broj elemenata liste čiji su brojevi jednaki zadnjem broju u listi. Kolika je složenost funkcije O()? (25 bodova)
2. Napisati funkciju koja prima listu. Funkcija iz liste izdvaja u zasebnu listu sve elemente sa pozitivnim brojevima. Funkcija vraća tu novu listu pozitivnih brojeva. Originalna lista mora i dalje biti ispravna vezana lista preostalih brojeva. Kolika je složenost funkcije O()? (35 bodova)
3. Prikazati quicksort algoritam na nizu { 10, 7, 8, 3, 3, 9, 5 }. (10 bodova)