Potrebno predznanje:

- elementarne funkcije;
- algebarski zapis kompleksnoga broja;
- osnovne algebarske operacije s kompleksnim brojevima.

Zadatak 1.

Izračunajte vrijednost brojevnoga izraza $\left(\left(\frac{3}{4}+\left(-\frac{2}{7}\right):\frac{4}{21}\right)\cdot\left(-\frac{4}{3}\right)-1\right)^{2021}+1$.

Zadatak 2.

Izračunajte vrijednost brojevnoga izraza $\sqrt[3]{\mathrm{e}^{\pi}-\pi^{\mathrm{e}}}+\sqrt{\log_2 5+\log^2 6}$ i zaokružite dobiveni rezultat s točnošću od 10^{-5}.

Zadatak 3.

Zadan je broj $z=e^{i\cdot\frac{\pi}{12}}$. Izračunajte $\left(\sqrt{2}+1\right)\cdot\left|\left(2\cdot\mathrm{Re}\left(z^3\right)-\mathrm{Im}\left(z^6\right)\right)\cdot\overline{z^9}\right|$.

Zadatak 4.

Zadani su brojevi $z_1=\sqrt{3}-i$ i $z_2=\mathrm{cis}\left(\frac{4}{3}\cdot\pi\right)$. Izračunajte modul broja $z_3=\frac{z_1+z_2}{z_1-z_2}$.

Zadatak 5.

Izračunajte $\dfrac{\sin\left(\frac{\pi}{8}\right)+2\cdot\sin\left(\frac{\pi}{4}\right)+\sin\left(\frac{3}{8}\cdot\pi\right)}{\cos\left(\frac{\pi}{8}\right)+2\cdot\cos\left(\frac{\pi}{4}\right)+\cos\left(\frac{3}{8}\cdot\pi\right)}-\dfrac{\mathrm{ctg}\left(\frac{\pi}{8}\right)+2\cdot\mathrm{ctg}\left(\frac{\pi}{4}\right)+\mathrm{ctg}\left(\frac{3}{8}\cdot\pi\right)}{\mathrm{tg}\left(\frac{\pi}{8}\right)+2\cdot\mathrm{tg}\left(\frac{\pi}{4}\right)+\mathrm{tg}\left(\frac{3}{8}\cdot\pi\right)}$.

Zadatak 6.

Izračunajte $\left(\arcsin\left(\frac{\sqrt{3}}{2}\right)-2\cdot\arccos\left(-\frac{1}{2}\right)+3\cdot\mathrm{arctg}\left(\sqrt{3}\right)-4\cdot\mathrm{arcctg}\left(-\frac{\sqrt{3}}{3}\right)\right)^{-1}\cdot\left(-\frac{4}{3}\cdot\pi\right)$.

Zadatak 7.

Izračunajte $\sqrt{\left\lfloor \mathrm{sh}\ 1+2\cdot\mathrm{ch}\ 2+3\cdot\mathrm{th}\ 3+4\cdot\mathrm{cth}\ 4\right\rfloor+\mathrm{ch}\ 0}-\left\lceil \mathrm{arsh}\ 1+2\cdot\mathrm{arch}\ 2+3\cdot\mathrm{arth}\left(-\frac{1}{2}\right)+4\cdot\mathrm{arcth}\ 4\right\rceil$.

Zadatak 8.

<u>Bez ispisa vrijednosti broja</u> $a=\mathrm{tg}\left(\frac{\pi}{4}\right)$ ispitajte valjanost jednakosti $a=1$ i $1+a=2$. Što zaključujete?

Zadatak 9.

<u>Bez ispisa vrijednosti brojevnoga izraza</u> odredite cijeli broj najbliži realnom broju $3^{\log_2^4\left(\frac{1}{3}\right)} - 2^{\log^3 5} + e^{\sin^2 15^\circ}$.

Zadatak 10.

Napišite logički uvjet prema kojemu neki realan broj x pripada skupu **nenegativnih** cijelih brojeva. (*Pomoć*: Koristite funkciju apsolutne vrijednosti, te neku od ugrađenih funkcija za zaokruživanje realnih brojeva.)

Rezultati zadataka

1. 1.

2. 2.59101.

3. 1.

4. 1.

5. 0.

6. –1.

7. 1.

8. Prva jednakost nije istinita. Druga jednakost je istinita.

9. 1025.

10. Npr. $\lfloor x \rfloor = |x|$.

Pojašnjenja programskih kodova

z1.m

U ovom zadatku trebamo se prisjetiti redoslijeda izvođenja osnovnih aritmetičkih operacija (zbrajanje, oduzimanje, množenje, dijeljenje, potenciranje), a posebno redoslijeda izvođenja operacija jednakoga ranga i redoslijeda izvođenja operacija naznačenih unutar zagrada.

Operacija najvećega ranga je potenciranje, potom slijede množenje i dijeljenje (u istom rangu), te naposljetku zbrajanje i oduzimanje kao operacije najslabijega ranga.

Također, ako je neka operacija naznačena unutar okrugle zagrade, ona se izvodi prva. Pritom istaknimo važno pravilo: **uglate i vitičaste zagrade se ne koriste u sastavu aritmetičkih izraza jer u MATLAB-u imaju sasvim drugo značenje** (o kojemu ćemo nešto više reći u kasnijim vježbama).

Pogledajmo redoslijed izvođenja aritmetičkih operacija u ovom zadatku. Najprije računamo izraz $\frac{3}{4}+\left(-\frac{2}{7}\right):\frac{4}{21}$. Tu se krije „zamka“ jer znak razlomačke crte zapravo označava znak dijeljenja. Zbog toga je ovaj izraz zapravo jednak $\frac{3}{4}+\frac{-\frac{2}{7}}{\frac{4}{21}}$. Drugi pribrojnik moramo zadati tako da njegov nazivnik napišemo unutar zasebnih zagrada jer ćemo u suprotnom dobiti netočan rezultat.

- *Pitanja za razmišljanje*: Što bi se dogodilo ako biste isti izraz napisali bez ijedne zagrade? Kako biste taj izraz pojednostavnili najviše što možete? A što bi se dogodilo ako biste napisali (-2/7)/(4/21), tj. ako biste i brojnik i nazivnik drugoga pribrojnika napisali unutar zagrada? Zbog čega prvi pribrojnik nije potrebno zapisati unutar zagrada?

Nakon što smo izvršili obje operacije unutar zagrada i izračunali vrijednost pripadnoga aritmetičkoga izraza, dobiveni rezultat množimo s $-\frac{4}{3}$, od umnoška oduzimamo 1, dobivenu razliku potenciramo na eksponent 2021 i dobivenoj potenciji dodajemo 1. Time je kod upotpunjen. Primijetite da za potenciranje koristimo znak ^ koji se dobije istovremenim pritiskom tipaka AltGr i 3 (i tipke Space nakon toga).

z2.m

U ovom zadatku upoznajemo neke ugrađene konstante i funkcije u MATLAB-u. Istaknimo vrlo važno pravilo: **konstanta *e* (baza prirodnoga logaritma) nije implementirana u MATLAB-u kao zasebna konstanta, nego njezinu vrijednost računamo koristeći ugrađenu funkciju** `exp`**.** Zbog toga smo u prvom retku koda *deklarirali varijablu e* čija je vrijednost `exp(1)`, tj. $e^1 = e$. Uočite pravilo pozivanja ugrađene funkcije u MATLAB-u: **napiše se naziv funkcije, pa se argument(i) funkcije navede unutar zagrada.**

Istaknimo i da znak jednostruke jednakosti u prvom retku koda označava da smo varijabli označenoj s *e* dodijelili vrijednost `exp(1)`. Kasnije ćemo vidjeti upotrebu znaka dvostruke jednakosti.

Konstanta π je ugrađena konstanta u MATLAB-u i poziva se jednostavno navođenjem `pi`. Dakle, nju ne moramo posebno deklarirati.

Jedina ugrađena funkcija u MATLAB-u koja se odnosi na korijene je funkcija `realsqrt`. Pozivanjem te funkcije određujemo drugi korijen iz nenegativnoga realnoga broja. Ostale (treći, četvrti, …) korijene možemo odrediti na dva načina:

- korištenjem ugrađene funkcije `nthroot` koja ima točno dva argumenta: prvi je radikand (broj pod korijenom), a drugi eksponent (izraz „na“ korijenu). Pritom **moramo paziti na poredak argumenata**: najprije se navodi radikand, a potom eksponent.
- korištenjem zapisa $\sqrt[n]{x^m} = x^{\frac{m}{n}}$, tj. zapisivanjem korijena kao potencije s racionalnim eksponentom.

U slučaju kad je radikand nešto složeniji aritmetički izraz prikladniji je drugi način, pa je on primijenjen u ovom kodu. Pritom istaknimo da je rezultat funkcija `realsqrt` i `nthroot` nužno **realan broj** (ako postoji).

- *Pitanje za razmišljanje*: „Preradite“ zadani kod tako da treći korijen izračunate koristeći ugrađenu funkciju `nthroot`. (Njezin drugi argument je 3.) Također, u nova dva retka komandnoga prozora upišite sljedeće izraze:

  ```
  realsqrt(-1)
  nthroot(-16,1/4)
  ```

  izvršite ih i objasnite što se dogodilo.

Napomenimo da u MATLAB-u postoji ugrađena funkcija `sqrt` koja određuje i drugi korijen iz kompleksnoga broja. Ona se „podudara“ s funkcijom `realsqrt` ako i samo ako je argument obiju funkcija nenegativan realan broj. Međutim, ako je argument funkcije `sqrt` strogo negativan realan broj ili čak kompleksan broj, onda ta funkcija najprije zapisuje svoj argument u *trigonometrijskom obliku*, pa vraća vrijednost drugoga korijena dobivena pomoću *de Moivréove formule za korjenovanje*. Pritom treba pripaziti: ta je vrijednost kompleksan broj s *najmanjim argumentom*, pri čemu taj argument nužno pripada segmentu $[-\pi,\pi]$. Takav pristup je i matematički potpuno korektan. U *Matematici* 1 smo pretpostavljali da argument kompleksnoga broja pripada intervalu $\left[0,2\cdot\pi\right\rangle$ i taj smo argument zvali *glavni argument* kompleksnoga broja. Međutim, u MATLAB-u argument kompleksnoga broja općenito nije jednak glavnomu argumentu toga broja – preciznije, argument kompleksnoga broja podudara se s glavnim argumentom toga broja ako i samo ako je $\varphi\in[0,\pi]$. Oba argumenta su pritom iskazana u *radijanima*.

- *Pitanja za razmišljanje*: Argument kompleksnoga broja u MATLAB-u određuje ugrađena funkcija `phase`. U nova dva retka komandnoga prozora upišite

  ```
  phase(i)
  phase(-i)
  ```

  izvršite te funkcije i objasnite dobivene rezultate. Usporedite ih s rezultatima koje bismo dobili analitički u *Matematici* 1.

Naposljetku treba spomenuti da u MATLAB-u postoji nekoliko ugrađenih funkcija koje računaju logaritme strogo pozitivnih realnih brojeva. To su `log`, `log2`, `log10` i `reallog`. Pritom argumenti prvih triju funkcija mogu biti i kompleksni brojevi, dok argument četvrte funkcije nužno mora biti strogo pozitivan. Te ugrađene funkcije određuju redom *prirodni* (matematička oznaka: ln) *binarni* (matematička oznaka: $\log_2$) i *dekadski* (matematička oznaka: log) logaritam iz strogo pozitivnoga realnoga broja. Istaknimo da u MATLAB-u **ne postoji ugrađena funkcija koja računa logaritam strogo pozitivnoga realnoga broja po bilo kojoj bazi** (strogo pozitivnoj i različitoj od 1), ali je relativno laka zadaća u MATLAB-u *napisati vlastitu funkciju* čiji su argumenti strogo pozitivan realan broj a i strogo pozitivan realan broj b različit od 1 (u navedenom redoslijedu), a koja vraća $\log_b a$. Pritom se koristi identitet (dokažite ga!):

$$\log_b a=\frac{\log a}{\log b}.$$

Obratite posebnu pozornost na način zadavanja člana $\log^2 6$. Prilikom računanja toga člana najprije računamo vrijednost $\log 6$ (dakle, *dekadski*(!) logaritam broja 6), a potom dobiveni rezultat kvadriramo. Zbog toga u kodu zapisujemo `log10(6)^2`. Redoslijed izvršavanja ugrađenih funkcija u tom članu u potpunosti odgovara gornjem redoslijedu: najprije se izvršava ugrađena funkcija `log10` čiji je jedini argument broj 6, a potom potenciranje.

- *Pitanja za razmišljanje*: U novi redak komandnoga prozora upišite

  ```
  log10^2(6)
  ```

  izvršite taj redak i objasnite što se dogodilo. Potom u novi redak upišite

  ```
  log10(6^2)
  ```

  izvršite taj redak i objasnite što je MATLAB izračunao. Kako biste izračunali vrijednost posljednjega izraza bez korištenja potenciranja unutar zagrada?

z3.m

U ovom zadatku upoznajemo osnovne MATLAB-ove operacije u radu s kompleksnim brojevima. Prije njih ukratko podsjetimo da se svaki kompleksan broj (različit od nule) može zapisati u algebarskom, trigonometrijskom i eksponencijalnom obliku. Za zapis u algebarskom obliku moramo znati realni i imaginarni dio kompleksnoga broja, dok za zapis u trigonometrijskom i eksponencijalnom obliku moramo znati apsolutnu vrijednost (modul) i glavni argument kompleksnoga broja.

Na popisu MATLAB-ovih ugrađenih konstanti je i konstanta koja označava imaginarnu jedinicu. Ona se označava jednostavno s `i`. Napomenimo da se istim slovom vrlo često označavaju npr. redak matrice, varijabla funkcije kojoj je prirodna domena skup $\mathbb{N}$ i dr.

U ovom zadatku potrebne su nam četiri MATLAB-ove ugrađene funkcije koje kao argumente imaju kompleksan broj. To su (abecednim slijedom):

`abs` – funkcija koja određuje apsolutnu vrijednost (modul) kompleksnoga broja;
`conj` – funkcija koja određuje konjugat kompleksnoga broja;
`imag` – funkcija koja određuje imaginarni dio kompleksnoga broja;
`real` – funkcija koja određuje realni dio kompleksnoga broja.

Prijeđimo na rješavanje zadatka. U prvom retku koda zadajemo kompleksan broj iz zadatka. Već smo rekli da potencije s bazom e određujemo korištenjem ugrađene funkcije `exp`. U ovom zadatku vidimo da argument te ugrađene funkcije može biti i izraz koji sadrži imaginarnu jedinicu i, pa zadani kompleksan broj ne moramo pretvarati u trigonometrijski ili algebarski oblik.

- *Pitanje za razmišljanje*: Izvršite prvi redak koda. Utvrdite promjenu nastalu u radnom prostoru (*Workspace*). U kojemu je obliku zapisan kompleksan broj u MATLAB-ovoj memoriji?

Traženu vrijednost algebarskoga izraza određujemo u drugom retku koda. Radi jednostavnosti, tu vrijednost pohranit ćemo u varijabli nazvanoj `rezultat`. Za računanje drugoga korijena ponovno koristimo funkciju `realsqrt`. Pritom treba pripaziti na redoslijed algebarskih operacija – najprije trebaju biti izvedene algebarske operacije u okrugloj zagradi, potom množenje s konjugatom broja z^9, pa određivanje modula i tek na kraju množenje s $\sqrt{2}+1$.

- *Pitanja za razmišljanje*: Što će MATLAB računati upišemo li `real(z)^3`, a što ako upišemo `real(z^3)`? Koji je redoslijed algebarskih operacija u svakom pojedinom slučaju? Postoji li kompleksan broj z takav da su rezultati operacija `real(z)^3` i `real(z^3)` međusobno jednaki? Ako postoji, odredite sve takve kompleksne brojeve. Odgovorite na analogna pitanja za funkcije `imag` i `conj`.

z4.m

Ovaj zadatak možemo riješiti koristeći sve ranije navedene MATLAB-ove ugrađene funkcije, a možemo koristiti i MATLAB-ove ugrađene funkcije `sin` i `cos`. Prva od njih određuje sinus kompleksnoga(!) broja, a druga kosinus kompleksnoga broja. Ova tvrdnja nas može zbuniti jer smo u dosadašnjem obrazovanju te funkcije promatrali kao realne funkcije jedne realne varijable (tj. funkcije čiji su i argument i vrijednost realni brojevi), a ne kao kompleksne funkcije kompleksne varijable (tj. funkcije čiji su i argument i vrijednost kompleksni brojevi). Međutim, ne postoji nijedan opravdan razlog za ikakvo uznemiravanje. U slučaju kad su argumenti funkcija `sin` i `cos` realni brojevi, one se podudaraju s uobičajenim trigonometrijskim funkcijama *sin* i *cos*. Zbog toga nećemo posvećivati posebnu pozornost argumentima ovih funkcija.

Međutim, vrijedi istaknuti da, ako su argumenti tih funkcija realni brojevi, onda MATLAB pretpostavlja da je pripadna „mjerna jedinica“ radijan, a ne stupanj. Zbog toga treba biti vrlo oprezan: upišemo li `cos(90)` i pritisnemo Enter, MATLAB će ispisati `ans = 0.4481`, a ne 0 koju bismo očekivali. Razlog je, dakako, što je MATLAB računao kosinus argumenta iskazanoga u radijanima, a ne u stupnjevima. Međutim, upišemo li

```
sin(pi/2)
```

i pritisnemo Enter, MATLAB će ispisati

```
ans = 1,
```

što smo i očekivali. Potpuno analogna razmatranja vrijede i za ostale trigonometrijske funkcije.

Struktura samoga programskoga koda je vrlo jednostavna. U prvom retku koda zadajemo broj z_1 koristeći ugrađenu funkciju `realsqrt` i imaginarnu jedinicu i.

U drugom retku koda zadajemo broj z_2 potpuno analogno kao i broj z u prethodnom zadatku. Prisjetimo se da je $z = r \cdot \text{cis}\, \varphi = r \cdot e^{i \cdot \varphi}$, pa umjesto ugrađenih funkcija `sin` i `cos` koristimo jednu ugrađenu funkciju `exp`.

U trećem retku koda naveden je komentar. On počinje znakom % što MATLAB-u daje do znanja da se radi o komentaru i da ne treba izvršiti taj redak. U tom je retku naveden alternativan način zadavanja broja z_2 koristeći njegov trigonometrijski oblik. Nažalost, u MATLAB-u **ne postoji** ugrađena funkcija `cis` koja bi dozvoljavala unos kompleksnoga broja u trigonometrijskom obliku (ali ćemo takvu funkciju kasnije moći samostalno napisati!), pa moramo koristiti funkcije `cos` i `sin`. Mjera glavnoga argumenta kompleksnoga broja je radijan, pa nema bojazni da će MATLAB ispisati pogrešne vrijednosti funkcija `cos` i `sin`.

Četvrtom retku koda valja posvetiti malo više pozornosti. Broj z_3 je „na papiru" potpuno korektno zadan. Međutim, u MATLAB-u za njegovo zadavanje moramo koristiti (okrugle) zagrade. To činimo kad god su u pitanju razlomci koji u brojniku ili nazivniku imaju višečlane izraze. Pritom napomenimo da, ako u brojniku imamo isključivo operaciju množenja (ili „jaču" operaciju potenciranja), onda izraz u brojniku ne moramo stavljati u zagradu jer su množenje i dijeljenje međusobno ravnopravne operacije koje se izvode redoslijedom kojim su navedene u retku koda. U ovom zadatku u brojniku imamo zbroj zadanih kompleksnih brojeva, a u nazivniku razliku tih brojeva, pa te izraze obavezno valja staviti u okrugle zagrade.

U četvrtom retku koda određujemo traženi modul koristeći ugrađenu funkciju `abs` i pohranjujemo ga u varijabli `rezultat`.

- *Pitanje za razmišljanje*: Što će MATLAB izračunati u svakom od sljedećih izraza: `z1+z2/z1-z2`, `(z1+z2)/z1-z2` i `z1+z2/(z1-z2)`? Precizno objasnite svoje odgovore.

z5.m

U ovom zadatku upoznajemo implementiranje osnovnih trigonometrijskih funkcija u MATLAB-u. Dvije od njih već smo upoznali u rješenju prethodnoga zadatka. Dobra vijest je: iako na kalkulatorima ne postoji tipka pomoću koje možemo određivati vrijednosti funkcije kotangens (pa njegovu vrijednost tada moramo računati kao količnik kosinusa i sinusa), u MATLAB-u postoji ugrađena funkcija koja određuje kotangens realnoga broja. Preciznije, ugrađena funkcija `tan` određuje tangens realnoga broja, a ugrađena funkcija `cot` određuje kotangens realnoga broja.

Na ovom mjestu odmah treba istaknuti sljedeću „neprirodnu pojavu“. Znamo da izraz $\mathrm{tg}\left(\frac{\pi}{2}\right)$ nije definiran. Međutim, utipkamo li

```
tan(pi/2)
```

u novi redak MATLAB-ova komandnoga prozora i pritisnemo Enter, na naše će iznenađenje MATLAB ispisati

```
ans = 1.6331e+16.
```

Dakle, u MATLAB-u vrijednost $\mathrm{tg}\left(\frac{\pi}{2}\right)$ postoji i jednaka je (približno) $1.663 \cdot 10^{16}$. Pobjedonosno zaključujemo da smo otkrili „bug“ u MATLAB-u (jer opravdano nimalo ne sumnjamo u svoje znanje trigonometrije) i o tome ponosno namjeravamo obavijestiti autore MATLAB-a. Međutim, nema potrebe za tim. Naime, kad god računa s „nezgodnim“ konstantama poput π, MATLAB radi grešku čija veličina zavisi o preciznosti računala. Upravo u toj pogrešci krije se i stvaran razlog gornjega pogrešnoga rezultata: vrijednost $\frac{\pi}{2}$ izračunata je **približno** s određenom točnošću, pa MATLAB prilikom „operiranja“ s tom vrijednošću ne računa s točnom vrijednosti broja $\frac{\pi}{2}$, nego s njezinom približnom vrijednošću. Zbog toga je ispisani rezultat „konkretan“ realan broj, a ne konstanta `Inf`.

Rješavajući prethodni zadatak možda ste stekli dojam da je prilikom računanja s trigonometrijskim funkcijama nužno argument tih funkcija izraziti u radijanima i da će biti problema ako je argument tih funkcija iskazan u stupnjevima. Nasreću, ipak nije tako. MATLAB ima ukupno osam ugrađenih osnovnih trigonometrijskih funkcija od kojih polovica za argument ima broj iskazan u radijanima, a druga polovica argument iskazan u stupnjevima. Dakle, nećemo morati pretvarati stupnjeve u radijane, ali ćemo morati pripaziti koju od tih funkcija treba koristiti. Precizirajmo:

`sin` – računa vrijednost trigonometrijske funkcije sinus kad je argument iskazan u *radijanima*;

`cos` – računa vrijednost trigonometrijske funkcije kosinus kad je argument iskazan u *radijanima*;

`tan` – računa vrijednost trigonometrijske funkcije tangens kad je argument iskazan u *radijanima*;

`cot` – računa vrijednost trigonometrijske funkcije kotangens kad je argument iskazan u *radijanima*;

`sind` – računa vrijednost trigonometrijske funkcije sinus kad je argument iskazan u *stupnjevima*;

`cosd` – računa vrijednost trigonometrijske funkcije kosinus kad je argument iskazan u *stupnjevima*;

`tand` – računa vrijednost trigonometrijske funkcije tangens kad je argument iskazan u *stupnjevima*;

`cotd` – računa vrijednost trigonometrijske funkcije kotangens kad je argument iskazan u *stupnjevima*.

(Slovo `d` u nazivima funkcija `sind`, `cosd`, `tand` i `cotd` dolazi od engleske riječi *degrees* = stupanj.)

Vidjeli smo da je $\mathrm{tg}\left(\frac{\pi}{2}\right)$ neki „jako veliki“, ali ipak konkretan realan broj. Je li i $\mathrm{tg}\left(90^\circ\right)$ isti taj broj? Provjerimo. U novi redak MATLAB-ova komandnoga prozora upišimo

```
tand(90)
```

i pritisnimo Enter. MATLAB će ispisati:

```
ans = Inf
```

kao što smo i očekivali. Razlog je jasan: vrijednost 90 nije aproksimirana, nego točna vrijednost argumenta funkcije `tand`, pa je i rezultat ispravan. Potpuno analogna razmatranja vrijede i za ostale realne brojeve za koje trigonometrijske funkcije tangens i kotangens nisu definirane. Dakle, u takvim je slučajevima bolje argument iskazati u stupnjevima i koristiti ugrađene „*d*-trigonometrijske“ funkcije.

Vratimo se na kod koji razmatramo. Teorijski, za rješavanje postavljenoga zadatka treba nam točno jedan redak. Međutim, taj bi redak bio vrlo dugačak i nepregledan, a pri unosu sadržaja retka mogli bismo napraviti barem jednu pogrešku. Zbog toga ćemo se radi jednostavnosti i čitljivosti opredijeliti za kod s malo većim brojem redaka.

U prvi redak koda unosimo brojnik umanjenika. Argumenti funkcije sinus iskazani su u radijanima, pa koristimo ugrađenu funkciju `sin`. Vrijednost brojnika pohranjujemo u varijablu `b1` (slovo `b` u tom nazivu dolazi od riječi *brojnik*, a broj 1 naveden je jer se radi o brojniku prvoga (od dvaju) razlomaka).

U drugi redak koda unosimo nazivnik umanjenika. On se razlikuje od brojnika jedino u korištenoj trigonometrijskoj funkciji `cos`. Zbog toga možemo prekopirati prvi redak koda, pa izmijeniti `sin` u `cos`. Nazivnik umanjenika pohranjujemo u varijablu `n1` (slovo `n` u tom nazivu dolazi od riječi *nazivnik*, a broj 1 naveden je jer se radi o nazivniku prvoga (od dvaju) razlomaka).

U treći redak koda unosimo brojnik umanjitelja. Argumenti funkcije sinus iskazani su u radijanima, pa koristimo ugrađenu funkciju `cot`. Možemo prekopirati drugi redak koda, pa izmijeniti `cos` u `cot`. Vrijednost brojnika pohranjujemo u varijablu `b2` (slovo `b` u tom nazivu dolazi od riječi *brojnik*, a broj 2 naveden je jer se radi o brojniku drugoga (od dvaju) razlomaka).

U četvrti redak koda unosimo nazivnik umanjitelja. On se razlikuje od brojnika jedino u korištenoj trigonometrijskoj funkciji `tan`. Zbog toga možemo prekopirati treći redak koda, pa izmijeniti `cot` u `tan`. Nazivnik umanjitelja pohranjujemo u varijablu `n2` (slovo `n` u tom nazivu dolazi od riječi *nazivnik*, a broj 2 naveden je jer se radi o nazivniku drugoga (od dvaju) razlomaka).

Naposljetku, traženu vrijednost izračunamo u petom retku koda. Točna vrijednost izraza jednaka je nuli. (Uvjerite se u to riješivši zadatak analitički.) Međutim, MATLAB nije ispisao nulu, nego

```
rezultat = -2.2204e-16
```

Sad već znamo razlog ove pogreške: to je računanje s približnom vrijednosti broja π. Vidimo da je red veličine učinjene pogreške 10^{-16}.

- *Pitanje za razmišljanje*: Preradite dobiveni programski kod tako da vrijednosti argumenata trigonometrijskih funkcija izrazite u stupnjevima, pa riješite zadatak koristeći odgovarajuće ugrađene MATLAB-ove funkcije. Što se u ovom slučaju dobiva kao konačno rješenje zadatka? Je li taj rezultat očekivan?

z6.m

U ovom zadatku upoznajemo implementaciju osnovnih ciklometrijskih funkcija u MATLAB-u. U prethodnom zadatku smo vidjeli da vrijednost svake trigonometrijske funkcije možemo računati kad je njezin argument iskazan u radijanima, odnosno stupnjevima. Zbog toga je u MATLAB-u implementirano ukupno osam arkus funkcija. Naziv svake od njih dobijemo tako da ispred naziva pripadne trigonometrijske funkcije dopišemo slovo `a`. Npr. arkus sinus je implementiran koristeći ugrađene funkcije `asin` i `asind` itd. Pregledno navedimo svih osam ugrađenih ciklometrijskih funkcija:

`asin` – računa vrijednost funkcije arcsin, pri čemu je ta vrijednost iskazana u *radijanima*;
`acos` – računa vrijednost funkcije arccos, pri čemu je ta vrijednost iskazana u *radijanima*;
`atan` – računa vrijednost funkcije arctg, pri čemu je ta vrijednost iskazana u *radijanima*;
`acot` – računa vrijednost funkcije arcctg, pri čemu je ta vrijednost iskazana u *radijanima*;
`asind` – računa vrijednost funkcije arcsin, pri čemu je ta vrijednost iskazana u *stupnjevima*;
`acosd` – računa vrijednost funkcije arccos, pri čemu je ta vrijednost iskazana u *stupnjevima*;
`atand` – računa vrijednost funkcije arctg, pri čemu je ta vrijednost iskazana u *stupnjevima*;
`acotd` – računa vrijednost funkcije arcctg, pri čemu je ta vrijednost iskazana u *stupnjevima*.

Znamo da je npr. $\arcsin\left(\frac{1}{2}\right)=\frac{\pi}{6}$. Doista, utipkamo li u novi redak komandnoga prozora

```
asin(1/2)
```

i pritisnemo Enter, MATLAB će ispisati:

```
ans = 0.5236
```

što je (približna) vrijednost broja $\frac{\pi}{6}$ zaokružena s točnošću od 10^{-4}. Utipkamo li u novi redak komandnoga prozora

```
asind(1/2)
```

i pritisnemo Enter, MATLAB će ispisati:

```
ans = 30.0000
```

To smo mogli i očekivati jer je $\frac{\pi}{6}\text{ rad} = 30^\circ$. Sad se otvara pitanje: koje ugrađene funkcije primijeniti kad – analogno kao u ovom zadatku – imamo algebarski izraz u kojemu se pojavljuju ciklometrijske funkcije? One koje vraćaju vrijednost u radijanima ili one koje vraćaju vrijednost u stupnjevima?

Odgovor na ovo pitanje je vrlo jednostavan. „*d*-ciklometrijske“ funkcije, odnosno ugrađene funkcije `asind`, `acosd`, `atand` i `acotd` primjenjuju se u slučajevima kad je taj rezultat *mjera nekoga kuta u geometrijskom objektu* (npr. u trokutu, četverokutu i dr.). Takvu mjeru u pravilu iskazujemo u stupnjevima, pa zbog toga i primjenjujemo navedene funkcije. Međutim, ako nam nije zadana interpretacija rezultata, onda primjenjujemo standardne ugrađene ciklometrijske funkcije `asind`, `acosd`, `atand` i `acotd`.

Vratimo se na zadatak koji rješavamo. Nemamo nikakvu informaciju o značenju vrijednosti koje predstavljaju pojedine članove zadanoga numeričkoga izraza, a u njemu se pojavljuje i član $-\frac{4}{3}\cdot\pi$. Zbog toga ćemo primijeniti standardne ugrađene ciklometrijske funkcije. Dodatno ćemo koristiti i ugrađenu funkciju `realsqrt`. Izraz čiju vrijednost tražimo nije predugačak, pa ga možemo zapisati u točno jednom retku koda. Njegovu vrijednost pohranjujemo u varijablu `rezultat`.

- *Pitanje za razmišljanje*: Preradite dobiveni programski kod tako da umjesto standardnih ciklometrijskih funkcija koristite odgovarajuće „*d*-ciklometrijske“ funkcije. Što se dobiva kao konačan rezultat? Biste li dobili rezultat dobiven izvršenjem gornjega programskoga koda ako biste $-\frac{4}{3}\cdot\pi$ zamijenili s -240? Precizno objasnite svoje odgovore.

z7.m

U ovom zadatku upoznajemo implementaciju osnovnih hiperbolnih i area funkcija u MATLAB-u. Znamo da postoje četiri osnovne hiperbolne funkcije: sh, ch, th i cth. Nazive pripadnih ugrađenih funkcija u MATLAB-u dobijemo tako da iza naziva analogne *trigonometrijske* funkcije čiji je argument iskazan u radijanima dopišemo slovo `h`. (To slovo potječe od prvoga slova riječi „hiperbolni“). Npr. sinus hiperbolni je implementiran koristeći ugrađenu funkciju `sinh` itd. Potpuno analogno kao i u slučaju ciklometrijskih funkcija, inverzi hiperbolnih funkcija, odnosno area funkcije, implementirani su ugrađenim funkcijama čije nazive dobijemo tako da ispred naziva pripadne hiperbolne funkcije nadopišemo `a`. Npr. ugrađena funkcija koja računa area sinus hiperbolni nekoga broja je `asinh` itd.

Pregledno navedimo sve hiperbolne i area funkcije implementirane u MATLAB-u:

`sinh` – računa vrijednost funkcije sh;
`cosh` – računa vrijednost funkcije ch;
`tanh` – računa vrijednost funkcije th;
`coth` – računa vrijednost funkcije cth;
`asinh` – računa vrijednost funkcije Arsh;
`acosh` – računa vrijednost funkcije Arch;
`atanh` – računa vrijednost funkcije Arth;
`acoth` – računa vrijednost funkcije Arcth.

Podsjetimo da argumenti hiperbolnih funkcija nisu (i ne mogu biti) izraženi u stupnjevima jer te funkcije za svoju osnovu nemaju namatanje pravca na kružnicu. Zbog toga ne postoje odgovarajući analogoni „*d*-trigonometrijskih“ i „*d*-ciklometrijskih“ funkcija.

U ovom ćemo zadatku upoznati i još dvije nove ugrađene MATLAB-ove funkcije. To su funkcije `ceil` i `floor`. Prije njihova opisa, definirajmo dvije matematičke funkcije „najmanje cijelo“ (ili „strop“) i „najveće cijelo“ (ili „pod“) redom s:

$$\lceil x \rceil := \text{najmanji cijeli broj jednak ili veći od } x;$$
$$\lfloor x \rfloor := \text{najveći cijeli broj jednak ili manji od } x.$$

Prirodna domena *obiju* funkcija je skup $\mathbb{R}$, a njihova slika je skup $\mathbb{Z}$. Primijetite da vrijede jednakosti

$$\lfloor x \rfloor = \lceil x \rceil = x, \ \forall x \in \mathbb{Z}.$$

Dakle, obje navedene funkcije se podudaraju s identitetom ako i samo ako je njihov argument cijeli broj. (Kratko i neprecizno kažemo da „pod i strop ostavljaju cijele brojeve na miru".)

Ove funkcije prirodno možemo interpretirati i kao zaokruživanje naviše, odnosno naniže. Međutim, neprekidno moramo imati na umu da je vrijednost svake od njih cijeli broj, pa te funkcije ne možemo koristiti ako npr. broj 12.345 želimo zaokružiti naviše na 12.35. Treba biti posebno oprezan ako je vrijednost varijable strogo negativan broj jer su npr.

$$\lceil -0.99 \rceil = 0,$$
$$\lfloor -0.99 \rfloor = -1.$$

Sad možemo navesti ugrađene MATLAB-ove funkcije koje za svaki realan broj računaju „najmanje cijelo" i „najveće cijelo":

`ceil` – računa vrijednost funkcije „najmanje cijelo";
`floor` – računa vrijednost funkcije „najveće cijelo".

- *Pitanje za razmišljanje*: U MATLAB-u argument obiju funkcija `ceil` i `floor` može biti i kompleksan broj. (Kasnije ćemo vidjeti da njihov argument može biti i matrica.) Definirajte $z = \sqrt{3} - \sqrt{2} \cdot i$, pa odredite `ceil(z)` i `floor(z)`. Što primjećujete? Objasnite „djelovanje" funkcija `ceil` i `floor` na argument koji je kompleksan broj.

Vratimo se na zadatak i promotrimo pripadni programski kod. Teorijski, zadatak bismo ponovno mogli riješiti u točno jednom retku koda. Međutim, radi jednostavnosti, preglednosti i razumljivosti, koristit ćemo malo više redaka.

U prvom retku računamo argument funkcije „najveće cijelo". Taj je argument namjerno označen s `i1`. (`i` je početno slovo riječi „izraz", a broj 1 označava da se radi o prvom (od dvaju) numeričkih izraza.) Ovdje ćemo vidjeti da, iako se u MATLAB-u s `i` označava imaginarna jedinica, neće biti problema s ovako uvedenom oznakom. Argument funkcije „najveće cijelo" zadajemo koristeći ugrađene hiperbolne funkcije.

U drugom retku računamo argument funkcije „najmanje cijelo". Taj je argument namjerno označen s `i2`. (`i` je početno slovo riječi „izraz", a broj 2 označava da se radi o drugom (od dvaju) numeričkih izraza.) Argument funkcije „najmanje cijelo" zadajemo koristeći ugrađene area funkcije.

U trećem retku računamo vrijednost zadanoga izraza koristeći varijable `i1` i `i2` iz prvih dvaju redaka, te ugrađene funkcije `realsqrt`, `floor`, `cosh` i `ceil`.

z8.m

U ovom ćemo zadatku pokazati kako u MATLAB-u možemo provjeriti valjanost neke logičke tvrdnje, a da pritom ne moramo ispisivati vrijednosti numeričkih izraza koji se pojavljuju u toj tvrdnji. Vidjet ćemo i kako u komandnom prozoru možemo ispisati i odgovarajući tekst.

Analizirajmo najprije zadatak napamet. Znamo da je $\operatorname{tg}\left(\frac{\pi}{4}\right)=1$. Prema tome, očekujemo da obje zadane jednakosti budu istinite. Vidjet ćemo hoće li MATLAB ispuniti naša očekivanja.

Prva „zamka“ u ovom zadatku je zadavanje jednakosti. Najčešća pogreška je napisati `a=1`. Što zapravo znači taj izraz? Taj izraz znači da smo deklarirali varijablu `a` i dodijelili joj vrijednost 1. Tako u pravilu postupamo kad s tom varijablom dalje želimo nešto računati. Međutim, kod provjere valjanosti neke logičke tvrdnje koja sadrži znak = moramo postupiti malo drugačije. U tom slučaju treba koristiti znak dvostruke jednakosti, tj. znak ==. Konkretno, napišemo li npr.

```
3+2==5
```

kako će MATLAB shvatiti taj unos? On će najprije izračunati vrijednost na lijevoj strani jednakosti. Potom će utvrditi jesu li lijeva i desna strana jednakosti identički jednake. Ako jesu, ispisat će 1. Ako nisu, ispisat će 0. Dakle, rezultat izvršavanja ovoga retka bit će ili 1 ili 0, ovisno o tome je li jednakost istinita ili nije.

- *Pitanje za razmišljanje*: Je li u MATLAB-u smislen izraz `cos(3+2==6)`? Ako jest, što je konačna vrijednost toga izraza? Odgovorite na pitanja bez korištenja MATLAB-a, pa potom provjerite svoje rješenje koristeći MATLAB. Sve svoje tvrdnje precizno obrazložite.

Prilikom provjere istinitosti neke logičke tvrdnje poželjno je ispisati i odgovarajući tekst, tj. ispisati je li tvrdnja istinita ili nije. Naravno da taj zaključak možemo izvesti i iz ispisanoga rezultata (0 ili 1), ali u praksi obično pretpostavljamo da korisnik nije detaljno upoznat s osnovama matematičke logike i da će mu rezultat biti prilično jasniji ako MATLAB ispiše odgovarajuću rečenicu. Za ispis teksta u komandni prozor koristimo ugrađenu funkciju `disp`. (Njezin naziv tvore prva četiri slova engleske riječi *display* = prikaz.) Ta funkcija je vrlo jednostavna: njezin argument je tekst nužno zapisan pod znakovima jednostrukih navodnika. Zaboravite li napisati jednostruke navodnike, MATLAB će javiti pogrešku.

Naposljetku spomenimo i logičku strukturu `if` … `else` … (engl.: „ako … inače …“) koju ćemo ovdje koristiti. U starijim programskim jezicima ta se struktura koristila u obliku `if` … `then` … `else` … (engl: „ako … onda … inače …) ili samo u obliku `if` … `then` … . Ta je struktura vrlo jednostavna. Iza riječi `if` nužno se navodi logički uvjet. Ako je taj uvjet ispunjen, onda se izvršava dio programskoga koda sve do riječi `else` (ako je ta riječ dio koda) ili do riječi `end` koja označava kraj strukture. Ako taj uvjet nije ispunjen, onda se izvršava dio programskoga koda od riječi `else`, pa do riječi `end` (tj. opet do kraja strukture). Kraj strukture **obavezno** mora biti naznačen ugrađenom funkcijom `end`. Ako prilikom „slaganja“ strukture zaboravite riječ `end`, MATLAB će javiti pogrešku.

- *Pitanje za razmišljanje*: Može li nastupiti slučaj da *nijedan* redak `if` … `else` … strukture *nijednom* ne bude izvršen? Ako može, uz koje će se uvjete to dogoditi? Precizno objasnite svoje odgovore.

Pogledajmo sada strukturu promatranoga koda. U njegovu prvomu retku određujemo vrijednost $\operatorname{tg}\left(\frac{\pi}{4}\right)$ i pohranjujemo je u varijablu `a`. Točka-zarez na kraju prvoga retka označava da dobivena vrijednost neće biti ispisana u komandni prozor kad jednom pokrenemo taj kod.

- *Pitanje za razmišljanje*: Vrijednost varijable `a` neće biti ispisana u komandnom prozoru, ali će nužno biti prikazana negdje drugdje (i na taj prikaz ne možemo utjecati). Gdje u okviru komandnoga prozora možete vidjeti vrijednost pohranjenu u varijabli `a` nakon izvršenja prvoga retka koda?

Od drugoga do šestoga retka koda smještena je prva `if` … `else` … struktura. S obzirom na postavljeni zadatak, ona je vrlo jednostavna. U drugom retku MATLAB provjerava valjanost jednakosti $a=1$. Ako je ta jednakost istinita, izvršit će se treći (u odnosu na drugi, malo uvučeni) redak koda, odnosno ispisat će se tekst da je ta jednakost istinita. Međutim, ako je ta jednakost lažna, onda MATLAB prelazi u četvrti redak koda (gdje je upisana funkcija `else`) i izvršava peti (i šesti) redak koda, tj. ispisuje tekst da je analizirana jednakost lažna i završava strukturu.

Potpuno analogan postupak provodimo i za drugu zadanu jednakost, i to u dijelu od sedmoga do jedanaestoga retka koda. Prilikom unosa tih redaka uobičajenom „kopiraj – zalijepi“ procedurom možete kopirati dio od drugoga do šestoga retka, pa potom u kopiranom dijelu izvršiti nekoliko potrebnih izmjena.

- *Pitanja za razmišljanje*: Što će MATLAB ispisati nakon izvršenja cijeloga koda? Jesu li dobiveni ispisi očekivani? Ako neki od njih nije, objasnite zašto je rezultat drugačiji od očekivanoga. Nadalje, zamijenite $\frac{\pi}{4}$ s 45°, a funkciju `tan` s `tand`, pa ponovno izvršite cijeli kod. Što se dogodilo? Jesu li sada dobiveni ispisi očekivani? Precizno objasnite svoje odgovore.

z9.m

U ovom zadatku upoznajemo još jednu novu MATLAB-ovu ugrađenu funkciju `round` namijenjenu zaokruživanju brojeva. Ta funkcija u pravilu ima dva argumenta: prvi je broj kojega želimo zaokružiti, a drugi je broj decimala na koje želimo zaokružiti prvi argument. Zaokruživanje se pritom provodi standardno, bez „rezanja" suviška decimala. Istaknimo da drugi argument (broj decimala) možemo i izostaviti ako prvi argument želimo zaokružiti na najbliži *cijeli* broj. To izostavljanje je, dakako, ekvivalentno unosu broja 0 kao drugoga argumenta funkcije `round`.

Primijetimo da, analogno funkcijama `ceil` i `floor`, i funkcija `round` „ostavlja na miru" sve cijele brojeve, tj. relacija `round(x)==x` je istinita ako i samo ako je x cijeli broj.

Promotrimo pripadni programski kod. Iako ga možemo strukturirati tako da sadrži točno jedan redak, ponovno radi jednostavnosti, preglednosti i razumljivosti koristimo malo veći broj redaka.

U prvom retku računamo prvi član zadanoga izraza i pohranjujemo dobiveni rezultat u varijablu `i1`. Pritom treba pripaziti na jednakost $\log_2^4\left(\frac{1}{3}\right)=\left(\log_2\left(\frac{1}{3}\right)\right)^4$. Dakle, u eksponentu najprije moramo odrediti binarni logaritam broja $\frac{1}{3}$, potom potencirati dobiveni rezultat na četvrtu potenciju, pa naposljetku potencirati broj 3 dobivenim rezultatom.

U drugom retku računamo drugi član zadanoga izraza i pohranjujemo dobiveni rezultat u varijablu `i2`. Pritom treba pripaziti na jednakost $\log^3 5=\left(\log_{10} 5\right)^3$. Dakle, u eksponentu najprije moramo odrediti dekadski logaritam broja 5, potom potencirati dobiveni rezultat na treću potenciju, pa naposljetku potencirati broj 2 dobivenim rezultatom.

U trećem retku računamo treći član zadanoga izraza i pohranjujemo dobiveni rezultat u varijablu `i3`. Pritom treba pripaziti na jednakost $\sin^2 15^\circ=\left(\sin 15^\circ\right)^2$. Dakle, u eksponentu najprije moramo odrediti sinus kuta čija je mjera 15° (koristeći funkciju `sind`), potom kvadrirati dobiveni rezultat, pa naposljetku potencirati broj e dobivenim rezultatom (koristeći funkciju `exp`).

U posljednjem retku koda određujemo traženu vrijednost koristeći spomenutu funkciju `round`. Njezin drugi argument izostavljamo jer zadatak zahtijeva određivanje cijeloga broja najbližega vrijednosti prvoga argumenta te funkcije.

- *Pitanja za razmišljanje*: Uz koji uvjet na vrijednost pripadnoga argumenta funkcije `round` i `ceil` komutiraju, tj. vrijedi jednakost `round(ceil(x)) =` `= ceil(round(x))`? Obrazložite svoj odgovor. Odgovorite (i obrazložite svoj odgovor) i na analogno pitanje dobiveno ako funkciju `ceil` zamijenimo funkcijom `floor`.

- *Pitanja za razmišljanje*: Kako biste – koristeći funkciju `round` – odredili *samo* prvu, *samo* drugu i *samo* treću decimalu broja π bez ispisa toga broja u komandni prozor? Kako biste odredili *prve dvije* decimale, *prve tri* decimale i *prve četiri* decimale broja e bez ispisa toga broja u komandni prozor? Provjerite svoje odgovore koristeći MATLAB.