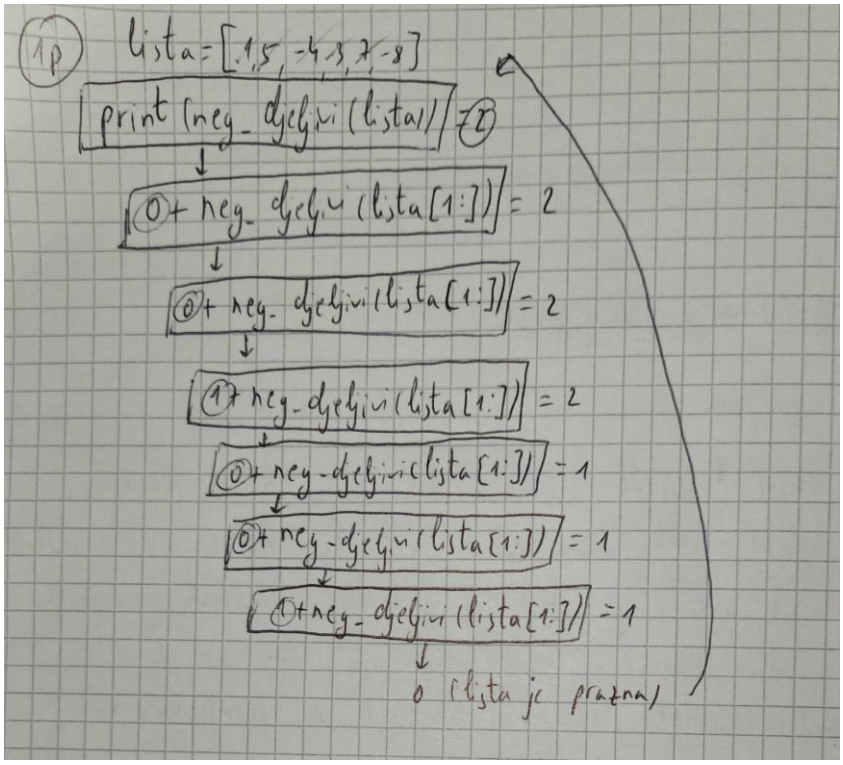
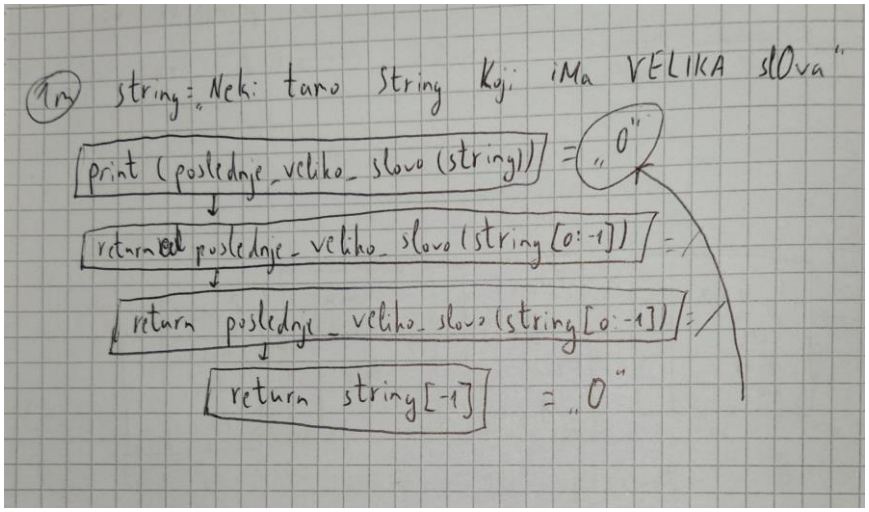
1p.



Funkcija provjerava da li je prvi broj liste negativan i djeljiv sa 2, ako jeste vraca se 1 i ponovo se poziva funkcija koja kao parametar uzima listu od prvog indeksa do kraja, sve dok lista nije prazna. Ako broj ne zadovoljava ove uslove, vraća se 0 + rekurzivna funkcija (neg_djeljivi(lista[1:])).

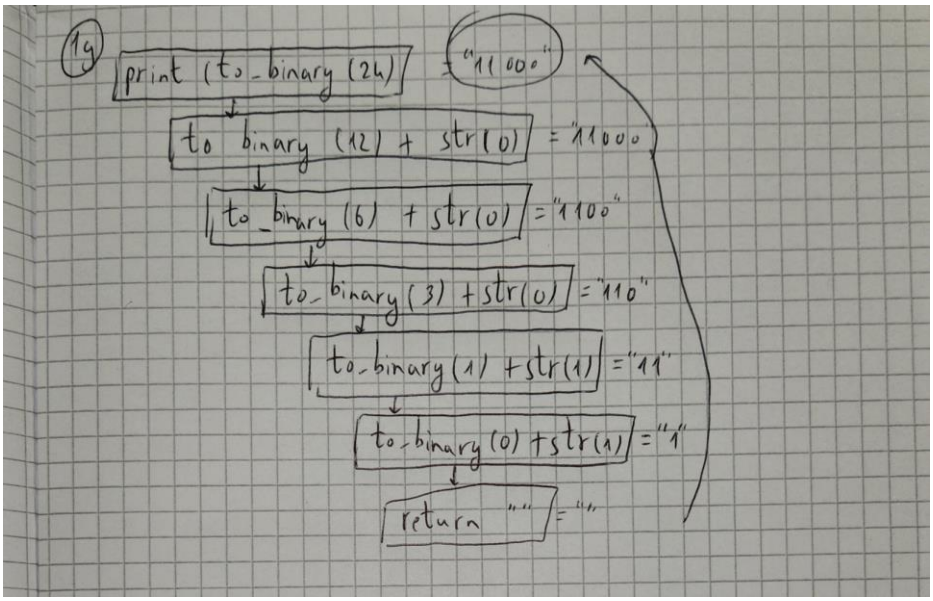
1m.



Ova rekurzivna funkcija prolazi kroz proslijeđeni string i provjerava da li je poslednji element napisan velikim slovom i da li je različit od razmaka ‘ ‘, ako jeste vraća to se slovo vraće, ako

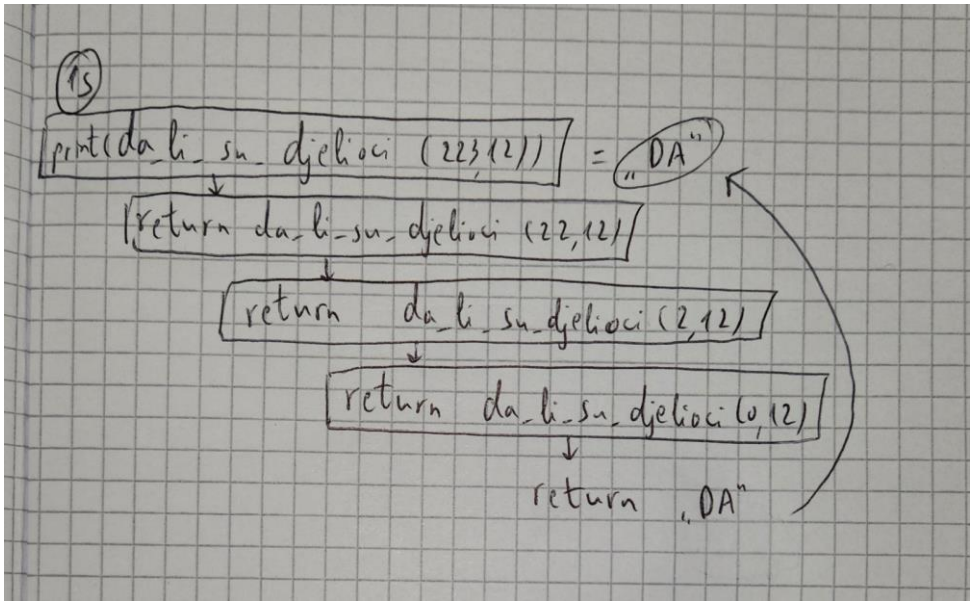
nije opet se poziva rekurzivna funkcija sa stringom koji će imati sve elemente osim poslednjeg koji je provjeren (poslednje_veliko_slovo(str[0:-1])).

1g.



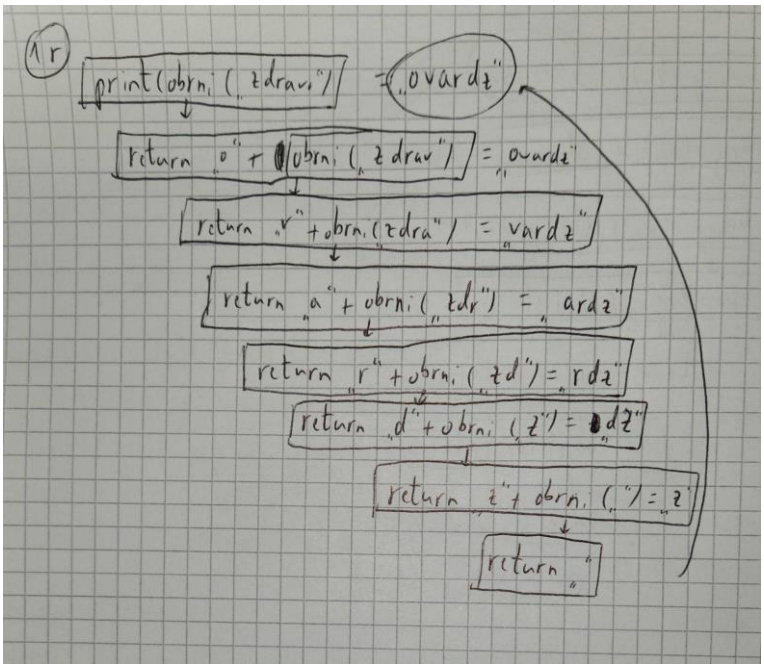
Ova rekurzivna funkcija će da pretvori dekadni broj u binarni tako što gleda da li ima ostatka pri dijeljenju sa 2. Na kraju je potrebno da dobijene cifre (0 ili 1) proslijedimo obratnim redosledom, zbog toga funkcija ide return to_binary(br//2) + str(br%2), da bi se rekurzivna funkcija izvršavala do kraja i da bi se pronašao poslednja cifra, odnosno ona koja će predstavljati prvu cifru binarnog zapisa, pa da se tek onda te cifre nadovezuju pravilnim redosledom. Brojevi se u funkciju pretvaraju u string kako bih mogao da ih nadodam jedan na drugi.

1s.



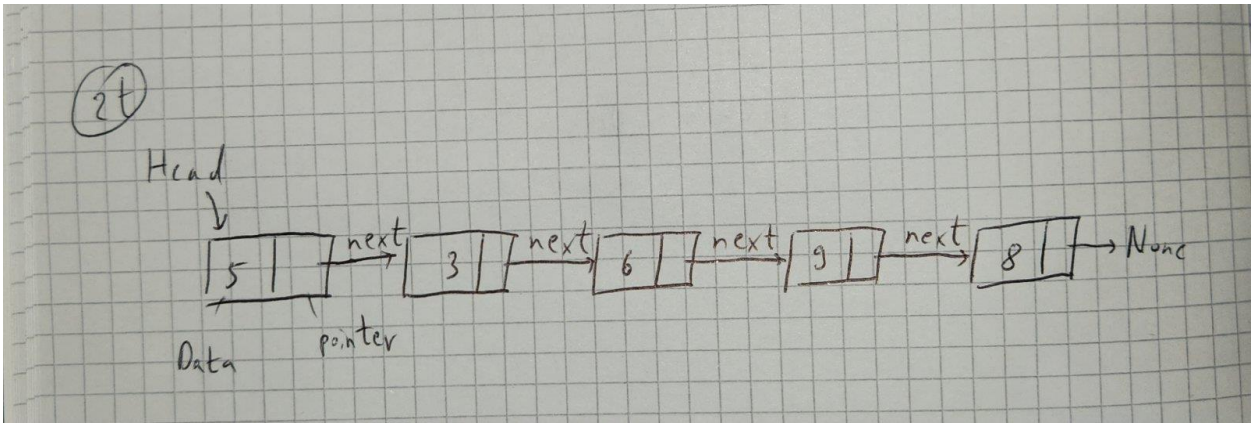
Ova funkcija ispituje da li je broj n djeljiv sa poslednjom cifrom broja m (m%10), ako jesu poziva se opet funkcija, gdje se broj cjelobrojno dijeli sa 10, da bi presli na preostale cifre broja m i to radimo sve dok m nije jednako 0, ako nije bilo problema, vraće se „Da“, ako je tokom izvršavanje funkcije naišao na cifru sa kojom broj n nije djeljiv, vratio bi „Ne“.

1r.



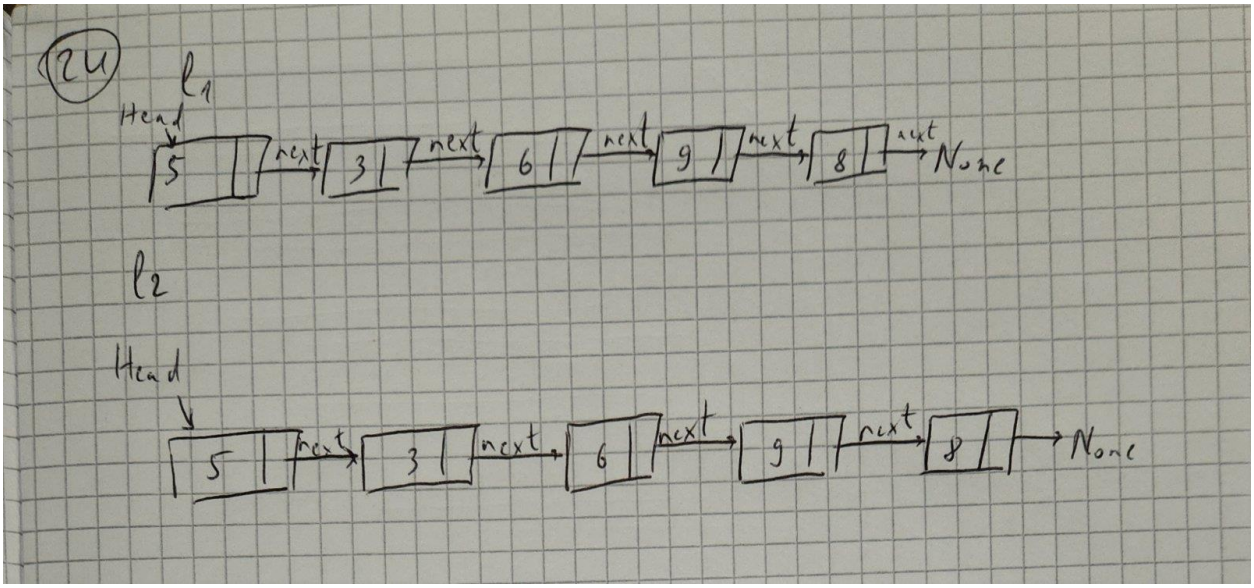
Ova funkcija vraća poslednji karakter iz string, i poziva rekurzivnu funkciju za isti string, samo što sada uzima sve karaktere ne uzimajući poslednji koji je upisan. To radi sve dok je dužina string veća od 0.

2t.



Current uzme node self.head, provjerava da li je vrijednost noda veća od proslijeđene vrijednosti i ako jeste povećava brojač za jedan. Nakon toga, current = current.next, current sada pokazuje na sledeći node i ponavlja korake, sve dok nije None, odnosno sve dok current pokazuje na neki nod u listi. Na kraju funkcije se vraće brojač

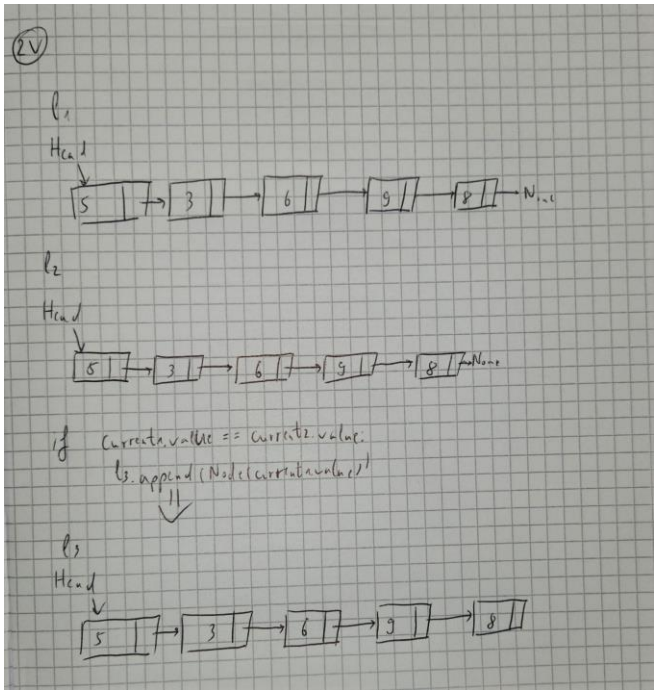
2u.



Current uzima self.head, current2 uzima l2.head, head pokazivač druge liste koja će biti proslijeđena kao parametar. Dok current1 i current2 nisu None, provjerava se da li su njihove vrijednosti iste current1.value == current2.value. Ako su iste, nastavlja do kraja i pokazivač za current1 i current2 prebaca na naredne nodove. Ako su svi elementi bili isti na kraju se vraća „Iste“, a ako neki od elemenata nije bio isti odmah se vratilo „Nisu iste“ i funkcija se prekida.

da_li_su_iste(self,l2)

2v.



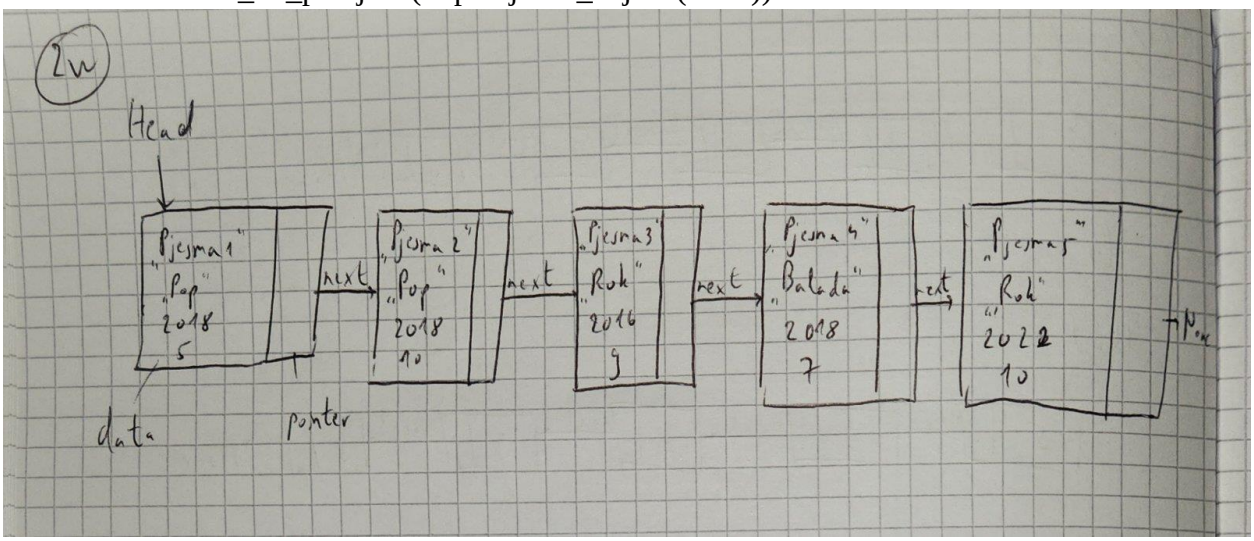
Ova funkcija opet uzima sa current1 i current2 head pokazivač iz dvije proslijeđene liste. Ako nisu None, pita se da li su vrijednosti oba noda iz ove dvije liste jednaki, current1.value == current2.value. Ako jesu, l3 će da appenduje novi nod sa tom vrijednošću, self.append(Node(current1.value)). Current se premješta na naredni nod i ponavlja ovo sve do kraja.

2w.

i. Funkciji se prosleđuje zadataka godina i ona prolazi kroz listu provjeravajući da li je proslijeđena godina jednaka sa godinom pjesme u trenutnom nodu, if god == current.godina, ako jeste na sumu se dodaje ocjena, a brojac se uvećava. Ovo se radi dok se ne izađe iz liste, nakon čega se vraće količnik sume i brojača, odnosno prosjek ocjena za pjesme koje su izašle u naznačenoj godini.

ii. Na isti način se prolazi kroz listu, kao u prethodnim primjerima i provjerava se da li je proslijeđena godina veća od godine za trenutnu pjesmu, if god > current.godina. Ako jeste onda se printa naziv pjesme, current.naziv.

iii. Ovoj funkciji se proslijedila i vrijednost, odnosno prosjek svih pjesama, koristio sam funkciju iz prvog dijela zadataka tako da će ova funkcija printovati naziv pjesama koje su veće od prosjeka za naznačenu godinu (u mom kodu je to godina 2018.), kao što se tražilo u prvom dijelu. if current.ocjena>value, printaće se naziv pjesme. l1.vece_od_prosjeka(l1.prosjecna_ocjena(2018))



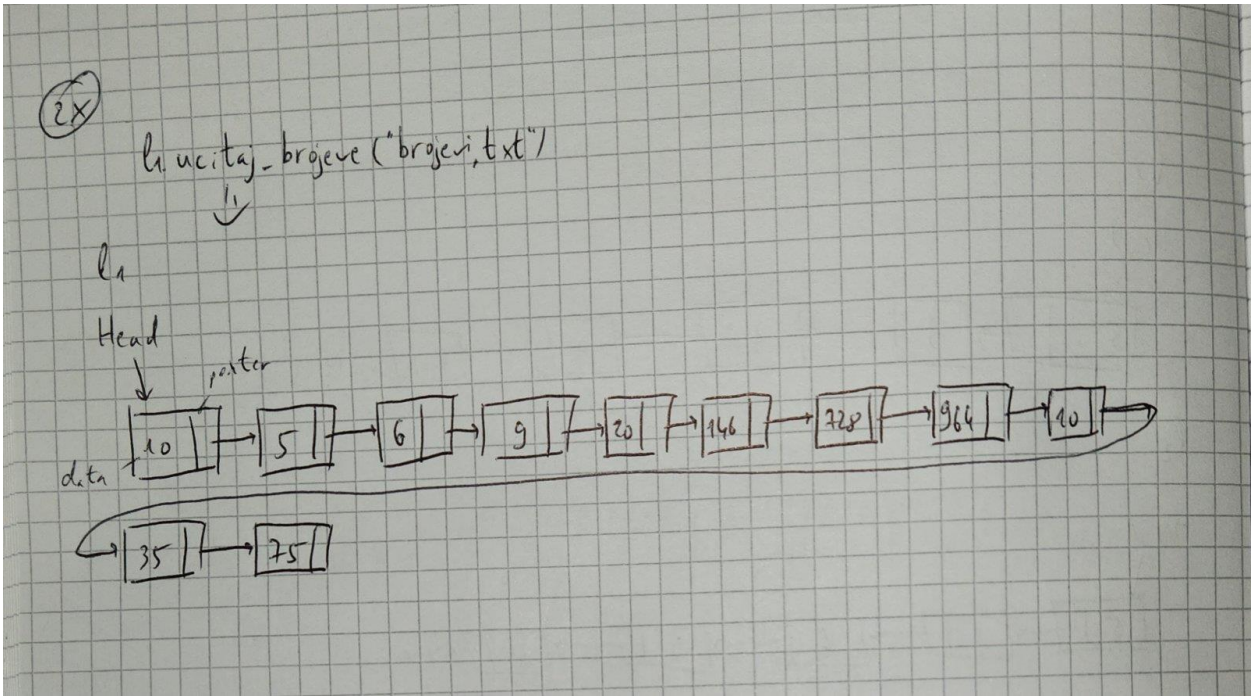
2x.

i. U prvoj funkciji otvara se proslijeđeni file za čitanje, prosljeđuje se file brojevi.txt, koristeći for petlju se prolazi kroz redove u ovom fajlu i lista appenduje Node sa tom vrijednošću self.append(Node(broj.strip()))

ii. Druga funkcija u jednom prolazu određuje maksimalnu rastuću podlistu, koristeći current prolazimo kroz listu. Ako je current.value manji od current.next.value, lista appenduje trenutni tu vrijednost, a brojac se povecava. Zbog problema gdje poslednji element u rastucem podnizu ne bi bio ubrojan u listu, jer bi broj poslije bio manji,

dodao sam if funkciju koja provjerava da li je current.next.value veća od current.next.next.value, ako jeste dodaje se i taj poslednji broj kao sastavni dio liste, jer predstavlja poslednji broj rastućeg podniza.
Ako brojač current.value nije manja od current.next.value, provjerava se da li je brojač veći od brojača koji treba da sadrži broj elemenata najdužeg podniza. Ako jeste veći lista1 (koja sadrži najduži rastući podniz) uzima vrijednost te liste, a brojač i lista se vraćaju na 0/[], nakon prolaza kroz cijelu listu, vraće se lista1.

iii. File ovaj put otvaramo sa „w“, kako bi mogli da pišemo u njemu a da se sav tekst koji se prethodno nalazio potpuno briše. Sa for petljom se prolazi kroz listu koja je argument ove funkcije i ispisuju se brojevi.
l1.ucitaj_brojeve("brojevi.txt")
print(l1.max_rastuca())
l1.upisi_rastuci(l1.max_rastuca(),"Rezultat.txt")



6.

i. Prosjek se zadaje kao parametar i provjerava se da li je current.prosjek (prosjek noda na koji je smješten pokazivač) veći od naznačenog prosjeka. Ako jeste, printaju se imena studenata, kao i broj tih studenata koji imaju prosjek veći od naznačenog.

ii. Za drugi dio zadatka, pravi se funkcija kojoj se kao argument prosljeđuje vrijednost odnosno pozicija, trebaju da se printaju svi studenti iza te pozicije u obrnutom poretku. Koristeći while petlju, current se pomjerao sve dok nije None i sve dok brojac nije dostigao naznačenu poziciju, odnosno index. Tako smo došli do toga da current pokazuje

na onaj nod koji ujedno predstavlja i proslijeđenu poziciju, pa je potrebno da se printaju svi studenti prije te pozicije. Sve dok current.prev nije None, odnosno dok postoji nod koji će prethoditi naznačem nodu, printova će se ime prethodnog studenta, print(current.prev.ime), a current pokazivač će se premjestiti na prethodni node, sa current = current.prev.

l1.izdvoji_studente(8.5)
l1.stampaj_prije_indeksa(5)