Zadaća 1.

Ova zadaća nosi ukupno 4 poena, od kojih svaki nosi po 1 poen. Svi zadaci se mogu uraditi na osnovu gradiva sa prva tri predavanja i pretpostavljenog predznanja iz predmeta "Osnove računarstva". Rok za predaju ove zadaće je nedjelja, 3. IV 2022. do kraja dana.

VEOMA VAŽNO: U svim zadacima, za indeksaciju vektora, dekova, modernih nizova i stringova, koristite funkciju "`at`" umjesto operatora "`[]`" (tj. umjesto "`a[i]`" odnosno "`a[i][j]`" pišite "`a.at(i)`" odnosno "`a.at(i).at(j)`". To je jedini način da se sa sigurnošću utvrdi da li pristupate elementima izvan dozvoljenog opsega. *Nepoštovanje ove odredbe može biti kažnjeno nepriznavanjem čitavog* (*inače tačnog*) *zadatka*!

1. Neka je data neka sekvenca brojeva $a_1, a_2, a_3, \ldots, a_n$. Za sekvencu $a_{k+1}, a_{k+2}, \ldots, a_n, a_1, a_2, \ldots, a_k$ kažemo da je *ciklička permutacija* prve sekvence s pomakom k. Na primjer, za sekvencu brojeva 3, 5, 1, 2, 4, 3, 5, 7, 8, 1, 2, 5, 9, 4, 7 sekvenca 5, 7, 8, 1, 2, 5, 9, 4, 7, 3, 5, 1, 2, 4, 3 predstavnja njenu cikličku permutaciju s pomakom 6. Pri tome, iz definicije slijedi da se svaka sekvenca može smatrati kao svoja vlastita ciklička permutacija s pomakom 0.

   Vaš zadatak je da napravite funkciju nazvana "`CiklickaPermutacija`", koja prima dva vektora cijelih brojeva kao parametre. Ti vektori predstavljaju dvije sekvence cijelih brojeva, a funkcija treba da utvrdi da li druga sekvenca predstavlja cikličku permutaciju prve sekvence. Ukoliko jeste, funkcija treba da vrati odgovarajući pomak kao rezultat, a u suprotnom, funkcija treba da vrati -1 kao rezultat (što signalizira da druga sekvenca nije ciklička permutacija prve).

   Obavezno napišite i mali testni program ("`main`" funkciju) u kojoj ćete testirati napisanu funkciju na dvije sekvence brojeva koje se unose sa tastature. Brojevi se unose dok se ne unese bilo šta što nije broj (npr. tačka ili riječ "kraj"). Nakon unosa dvije sekvence, program treba da utvrdi da li druga od njih predstavlja cikličku permutaciju prve, i da ispiše na ekran rezultate analize. Dijalog između korisnika i programa treba da izgleda kao na sljedećim slikama:

```
Prva sekvenca: 3 5 1 2 4 3 5 7 8 1 2 5 9 4 7 .
Druga sekvenca: 5 7 8 1 2 5 9 4 7 3 5 1 2 4 3 .

Druga sekvenca je ciklicka permutacija prve s pomakom 6.
```

```
Prva sekvenca: 1 2 3 4 5 kraj
Druga sekvenca: 2 3 4 5 6 7 8 kraj

Druga sekvenca nije ciklicka permutacija prve.
```

2. U mnogim oblastima nauke i tehnike, koristi se modeliranje realnih fizičkih objekata putem matrica. Takvo modeliranje je vrlo uobičajeno u elektrotehnici (matrice su osnovni model koji se koristi za potrebe opisivanje raznih električnih kola). Na primjer, za potrebe modeliranja nekog elektroenergetskog sistema i proračun pojedinih parametara sistema, uvijek se koriste matrice. Recimo, rezultati mjerenja unutar nekog elektroenergetskog sistema (npr. napon u čvorišnim tačkama sistema), tipično se zapisuju u kvadratne matrice. Ovisno od kompleksnosti sistema, matrice mogu biti velikog reda (u opsegu od dimenzija matrice 1000×1000 za manje sisteme, a za veće sisteme su matrice reda 10000×10000, pa i više). Matematičke operacije nad matricama velikih dimenzija, kao što su množenje ili traženje inverzne matrice, te samo memoriranje matrice u memoriji računara, zahtjeva mnogo vremena i resursa, zbog čega su razvijene različite metode za optimizaciju rada.

   Jedna od metoda za optimizaciju rada sa matricama je tzv. *LU faktorizacija matrica*, u kojoj se neka kvadratna matrica $\boldsymbol{H}$ razlaže na proizvod dvije kvadratne matrice $\boldsymbol{L}$ i $\boldsymbol{U}$ (dakle, vrijedi relacija $\boldsymbol{H} = \boldsymbol{L} \cdot \boldsymbol{U}$) od kojih je matrica $\boldsymbol{L}$ donja trougaona matrica (tj. svi elementi iznad glavne dijagonale su joj jednaki nuli), dok je matrica $\boldsymbol{U}$ gornja trougaona matrica. Ovo razlaganje nije jedinstveno (tj, matrice $\boldsymbol{L}$ i $\boldsymbol{U}$ nisu jednoznačno određene). Za praktične potrebe najčešće se koriste tzv. *Doolittleovo razlaganje* i *Croutovo razlaganje*. Razlika između ova dva razlaganja je što se u Doolittleovom razlaganju u matrici $\boldsymbol{L}$ nalaze sve jedinice na glavnoj dijagonali, dok se u Croutovom razlaganju u matrici $\boldsymbol{U}$ nalaze sve jedinice na glavnoj dijagonali.

LU razlaganje najčešće se obavlja pomoću *Croutovog algoritma* (kojim se može vršiti razlaganje bilo Doolittleovog, bilo Croutovog tipa). U nastavku ćemo se fokusirati samo na Doolittleovo razlaganje. Dakle, prema ovom razlaganju, vrši se faktorizacija oblika

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2n} \\ h_{31} & h_{32} & h_{33} & \dots & h_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{n1} & h_{n2} & h_{n3} & \dots & h_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{21} & 1 & 0 & \dots & 0 \\ l_{31} & l_{32} & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ 0 & u_{22} & u_{23} & \dots & u_{2n} \\ 0 & 0 & u_{33} & \dots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & u_{nn} \end{bmatrix}$$

Prema Croutovom algoritmu, elementi matrica $\boldsymbol{L}$ i $\boldsymbol{U}$ mogu se izračunati prema sljedećim relacijama:

$$l_{ij} = \frac{1}{u_{jj}} \left(h_{ij} - \sum_{k=1}^{j-1} l_{ik} \cdot u_{kj} \right) \qquad \text{za } j < i$$

$$u_{ij} = h_{ij} - \sum_{m=1}^{i-1} l_{im} \cdot u_{mj} \qquad \text{za } j \geq i$$

Ukoliko za primjer uzmemo sljedeću matricu dimenzija 3×3 kao matricu $\boldsymbol{H}$:

$$\boldsymbol{H} = \begin{bmatrix} 8 & 1 & -2 \\ 2 & -7 & 1 \\ 1 & 2 & 9 \end{bmatrix}$$

tada prethodno opisani postupak LU faktorizacije daje sljedeće matrice $\boldsymbol{L}$ i $\boldsymbol{U}$:

$$\boldsymbol{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0.25 & 1 & 0 \\ 0.125 & -0.258 & 1 \end{bmatrix} \quad \boldsymbol{U} = \begin{bmatrix} 8 & 1 & -2 \\ 0 & -7.25 & 1.5 \\ 0 & 0 & 9.637 \end{bmatrix}$$

Nije teško provjeriti da zaista vrijedi da je $\boldsymbol{L} \cdot \boldsymbol{U} = \boldsymbol{H}$.

Vaš zadatak je da napišete tri funkcije nazvane redom "`MnozenjeMatrica`", "`DoolittleLU`" i "`ProvjeriFaktorizaciju`". Funkcija "`MnozenjeMatrica`" prima dvije matrice, predstavljene kao vektori vektôrā realnih brojeva, a kao rezultat vraća matricu (predstavljenu također kao vektor vektôrā realnih brojeva) koja je rezultat množenja matrica prenesenih putem parametara u funkciji. Ukoliko unesene matrice nisu saglasne za množenje, neophodno je baciti izuzetak tipa "`domain_error`" uz prateći tekst "Matrice nisu saglasne za mnozenje!". Također, ukoliko parametri nemaju ispravnu strukturu matrice, tj. ukoliko svi redovi nemaju isti broj elemenata, treba baciti izuzetak istog tipa uz prateći tekst "Neispravna matrica!".

Funkcija "`DoolittleLU`" kao ulazni parametar prima matricu $\boldsymbol{H}$ iz prethodno opisanog postupka LU faktorizacije, predstavljenu kao vektor vektôrā realnih brojeva, te vrši kreiranje $\boldsymbol{L}$ i $\boldsymbol{U}$ matrica. Ukoliko matrica proslijeđena funkciji "`DoolittleLU`" nije kvadratnog oblika, potrebno je baciti izuzetak tipa "`domain_error`" uz prateći tekst "Matrica mora biti kvadratna!". Također, ukoliko parametar nema ispravnu strukturu matrice, treba baciti izuzetak istog tipa uz prateći tekst "Neispravna matrica!". Uz pretpostavku da je ulazna matrica $\boldsymbol{H}$ formata $n \times n$, ova funkcija vraća kao rezultat trodimenzionalnu matricu formata $2 \times n \times n$ organiziranu kao moderni niz (tj, objekat tipa "`array`") od samo dva elementa (s indeksima 0 i 1), a čiji su elementi vektori vektôrā realnih brojeva (formata $n \times n$). Prvi element vraćenog rezultata (s indeksom 0) predstavlja matricu $\boldsymbol{L}$, a drugi element (s indeksom 1) matricu $\boldsymbol{U}$. Inače, vraćenu trodimenzionalnu matricu možemo vizualizirati kao kvadar koji ima dva "sprata". Prvi indeks predstavlja redni broj sprata, dok drugi i treći indeks predstavljaju redni broj reda odnosno kolone, kao što je uobičajeno (pri čemu svi indeksi kreću od nule nadalje).

Konačno, funkcija "`ProvjeriFaktorizaciju`" provjerava da li je faktorizacija korektna. Kao prvi parametar, ova funkcija prima matricu $\boldsymbol{H}$, a kao drugi parametar trodimenzionalnu matricu koja je onakva kakvu vraća funkcija "`DoolittleLU`" (dva sprata, koji bi redom trebali da predstavljaju matrice $\boldsymbol{L}$ i $\boldsymbol{U}$. Ukoliko su oba parametra ispravnog formata, i ukoliko je pri tome $\boldsymbol{L} \cdot \boldsymbol{U} = \boldsymbol{H}$, ova

funkcija vraća logičku vrijednost "tačno" kao rezultat. U suprotnom, ona vraća kao rezultat logičku vrijednost "netačno". Vodite računa da ova funkcija ne smije da baci izuzetak čak ni u slučaju da neka od funkcija koju eventualno pozivate iz ove funkcije baci izuzetak. Zbog netačnosti računarske aritmetike, umjesto egzaktnog testiranja na jednakost $x = y$, koristite aproksimativni test oblika $|x - y| < \varepsilon$, gdje je ε dozvoljena tolerancija za koju ćete uzeti $\varepsilon = 10^{-10}$.

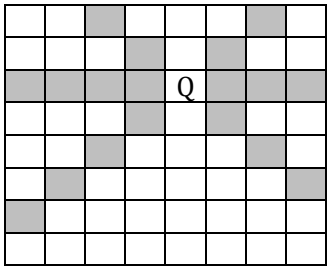
Treba napomenuti da LU faktorizacija ne postoji ukoliko je matrica $\boldsymbol{H}$ singularna, tj. ukoliko za nju vrijedi da je $\det \boldsymbol{H} = 0$ (ona tada nema ni inverznu matricu). Naime, u slučaju da se to dogodi, prilikom računanja elemenata l_{ij}, u nekom trenutku će se dogoditi da imamo $u_{jj} = 0$, tako da ćemo imati dijeljenje nulom. Stoga je prilikom računanja l_{jj} potrebno prethodno provjeriti da li je $u_{jj} = 0$, i ukoliko jeste, funkcija "`DoolittleLU`" treba baciti izuzetak tipa "`domain_error`" uz prateći tekst "Matrica je singularna!". Zbog netačnosti aritmetike s realnim brojevima, umjesto uvjeta $u_{jj} = 0$ testirajte uvjet $|u_{jj}| < \varepsilon$, gdje je ε dozvoljena tolerancija za koju ćete uzeti $\varepsilon = 10^{-10}$.

Obavezno napišite i mali testni program ("`main`" funkciju) u kojoj ćete testirati napisanu funkciju `DoolitleLU`" na skupini brojeva koji se unose sa tastature. Predvidite i hvatanje izuzetaka na mjestima gdje može doći do problema. Dijalozi između programa i korisnika trebaju izgledati kao na sljedećim slikama:

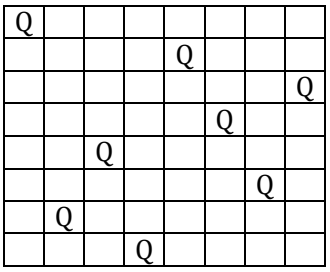
```
Broj redova/kolona matrice: 3
Elementi matrice:
8 1 -2
2 -7 1
1 2 9

Matrica L:
1 0 0
0.25 1 0
0.125 -0.258 1

Matrica U:
8 1 -2
0 -7.25 1.5
0 0 9.637
```

```
Broj redova/kolona matrice: 3
Elementi matrice:
1 2 3
4 5 6
7 8 9

LU faktorizacija ne postoji, jer je matrica singularna!
```

3. U igri šah, dama (ili kraljica) je figura koja ima sposobnost da napada sva polja koja se nalaze u istom redu, ili koloni, ili jednoj od dvije dijagonale u kojoj se i sama nalazi. Na primjer, na sljedećoj slici, osjenčeno su prikazana polja koje napada dama (kraljica) označena s "Q" (od engl. Quinn):



U rekreativnoj matematici, zanimljiv je tzv. *problem osam dama*, koji se sastoji u tome da se na šahovsku damu rasporedi 8 dama tako da ni jedna od njih ne napada ni jednu drugu (inače, ovaj problem se može generalizirati na problem n dama, gdje na poopćenu šahovsku tablu formata $n \times n$ treba rasporediti n dama koje se međusobno ne napadaju). Sam problem nije jednostavan za rješavanje (tako da se ne trebate brinuti, *nećete praviti program za nalaženje rješenja ovog problema*). Konkretno, problem 8 dama ima ravno 92 različita rješenja, od kojih je jedno prikazano na sljedećoj slici:



Vaš zadatak je da napravite funkciju "DaLiSeDameNapadaju" koja testira da li se dame raspoređene na generaliziranu šahovsku ploču međusobno napadaju ili ne. Funkcija prima jedan parametar koji predstavlja opis šahovske ploče. Taj parametar je po tipu matrica organizirana kao vektor vektôrā čiji su elementi logičke vrijednosti (tj. tipa "**bool**"). Element "netačno" označava prazno polje, a element "tačno" označava polje na kojem se nalazi dama. Pri tome, šahovska ploča je generalizirana u smislu da ne samo da ne mora biti formata 8×8, nego ne mora čak imati ni isti broj redova i kolona. Štaviše, pojedini redovi ne moraju ni imati isti broj elemenata (tj. šahovska ploča može biti "grbava"). Kao rezultat funkcija vraća "tačno" ili "netačno", ovisno od toga da li se dame raspoređene na šahovsku ploču napadaju ili ne.

Napisanu funkciju treba testirati u testnom programu ("main" funkciji) koja omogućava da sa tastature unesete opis šahovske table, a koji će ispisati da li se dame raspoređene na tabli napadaju ili ne. Opis table se zadaje red po red (nakon unosa svakog od redova treba pritisnuti ENTER), a unos se završava nakon što se ENTER pritisne naprazno (bez prethodno unesenog ijednog znaka). Damu označava znak Q, a bilo koji drugi znak označava prazno polje. Primjeri dijaloga između korisnika i programa prikazan je na sljedećoj slici:

```
Unestite opis sahovske ploce
(Q za kraljicu, bilo sta drugo za prazno polje, ENTER naprazno za kraj unosa):
Q-------
----Q---
-------Q
-----Q--
--Q-----
------Q-
-Q------
---Q----

Dame se medjusobno ne napadaju.
```

```
Unestite opis sahovske ploce
(Q za kraljicu, bilo sta drugo za prazno polje, ENTER naprazno za kraj unosa):
----Q--
Q-
--Q-
----
---Q--

Dame se medjusobno napadaju.
```

U posljednjem primjeru, dama u prvom redu na poziciji $(1, 5)$ i dama u trećem redu na poziciji $(3, 3)$ međusobno se napadaju (jer se nalaze u istoj dijagonali).

4. Potrebno je napisati funkciju "RazvrstajRijeciPoDuzini" koja izdvaja riječi iz rečenice koja joj se zadaje kao parametar i razvrstava ih po dužini. Parametar funkcije je tipa "string". Smatraćemo da je riječ bilo koja neprekinuta sekvenca alfanumeričkih znakova (tj. znakova koji su slova ili cifre), a koja je s obje strane omeđena nečim što nije alfanumerički znak (tj. razmakom, znakom interpunkcije, itd.), ili početkom odnosno krajem rečenice. Na primjer, u rečenici "Kisa pada, trava raste, a gora zeleni!", riječi su "Kisa", "pada", "trava", "raste", "a", "gora" i "zeleni". Kao rezultat funkcija treba da vrati moderni niz od 100 elemenata, čiji je element s indeksom i vektor koji se sastoji od svih riječi (tipa "string") koje su dugačke tačno i znakova, smještene u onom redoslijedu na koje smo nailazili pri analizi rečenice slijeva nadesno, i sa svim slovima pretvorenim u velika slova. Recimo, za rečenicu iz prethodnog primjera, element s indeksom 1 sadržavaće riječ "A", element s indeksom 4 sadržavaće riječi "KISA", "PADA" i "GORA" (tim redom), element s

indeksom 5 sadržavaće riječi "TRAVA" i "RASTE" (tim redom), element s indeksom 6 sadržavaće riječ "ZELENI", a svi ostali elementi sadržavaće prazne vektore (jer nema riječi s tim brojem slova). Dimenzija 100 za izlazni moderni niz odabrana je uz pretpostavku da niti jedan svjetski jezik nema nijednu razumnu riječ s brojem slova većim od 99. Ukoliko se pri obradi rečenice nađe duža riječ, funkcija treba baciti izuzetak tipa "`range_error`" uz prateći tekst "Predugacka rijec!". Inače, u bosanskom jeziku, najduža riječ u izvornom obliku je "prijestolonasljednikovica" (što znači žena od prijestolonasljednika), ali ukoliko je stavimo u deminutivni oblik, zatim u dativ, instrumental ili lokativ množine, dobijamo riječ "prijestolonasljednikovičicinima", koja je službeno najduža riječ bosanskog jezika (33 slova). Pored ove, još neke podugačke riječi bosanskog jezika su recimo "antisamoupravnosocijalistički" ili "ezofagogastroduodenoskopija". U nekim germanskim jezicima, koji imaju sklonost ka pravljenju tzv. složenica (npr. njemački jezik), ili u aglutinativnim jezicima kod kojih se riječi tvore spajanjem kraćih korijenskih riječi (npr. turski jezik), nisu nikakva rijetkost ni daleko duže riječi. Recimo, u njemačkom jeziku postoji riječ (dužine 63 slova) "Rindfleischetikettierungsüberwachungsaufgabenübertragungsgesetz" (koja u prevodu znači "zakon o delegaciji dužnosti nadzora obilježavanja i označavanja goveđeg mesa", dok u turskom jeziku postoji riječ "Muvaffakiyetsizlestiricilestiriveremeyebileceklerimizdenmissinizcesine" (u prevodu "kao da ste jedan od onih koje možda nećemo moći učiniti neuspješnim").

Napisanu funkciju trebate testirati u testnom programu ("main" funkciji) koji analizira rečenicu unesenu s tastature, i ispisuje rezultate testiranja na način vidljiv iz sljedećih primjera dijaloga između korisnika i programa:

```
Unestite recenicu: Kisa pada, trava raste, a gora zeleni!

Duzina 1: A
Duzina 4: KISA PADA GORA
Duzina 5: TRAVA RASTE
Duzina 6: ZELENI
```

```
Unestite recenicu: Kazu da je starogrcki pisac Aristofan sastavio rijec od 183 slova
lopadotemachoselachogaleokranioleipsanodrimhupotrimmatosilphiokarabomelitokatakechum
enokichlepikossuphophattoperisteralektruonoptokephalliokigklopeleiolagoiosiraiobaphe
traganopterugon, tako da ima (nerazumnih) rijeci koje su duze od 100 znakova...

Greska: Recenica sadrzi predugacku rijec!
```