Zadatak 1.

Opći član niza $(a_n)_{n\in\mathbb{N}}$ je zadan pravilom $a_n = \dfrac{n^2 + (2\cdot n+1)^2}{(3\cdot n+1)^2 - (2\cdot n-1)^2}$. Izračunajte 70. član, zbroj prvih 100 članova (s točnošću od 10^{-5}) i graničnu vrijednost L zadanoga niza. Potom nađite najmanji $k \in \mathbb{N}$ takav da je $|a_k - L| < 10^{-5}$.

Zadatak 2.

Opći član niza $(b_n)_{n\in\mathbb{N}}$ je zadan pravilom $b_n = \sqrt[4]{\left(1+\dfrac{8}{9\cdot n}\right)^{3\cdot n}}$. Odredite graničnu vrijednost L zadanoga niza, pa nađite najmanji $k \in \mathbb{N}$ takav da je $|b_k - L| < 10^{-9}$.

Zadatak 3.

Opći član niza $(c_n)_{n\in\mathbb{N}}$ je $c_n = \dfrac{n\cdot \operatorname{arctg} n}{\sqrt{n^2+1}}$. Odredite graničnu vrijednost L zadanoga niza, pa nađite najveći $k \in \mathbb{N}$ takav da je $|a_k - L| \geq 10^{-6}$.

Zadatak 4.

Izračunajte $L = \lim\limits_{x\to 1} \dfrac{4\cdot\sqrt{x^2+3}-8}{x^2-1}$.

Zadatak 5.

Izračunajte $L = \lim\limits_{t\to 0+} \dfrac{\sqrt{1-e^{-t}} - \sqrt{1-\cos t}}{\sqrt{\sin t}}$.

Zadatak 6.

Izračunajte $L = \lim\limits_{y\to 2-} \dfrac{1}{1-2^{\frac{1}{y-2}}}$.

Zadatak 7.

Izračunajte $L = \lim\limits_{u\to +\infty} \left(\sqrt{u^2+5} - \sqrt{u^2-2\cdot u+3}\right)$.

Zadatak 8.

Izračunajte $L = \lim\limits_{w \to -\infty} \dfrac{w - \sqrt{w^2 + 3 \cdot w}}{2 \cdot w + 1}$.

Zadatak 9.

Odredite prve tri derivacije funkcije $f(x) = \dfrac{(x-1)^2}{e^{1-x}}$, pojednostavnite njihova pravila najviše što možete, pa izračunajte njihovu vrijednost u točki $c = 1$.

Zadatak 10.

Odredite sve stacionarne točke funkcije $g(t) = t \cdot (\ln^2 t - 3 \cdot \ln t + 3)$.

Rezultati zadataka:

1. $a_{70} = \dfrac{24\ 781}{25\ 200}$, $S_{100} \approx 95.69953$, $L = 1$, $k_{\min} = 119\ 998$.

2. $L = \sqrt[3]{e^2}$, $k_{\min} = 56\ 659\ 286$.

3. $L = \dfrac{\pi}{2}$, $k_{\max} = 1\ 000\ 000$.

4. 1.

5. –1.

6. 1.

7. 1.

8. 1.

9. $f'(x) = \dfrac{x^2 - 1}{e^{1-x}}$, $f'(1) = 0$, $f''(x) = \dfrac{x^2 + 2 \cdot x - 1}{e^{1-x}}$, $f''(1) = 2$, $f'''(x) = \dfrac{x^2 + 4 \cdot x + 1}{e^{1-x}}$, $f'''(1) = 6$.

10. $x = 1$ i $x = e$.

Komentari/objašnjenja programskih kodova

z1.m

U ovom zadatku bavimo se nizovima realnih brojeva. Podsjetimo, niz realnih brojeva je svaka funkcija $f : \mathbb{N} \to \mathbb{R}$. Ta funkcija može (ali ne mora) imati svoju graničnu vrijednost, tj. „konkretan“ realan broj L takav da se u *svakom* otvorenom intervalu oko L nalazi beskonačno mnogo članova niza, a izvan toga intervala konačno mnogo njih.

Niz je funkcija, pa je zadan svojim pravilom. Da bi se istaknula njegova posebnost, varijabla niza označava se sa n. Tako je i u posljednjim trima zadacima ove vježbe. Radi brzine izvođenja programa, pri radu s nizovima treba kombinirati račun s realnim brojevima (tj. račun s varijablama tipa `double`) i račun sa simboličkim varijablama. Preciznije, gdje god budemo mogli, koristit ćemo varijable tipa `double`.

Za određivanje granične vrijednosti funkcija MATLAB ima ugrađenu funkciju `limit`. Argumenti te funkcije su pravilo funkcije (ili, eventualno, oznaka funkcije, pri čemu pravilo funkcije mora biti definirano u istom retku u kojemu je definirana oznaka funkcije), varijabla funkcije i točka u kojoj se računa granična vrijednost. U slučaju granične vrijednosti niza, ta je „točka“ konstanta `Inf` koja u MATLAB-u označava beskonačno veliku veličinu.

Pogledajmo strukturu programskoga koda. Pravilo niza zadano nam je u obliku koji nije do kraja pojednostavljen. Kvadriranje, reduciranje i sređivanje izraza možemo obaviti „ručno“, a možemo ih prepustiti i MATLAB-u. Odlučit ćemo se za potonju varijantu.

U prvom retku koda deklariramo simboličku varijablu (u ovom slučaju ona je označena s n). U drugom i trećem retku zadali smo pravilo brojnika, odnosno nazivnika pravila niza a_n . U četvrtom retku zadajemo pravilo samoga niza, pri čemu koristeći funkciju `simplify` najprije pojednostavljujemo algebarske izraze u brojniku i nazivniku, a potom koristeći ugrađenu funkciju `collect` „skupljamo“ koeficijente uz istu potenciju varijable n i u brojniku i u nazivniku, te tako dodatno reduciramo izraz.

Pravilo kojim zadajemo niz a_n je simbolički objekt. Da bismo izračunali 70. član niza, u to pravilo umjesto varijable n moramo uvrstiti 70. To ćemo napraviti koristeći ugrađenu funkciju `subs`. Rezultat će biti (egzaktni) simbolički broj, tj. dobit ćemo pravu, a ne približnu vrijednost člana a_{70}.

Potom trebamo izračunati zbroj prvih 100 članova s točnošću od 10^{-5}. Dakle, tražimo približnu vrijednost toga zbroja. Zbroj simboličkih brojeva određujemo koristeći ugrađenu funkciju `symsum`. Argumenti te funkcije su redom pravilo ili oznaka izraza čije vrijednosti u pojedinim točkama treba zbrojiti, oznaka varijable dotičnoga izraza, početna vrijednost varijable i krajnja vrijednost varijable. U ovome su slučaju oznaka niza a, oznaka varijable n, početna vrijednost varijable 1, a krajnja vrijednost 100. Rezultat će ponovno biti simbolički broj, i to razlomak s relativno mnogoznamenkastim brojnikom, odnosno nazivnikom. Međutim, mi ionako tražimo približnu vrijednost toga zbroja, pa ćemo navedeni razlomak pretvoriti u „običan“ realan broj koristeći funkciju `double`. Na taj ćemo način dobiti željenu približnu vrijednost.

Naposljetku, graničnu vrijednost zadanoga niza određujemo koristeći ugrađenu funkciju `limit`. Njezin prvi argument je oznaka niza (tj. a), drugi argument oznaka varijable niza (tj. n), a treći argument točka u kojoj računamo graničnu vrijednost (u ovom slučaju konstanta `inf`). Budući da tražimo egzaktnu, a ne približnu vrijednost granične vrijednosti, ovdje nećemo koristiti funkciju `double`.

Preostaje odrediti najmanji $k \in \mathbb{N}$ takav da se svi članovi niza počevši od k-toga nalaze u intervalu širine $2 \cdot 10^{-5}$ oko broja L. Ideja određivanja te vrijednosti je vrlo jednostavna: efektivno računamo apsolutnu vrijednost razlike svakoga člana niza, počevši od prvoga, i broja L. Zbog definicije granične vrijednosti, (ispravno) očekujemo da će se ta razlika sve više i više smanjivati, pa će u nekom trenutku postati manja od 10^{-3} i nastaviti se dalje smanjivati. Tako se zapravo nameće potreba korištenja `while`-petlje: za svaki član niza računajmo navedene apsolutne vrijednosti sve dok su one jednake ili veće od 10^{-3}. Čim prvi put „naletimo“ na vrijednost k takvu da je spomenuta apsolutna vrijednost strogo manja od 10^{-3}, petlja će se prestati izvršavati. Dotična vrijednost k bit će ujedno i rješenje ovoga dijela zadatka.

Posebno istaknimo sljedeće. Određivanje apsolutne vrijednosti u cijelosti možemo provesti sa simboličkim brojevima. Međutim, tada bi izvršavanje cijeloga programskoga koda trajalo prilično dugo. Računanje s varijablama tipa `double` je bitno brže, pa ćemo pribjeći sljedećem lukavstvu.

U 9. retku koda pretvaramo simboličku funkciju u anonimnu funkciju korištenjem ugrađene funkcije `matlabFunction`. Jedini njezin argument je pravilo ili oznaka simboličke funkcije. U ovom slučaju pretvaramo simboličku funkciju a u anonimnu funkciju koju ćemo također označiti s a. Dakle, od 9. retka koda a više nije simbolička, nego anonimna funkcija.

U 10. retku koda simbolički broj L pretvaramo u „običan“ realan broj (varijablu tipa `double`) korištenjem ugrađene funkcije `double`. I na taj ćemo način ubrzati izvođenje koda jer ćemo izbjeći formalno računanje s mješovitim tipovima varijabli (`sym` i `double`).

Retci 11. – 13. predstavljaju implementaciju gore opisane ideje, pa ih ovdje nećemo dodatno komentirati. U retku 14. zadajemo ispisivanje svih triju traženih vrijednosti i time dovršavamo cijeli kod.

Pitanja za razmišljanje: Kako biste odredili prve dvije tražene vrijednosti bez korištenja ijednoga simboličkoga objekta (tj. samo korištenjem varijabli tipa `double`)? Može li se – ponovno samo korištenjem varijabli tipa `double` – naslutiti granična vrijednost niza? Ako može, opišite algoritam kojim bismo mogli naslutiti tu vrijednost.

z2.m i z3.m

U oba zadatka zapravo imamo istu strategiju rješavanja, ali različite izlazne vrijednosti. Najprije zadajemo oznaku simboličke varijable. Potom zadajemo pravilo niza, pa računamo njegovu graničnu vrijednost.

Nakon toga prelazimo na određivanje najmanjega (u 9. zadatku), odnosno najvećega $k \in \mathbb{N}$ koji ima neko svojstvo. Ideja je potpuno analogna kao u rješenju 8. zadatka. Najprije pretvorimo simboličku funkciju u anonimnu, a simbolički broj (jednak graničnoj vrijednosti niza) u „običan" realan broj (varijablu tipa `double`). Krenemo od prvoga člana niza, pa u nizu idemo sve dalje i dalje, i to sve dok je apsolutna vrijednost razlike tekućega člana niza i granične vrijednosti niza jednaka ili veća od zadanoga broja. Prema definiciji granične vrijednosti niza, u nekom će trenutku ta razlika prvi put postati strogo manja od zadanoga broja. Tu nastupaju razlike u posljednjem retku koda. U rješenju 9. zadatka program treba ispisati vrijednost (k) za koju se `while`-petlja prestala izvršavati, dok u rješenju 10. zadatka program treba ispisati posljednju vrijednost k za koju je `while`-petlja izvršena. Kako vidimo, od ključne je važnosti *znati interpretirati što zapravo želimo odrediti*, pa algoritam osmisliti u skladu s tim, a ne znati napamet neke algoritme bez razumijevanja što oni zapravo rade i što ispisuju pripadni programi.

Pitanje za razmišljanje: Mogu li se obje određene prirodne vrijednosti ($k_{\min}$ u 9. zadatku i $k_{\max}$ u 10. zadatku) odrediti bez korištenja simboličkih objekata? Mogu li se te iste vrijednosti odrediti alternativno korištenjem neke druge strukture različite od `while`-petlje (npr. *for*-petlje, *if-then-else* strukture i sl.)? Ako mogu, opišite pripadne algoritme. Sve svoje tvrdnje precizno obrazložite.

z4.m, z5.m., z6.m, z7.m i z8.m

U ovih pet zadataka naučit ćemo određivanje granične vrijednosti funkcija u MATLAB-u. To određivanje zapravo već možemo i naslutiti na temelju rješenja prethodnih triju zadataka. Niz jest posebna vrsta funkcija, ali je ipak funkcija, pa se određivanje granične vrijednosti funkcije u točki zapravo ne razlikuje od određivanja granične vrijednosti niza.

Dogovor: Dalje u tekstu pišemo *granična vrijednost funkcije*, pri čemu mislimo na *graničnu vrijednost funkcije u točki*. (Dakle, izostavljamo točku u kojoj određujemo tu graničnu vrijednost.)

Analitički, granične vrijednosti funkcija određujemo različitim postupcima, npr. L'Hôpital-Bernoullijevim pravilom, racionalizacijom brojnika i sl. U MATLAB-u tih postupaka nema, bez obzira na tip granične vrijednosti koju treba odrediti. *Bilo koju* graničnu vrijednost funkcije određujemo koristeći ugrađenu funkciju `limit`. Međutim, za razliku od prethodnih zadataka, ta će funkcija kao svoju posljednju ulaznu varijablu moći imati i „konkretan“ realan broj, a ne konstantu `Inf`.

Pogledajmo najprije programski kod **z4.m.** U prvom retku deklariramo nezavisnu varijablu (to je x). U drugom retku zadajemo pravilo brojnika, a u trećem retku pravilo nazivnika funkcije pod graničnom vrijednosti. U četvrtom retku zadajemo pravilo same funkciju kao količnik njezina brojnika i nazivnika. Naposljetku, u petom retku određujemo traženu graničnu vrijednost koristeći funkciju `limit`. Prvi argument te ugrađene funkcije je funkcija čiju graničnu vrijednost tražimo, drugi argument je varijabla te funkcije, a treći točka u kojoj tražimo graničnu vrijednost. **Poredak argumenata ne smije se mijenjati**. I to je sve – pokretanjem koda dobit ćemo rješenje zadatka.

Zadatak 5., odnosno kod **z5.m,** razlikuje se od prethodnoga zadatka ponajprije prema tipu granične vrijednosti. (Naravno, i funkcija pod graničnom vrijednosti je različita u odnosu na onu iz zadatka 4.) Naime, u ovom zadatku riječ je o jednostranoj graničnoj vrijednosti, i to posebno o graničnoj vrijednosti zdesna. Zbog toga moramo MATLAB-u eksplicitno reći da tražimo graničnu vrijednost zdesna. Kako to učiniti? Jednostavno: dodajemo četvrti argument ugrađene funkcije `limit`. Taj argument bit će **riječ**: ako tražimo graničnu vrijednost zdesna, upisat ćemo `right`, a ako tražimo graničnu vrijednost slijeva, upisat ćemo `left`. Tu riječ moramo staviti pod znakove jednostrukoga navodnika (apostrofa). Upravo to je i učinjeno u petom retku koda u kojemu je kao posljednji argument funkcije `limit` stavljena riječ `'right'`. **Poredak argumenata** (ponovno) **ne smijete mijenjati**!

Oprez: U ostalim retcima pripazite na nužnost svih stavljenih okruglih zagrada u drugom i trećem retku koda!

Pitanje za razmišljanje: Možemo li u ovom zadatku odrediti graničnu vrijednost slijeva? Postoji li obostrana granična vrijednost? Obrazložite svoje odgovore. (*Uputa*: Promotrite vrijednost radikanda (izraza pod korijenom) u nazivniku za apsolutno male negativne realne brojeve.)

Zadatak 6. je, slobodno govoreći, „zrcalna slika“ zadatka 5. U njemu se traži određivanje granične vrijednosti slijeva. Zbog toga je u posljednjem retku koda **z6.m** kao posljednji argument funkcije `limit` napisana riječ `'left'`.

Oprez: Pripazite na nužnost svih stavljenih okruglih zagrada u drugom retku koda!

Pitanje za razmišljanje: Možemo li u ovom zadatku odrediti graničnu vrijednost zdesna? Postoji li obostrana granična vrijednost? Obrazložite svoje odgovore.

Kod **z7.m** koji predstavlja rješenje zadatka 7. nema potrebe posebno komentirati. Tražimo graničnu vrijednost funkcije kad njezina varijabla teži u $+\infty$. Taj je slučaj sintaktički identičan određivanju granične vrijednosti niza: treća varijabla funkcije limit treba biti jednaka konstanti `inf`. Ostali dijelovi koda su standardni: najprije se deklarira nezavisna varijabla, potom pravilo funkcije i naposljetku izraz za određivanje granične vrijednosti.

Pitanje za razmišljanje: Je li u ovom zadatku svejedno teži li varijabla ka $+\infty$ ili $-\infty$? Dobiju li se isti rezultati u oba slučaja? Pokušajte dati svoje odgovore **bez analitičkoga računanja**, odnosno korištenja MATLAB-a, pa ih potom provjerite koristeći MATLAB. Što primjećujete?

Pitanje za razmišljanje: Rezultat zadatka 7. je „konkretan“ realan broj. Kako biste (svojim riječima) interpretirali njegovo značenje?

Ni kod **z8.m** koji predstavlja rješenje zadatka 8. nema potrebe posebno komentirati. Tražimo graničnu vrijednost funkcije kad njezina varijabla teži u $-\infty$. Prva tri retka koda su standardna: najprije se deklarira nezavisna varijabla, pa potom pravila brojnika, nazivnika i same funkcije pod graničnom vrijednošću. U petom retku se kao posljednji argument funkcije `limit` pojavljuje konstanta `-Inf`. Nju pišemo kad tražimo graničnu vrijednost funkcije čija varijabla teži u $-\infty$.

Pitanje za razmišljanje: Je li u ovom zadatku svejedno teži li varijabla ka $+\infty$ ili $-\infty$? Dobiju li se isti rezultati u oba slučaja? Pokušajte dati svoje odgovore **bez**

analitičkoga računanja, odnosno korištenja MATLAB-a, pa ih potom provjerite koristeći MATLAB. Što primjećujete?

Pitanje za razmišljanje: Rezultat zadatka 8. je „konkretan“ realan broj. Kako biste (svojim riječima) interpretirali njegovo značenje?

z9.m

U ovom ćemo zadatku vidjeti kako možemo odrediti derivaciju neke funkcije u točki, odnosno *bilo koju* (!) derivaciju neke funkcije. Podsjetimo, derivacija funkcije u točki (ako postoji) je „konkretan“ realan broj, dok je derivacija te funkcije ponovno funkcija. Jezični izrazi su vrlo slični, ali u matematičkom smislu radi se o potpuno različitim objektima, pa zbog toga ne smijemo izostaviti riječi *u točki* koje smo smjeli izostaviti kod računanja granične vrijednosti funkcije.

Gotovo sve ugrađene funkcije koje se pojavljuju u ovom kodu upoznali smo već ranije. Funkcija `simplify` pojednostavljuje simbolički algebarski izraz, dok funkcija `subs` računa vrijednost simboličke funkcije u „konkretnoj“ točki. Jedina nova ugrađena funkcija koja se ovdje pojavljuje je funkcija `diff`, pa recimo nešto više o njoj.

Funkciju diff možemo koristiti za određivanje derivacija funkcije jedne realne varijable, ali i tzv. parcijalnih derivacija funkcije barem dviju varijabli. Potonje funkcije bolje ćete upoznati upišete li specijalistički diplomski studij elektrotehnike na TVZ i predmet *Matematika* u 1. semestru toga studija. Ovdje se njima nećemo baviti, nego ćemo se ograničiti na derivacije funkcije jedne realne varijable. Potpuno analogno kao i funkcija `limit`, i funkcija `diff` ima točno tri ulazne varijable na čiji poredak moramo pripaziti. Prva ulazna varijabla je pravilo ili oznaka funkcije koju deriviramo. Druga ulazna varijabla je varijabla te funkcije. Treća ulazna varijabla je *red* derivacije koju tražimo.

Prisjetite se da u *Matematici* 1 nije bilo ni najmanje najmanje jednostavno odrediti npr. 50. derivaciju neke funkcije. U posebnim slučajevima mogli smo izvesti formulu za n-tu derivaciju neke funkcije, i to za svaki $n \in \mathbb{N}$. U nekim slučajevima morali smo primijeniti Leibnizovu formulu. A u ostalim slučajevima morali smo slegnuti ramenima i reći kao nakon poraza našega omiljenoga nogometnoga kluba ili „pada“ na kolokviju iz manje omiljenoga predmeta: „Nažalost, ništa.“ jer je određivanje derivacije bilo kojega reda bilo preteško. No, maloprije je pisalo da MATLAB može odrediti derivaciju bilo kojega reda. Znači li to da profesor koji vam je predavao *Matematiku* 1 nema pojma (pa pod hitno treba tražiti povrat uložene školarine) ili da pisac ovoga teksta bezočno laže tvrdeći da MATLAB može odrediti derivaciju bilo kojega reda?

Ispravan odgovor na prethodno pitanje je: ni jedno, ni drugo. Pitanje je postavljeno namjerno jer, iako u teoriji doslovno vrijedi da kao treću ulaznu varijablu funkcije `diff` možete napisati bilo koji konkretan prirodan broj, vrlo lako je moguće da ćete u većini slučajeva u kojima imate relativno složeno pravilo funkcije koju treba derivirati i relativno velik red derivacije jednostavno „zablokirati“ MATLAB-ov daljnji rad i morati pritisnuti tipke `Ctrl-C` da biste prekinuli izvršavanje koda. Dakle, kao i u životu općenito, ne pretjerujte – u ovom slučaju, sa zadavanjem reda derivacije funkcije.

Pogledajmo strukturu našega programskoga koda. Prva dva retka su nam već dobro poznata: najprije deklariramo simboličku varijablu, a potom pravilo funkcije koju želimo derivirati. U retcima 3. – 5. određujemo prve tri derivacije zadane funkcije. S obzirom da se oznaka ' (apostrof) već koristi u MATLAB-u kao alternativni zapis funkcije transponiranja, prvu derivaciju ne možemo uobičajeno označiti s f'. Ipak, sugestivno je možemo označiti s `f1`, što smo i napravili u 3. retku. Što radi 3. redak koda? Izraz `diff(f,x,1)` određuje pravilo prve derivacije funkcije *f* (po varijabli *x*, ali to je nebitno jer se u pravilu funkcije *f* ne pojavljuje nijedna druga oznaka, konstanta itd.) Ugrađena funkcija `simplify` će potom pojednostavniti dobiveni izraz kako najbolje zna i umije. Time će u cijelosti biti određeno pravilo prve derivacije.

Potpuno analogno postupamo u retcima 4. i 5. U 4. retku određujemo pravilo druge derivacije, dok u 5. retku određujemo pravilo treće derivacije. Oba pravila pojednostavljujemo koristeći funkciju `simplify`. Radi preglednosti, sva tri pravila derivacija pohranjujemo u matricu *F* tipa (3,1) i ispisujemo je u 6. retku koda.

Kad znamo pravila prvih triju derivacija funkcije *f*, lako je izračunati vrijednost tih derivacija u konkretnoj točki. Jednostavno, u svako pravilo umjesto varijable *x* treba uvrstiti konkretnu točku. U tu će nam svrhu poslužiti funkcija `subs`. U retcima 7. – 9. primjenom upravo te funkcije izračunate su vrijednosti prvih triju derivacija u zadanoj točki. Radi preglednosti, sve tri vrijednosti pohranjene su u matricu *C* tipa (1, 3), te se ispisuju u 10. retku koda.

Pitanje za razmišljanje: Ranije smo vidjeli da ugrađena funkcija „napada“ matricu tako da djeluje na svaki element te matrice. Ako matricu $X = \begin{bmatrix} 1 & 500 & 600 \end{bmatrix}$ „napadnemo“ funkcijom `diff`, što očekujemo kao rezultat? (*Uputa*: Što je prva derivacija svake konstantne funkcije?) A što je *zapravo* `diff(X)`? Provjerite rješenje koristeći MATLAB.

z10.m

U posljednjem zadatku ove vježbe ključno je prisjetiti se strategije analitičkoga rješavanja ovoga tipa zadatka. Dakle, kako bismo ovaj zadatak riješili analitički? Podsjetimo, stacionarna točka funkcije g je svaka nultočka njezine prve derivacije. Samim tim slijedi strategija rješavanja zadatka. Najprije ćemo zadati pravilo funkcije g, potom odrediti pravilo njezine prve derivacije i potom riješiti jednadžbu $g'(t)=0$. Khm, khm, je li ovaj posljednji korak u cijelosti ispravan? Jesu li sva rješenja jednadžbe $g'(t)=0$ ujedno i nultočke funkcije g'? Nisu, naravno, kao što smo to istaknuli rješavajući vježbu 7. Tražene nultočke bit će samo ona rješenja koja pripadaju prirodnoj domeni *polazne funkcije g*.

Prirodnu domenu funkcije g odredimo napamet. Prirodni logaritam postoji samo za strogo pozitivne realne brojeve. Dakle, rješenje zadatka bit će ona rješenja jednadžbe $g'(t)=0$ koja su strogo pozitivni realni brojevi. Sva rješenja jednadžbe općenito su kompleksni brojevi. Zbog toga moramo smisliti logički uvjet kako provjeriti je li neki kompleksan broj ujedno i strogo pozitivan realan broj. Taj uvjet nije težak i glasi:

$$(z>0) \Leftrightarrow ((\operatorname{Re} z>0) \wedge (\operatorname{Im} z=0)).$$

Oprez: U matematici je sasvim dovoljno napisati $z>0$ da bismo zaključili da z ne može biti kompleksan broj čiji je imaginarni dio različit od nule. Naime, u *Matematici* 1 smo naučili da kompleksne brojeve ne možemo uspoređivati „po veličini“ onako kako to radimo s realnim brojevima. Preciznije, za dva kompleksna broja ne možemo reći koji je veći, a koji manji. Možemo samo ustvrditi jesu li oni međusobno jednaki ili su međusobno različiti. No, u MATLAB-u postoji problem jer vrijedi:

$$(z>0) \Leftrightarrow (\operatorname{Re} z>0).$$

Npr. ako je $z=1+i$, onda će izvršavanjem `z>0` MATLAB vratiti 1, tj. potvrdit će nam da je ta nejednakost točna iako je ona matematički besmislena. Zbog toga moramo koristiti gornju ekvivalenciju.

Preostaje nam još navesti ugrađenu funkciju koja će nam poslužiti kao rješavač jednadžbi. Riječ je o funkciji `solve`. Ta funkcija može rješavati (ne)algebarske jednadžbe, sustave (ne)algebarskih jednadžbi itd. Zbog toga je njezina sintaksa prilično raznolika. Ovdje ćemo koristiti sintaksu u skladu s kojom je prva ulazna varijabla jednadžba koju treba riješiti, a druga oznaka nepoznanice.

Pogledajmo sada strukturu našega koda. Prva dva retka su jasna: najprije deklariramo ime varijable, a potom zadamo pravilo funkcije g. U trećem retku određujemo rješenja jednadžbe $g'(t)=0$. Skup svih tih rješenja označimo s `rjesenja`. Potom pišemo znak =. Nakon toga pozivamo funkciju `solve`. Ona će riješiti jednadžbu $g'(t)=0$ u skupu kompleksnih brojeva. Kako zadati tu jednadžbu?

Jednostavno: prvu derivaciju zadamo koristeći netom naučenu funkciju `diff`. Potom napišemo znakove == kojima ćemo označiti da je riječ o jednadžbi, da je „ono što piše ispred tih znakova“ lijeva strana jednadžbe, a da je „ono što piše iza tih znakova“ desna strana jednadžbe. Zašto **ne smijemo** napisati samo jedan znak =? Razlog je jednostavan: MATLAB će u tom slučaju nastojati dodijeliti objektu ispred znaka = vrijednost iza toga znaka, a to je nemoguće jer je objekt ispred znaka = veličina definirana ugrađenom funkcijom, a ne varijabla. Preostaje iza znakova == napisati 0. Time je jednadžba koju treba riješiti potpuno zadana. Preostaje upisati zarez, te zaključno oznaku nepoznanice (t).

Da nam odgovara bilo kakvo *kompleksno* rješenje jednadžbe, posao bi bio gotov. Međutim, rekli smo da kao rješenje zadatka u obzir dolaze samo strogo pozitivni brojevi. Zbog toga u posljednjem retku koda moramo implementirani ranije navedeni logički uvjet koji će među svim kompleksnim rješenjima jednadžbe prepoznati samo ona koja su strogo pozitivni realni brojevi i upravo ta rješenja pohraniti u matricu `rezultat`.

Zaključno primijetimo nešto drugačiji oblik korištenja logičkoga operatora AND. Ranije smo taj operator pisali u „dvostrukom“ zapisu &&. Međutim, ovdje je zapisan u „jednostrukom“ zapisu &. Razlog ovoj promjeni je vrlo jednostavan. U MATLAB-u postoje dva logička operatora AND. Prvi (&) je operator koji djeluje na matrice prema standardnom načelu „član po član“: uspoređuje prvi član prve matrice s prvim članom druge matrice, drugi član prve matrice s drugim članom druge matrice itd. Drugi (&&) je operator koji uspoređuje skalare (realne ili kompleksne brojeve). S obzirom da je varijabla `rjesenje` zapravo matrica sa simboličkim objektima, primijenjen je prvi oblik toga operatora.

Domaća zadaća

1. Opći član niza $(a_n)_{n\in\mathbb{N}}$ definiran je pravilom $a_n = \frac{(-1)^n}{n}$. Odredite 100. član, zbroj prvih 200 članova (s točnošću od 10^{-5}) i graničnu vrijednost zadanoga niza. Potom odredite najmanji $k \in \mathbb{N}$ takav da je $|a_k - L| < 10^{-5}$.

2. Opći član niza $(b_n)_{n\in\mathbb{N}}$ definiran je pravilom $b_n = \frac{(6\cdot n-5)^2+(8\cdot n+7)^2}{(26\cdot n+25)^2-(24\cdot n-23)^2}$. S točnošću od 10^{-5} odredite 50. član, zbroj prvih 100 članova i graničnu vrijednost zadanoga niza. Potom odredite najveći $k \in \mathbb{N}$ takav da je $|a_k - L| \geq 10^{-5}$.

3. Odredite sljedeće granične vrijednosti:

a) $L_1 = \lim\limits_{x\to 1} \frac{(1+x)^x - 2}{x-1}$;

b) $L_2 = \lim\limits_{t\to 0-} \left(e^{\frac{1}{2-\frac{3}{t}}} \right)$;

c) $L_3 = \lim\limits_{y\to 2+} \left(2\cdot \operatorname{arctg}\left(\frac{1}{2-y} \right) \right)$;

d) $L_4 = \lim\limits_{w\to -\infty} \left(\sqrt{w^2 - 2\cdot w + 2} + w \right)$.

4. Funkcija $f : \mathbb{R} \to \mathbb{R}$ definirana je pravilom $f(x) = x\cdot \sin x$. Odredite $f^{(2020)}(0)$.

5. Odredite skup svih stacionarnih točaka polinoma $p(t) = 4\cdot t^5 + 5\cdot t^4 - 10\cdot t^2 - 20\cdot t + 1$.

Rezultati zadataka za domaću zadaću

1. $a_{100} = \frac{1}{100}$, $S_{200} = -0.69065$, $L = 0$, $k_{\min} = 100\ 001$.

2. $a_{100} = 0.68236$, $S_{200} = 61.87193$, $L = 1$, $k_{\max} = 2\ 351\ 975$.

3. **a**) $L_1 = 2\cdot \ln 2 + 1$; **b**) $L_2 = 1$; **c**) $L_3 = -\pi$; **d**) $L_4 = -1$.

4. –2020.

5. $S = \{-1, 1\}$.