# Zadaća 2.

***Ova zadaća nosi ukupno 4,5 poena, od kojih prvi zadatak nosi 1,3 poena, a ostali po 0,8 poena. Svi zadaci se mogu uraditi na osnovu gradiva s prvih 6 predavanja i pretpostavljenog predznanja iz predmeta "Osnove računarstva". Svi zadaci osim prvog mogu se uraditi tako da budu prilično kratki, a i prvi zadatak se može uraditi u manje od 150 linija kôda ukoliko se malo inteligentnije osmisle neke stvari, a ne samo nabacuje nepregledna sekvenca if-naredbi (što ne vodi ničemu dobrom). Rok za predaju ove zadaće je ponedjeljak, 17. IV 2023. 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. Neki robotski manipulator može se kretati po pravougaonom terenu. Lokacija robota na terenu opisuje se Descartesovim koordinatama $(x, y)$, pri čemu $x$ može biti između $-30$ i 30, a $y$ između $-10$ i 10 uključivo (tako da se središte terena nalazi na lokaciji poziciji $(0, 0)$). Prilikom kretanja, robot se može isključivo zaustavljati na lokacijama koje imaju cjelobrojne koordinate. U slučaju se robot nalazi na nekom od krajeva terena, a naloži mu se da ide u smjeru koji izlazi van terena, robot se magično "teleportira" na suprotnu stranu terena (ovo se u matematici naziva *topologija torusa*), slično kao u poznatoj staroj igrici PacMan, odnosno u skladu s jednim od modela kako ravnozemljaši objašnjavaju šta se dešava ukoliko neko dođe do "ruba" ploče koja predstavlja ravnu zemlju. Za upravljanje ovim manipulatorom koristi se skup upravljačkih funkcija čiji su prototipovi sljedeći:

   ```
   void Idi(int &x, int &y, Pravci orijentacija, int korak);
   void Rotiraj(Pravci &orijentacija, int ugao);
   void PostaviVidljivost(bool vidljiv);
   void IspisiPoziciju(int x, int y, Pravci orijentacija);
   void PrikaziTeren();
   void PrijaviGresku(KodoviGresaka kod_greske);
   void IzvrsiKomandu(Komande komanda, int parametar, int &x, int &y,
     Pravci &orijentacija);
   ```

   Funkcija "`Idi`" pomjera robota za broj koraka zadan parametrom "`korak`" u smjeru koji je određen parametrom "`orijentacija`". Tip ovog parametra je "`Pravci`", što predstavlja pobrojani tip koji je deklariran na globalnom nivou kao

   ```
   enum class Pravci {Sjever, Sjeveroistok, Istok, Jugoistok, Jug, Jugozapad, Zapad,
     Sjeverozapad};
   ```

   Značenja pojedinih pobrojanih konstanti u ovom tipu su očigledna. Za teren se pretpostavlja da je orijentiran tako da kretanje na sjever odgovara povećanju $y$ koordinate, uz zadržavanje $x$ koordinate nepromijenjenom. U slučaju "mješovitih" pravaca kao što je npr. jugoistok jedan korak na jugoistok tretira kao da je ekvivalentan jednom koraku na jug i jednom koraku na istok (slično vrijedi i za sve ostale "mješovite" pravce). Trenutna pozicija robota određena je vrijednostima parametara "`x`" i "`y`" na ulazu, a po završetku funkcije isti parametri trebaju sadržavati ažuriranu poziciju. Ukoliko je parametar "`korak`" negativan, kretanje se vrši u smjeru suprotnom od tekuće orijentacije za iznos jednak apsolutnoj vrijednosti parametra.

   Funkcija "`Rotiraj`" mijenja pravac u kojem robot gleda, odnosno obrće ga nalijevo ili nadesno za ugao od $n \cdot 45°$ u smjeru suprotnom od kazaljke na satu, pri čemu se $n$ zadaje parametrom "`ugao`" (recimo, $n = 2$ odgovara rotaciji za 90°). Parametar $n$ može biti i negativan, ali tada se rotacija vrši u smjeru kazaljke na satu. Parametar "`orijentacija`" predstavlja pravac u kojem robot trenutno gleda, pri čemu funkcija utiče na njegovu vrijednost, tako da će on po završetku funkcije sadržavati novu orijentaciju robota nakon obavljene rotacije.

   Funkcija "`PostaviVidljivost`" postavlja vidljivost robota na vrijednost zadanu parametrom "`vidljiv`", pri čemu "`true`" govori da je robot vidljiv, a "`false`" da je nevidljiv. Sve dok je robot vidljiv, prati se (tj. bilježi) svaka lokacija na kojoj se robot nađe tokom svog kretanja. Međutim, dok je robot nevidljiv, njegovo kretanje se ne prati, odnosno on se može kretati kuda god hoće, a da detalji o tome kuda se tačno kretao i gdje je za to vrijeme bio ne budu zabilježeni.

Funkcija "IspisiPoziciju" ispisuje na ekran informacije o poziciju robota u sljedećem formatu:

**Robot je** *status*, **nalazi se na poziciji (**$x$, $y$**) i gleda na** *orijentacija*.

"*status*" može biti "vidljiv" ili "nevidljiv", ovisno da li je robot trenutno aktivan ili ne. "$x$" i "$y$" su pozicija robota, koja je određena istoimenim parametrima, dok "*orijentacija*" može biti "sjever", "sjeveroistok", "istok", "jugoistok", "jug", "jugozapad", "zapad" ili "sjeverozapad", u zavisnosti od vrijednosti istoimenog parametra.

Funkcija "PrikaziTeren" prikazuje stanje na terenu na ekranu. Tačan izgled prikaza se može vidjeti u primjeru dijaloga između korisnika i programa, a ukratko izgleda ovako. Teren je omeđen znakovima "#" koji predstavljaju "ogradu" oko terena. Trenutna pozicija robota (bio on vidljiv ili ne) prikazana je znakom "0", dok su sve lokacije na kojima se robot nalazio dok je bio vidljiv označene zvjezdicom (znakom "*"). Sve ostale pozicije (tj. neposjećene, ili one na kojima se robot nalazio *jedino dok je bio nevidljiv*) ostaju prazne.

Funkcija "PrijaviGresku" ima parametar "kod_greske" tipa "KodoviGresaka" koji predstavlja pobrojani tip definiran kao

```
enum class KodoviGresaka {PogresnaKomanda, NedostajeParametar, SuvisanParametar,
  NeispravanParametar};
```

Ova funkcija, u zavisnosti od vrijednosti parametra koji joj je proslijeđen, na ekran ispisuje neki od tekstova koji se mogu javiti kao greška u radu sa robotom, u skladu sa sljedećom tabelom (objašnjenje će uslijediti poslije):

| Kôd greške: | Tekst koji treba ispisati: |
|---|---|
| PogresnaKomanda | Nerazumljiva komanda! |
| NedostajeParametar | Komanda trazi parametar koji nije naveden! |
| NeispravanParametar | Parametar komande nije ispravan! |
| SuvisanParametar | Zadan je suvisan parametar nakon komande! |

Funkcija "IzvrsiKomandu" ima parametar "komanda" tipa "Komande" koji predstavlja pobrojani tip definiran kao

```
enum class Komande {Idi, Rotiraj, Sakrij, Otkrij, PrikaziTeren, Kraj};
```

Ova funkcija, u zavisnosti od vrijednosti parametra "komanda", izvršava zadanu komandu, pozivom odgovarajuće funkcije za izvršenje komande (osim ukoliko je komanda "Kraj"; tada funkcija ne radi ništa). Parametar "parametar" koristi se samo ukoliko je komanda "Idi" ili "Rotiraj", i on tada predstavlja vrijednost koja se prosljeđuje kao posljednji parametar istoimenim funkcijama (tj. funkcijama "Idi" odnosno "Rotiraj"). U svim ostalim slučajevima, vrijednost ovog parametra se ignorira. Preostali parametri funkcije "IzvrsiKomandu" predstavljaju informaciju o poziciji i orijentaciji robota.

Komunikacija između korisnika i robota vrši se posredstvom funkcije "UnosKomande" koja ima sljedeći prototip:

```
bool UnosKomande(Komande &komanda, int &parametar, KodoviGresaka &kod_greske);
```

Ova funkcija očekuje od korisnika da zada unos komande putem tastature. Legalne komande su sljedeće (u komandama "S+" i "S-" znakovi "+" i "-" su dio komande a ne parametar):

| Komanda: | Značenje |
|---|---|
| **I** *korak* | Pomjera robota za navedeni broj koraka |
| **R** *ugao* | Rotira robota za navedeni ugao, pri čemu je jedinica mjere 45° |
| **S+** | Sakriva robota, tj. čini ga nevidljivim |
| **S–** | Otkriva robota, tj. čini ga vidljivim |
| **T** | Prikazuje stanje na terenu |
| **K** | Završetak rada programa |

Razmaci ispred i iza komande su dozvoljeni, ali bilo kakav neočekivani znak (osim razmaka) nakon komande tretira se kao suvišan parametar. Komande "I" odnosno "R" su praćene cijelim brojem (koji može biti i negativan) koji predstavlja broj koraka koji će robot napraviti, odnosno ugao

rotacije. Razmaci između komande i broja su dozvoljeni (ali ne i obavezni), pri čemu se bilo šta što ne predstavlja ispravan cijeli broj (uključujući suvišne znakove iza broja) tretira kao neispravan parametar. U slučaju da se prepozna ispravna komanda, funkcija smješta njen kôd u parametar "`komanda`", eventualni parametar komande smješta se u parametar "`parametar`" (u slučaju da komanda nema parametra, vrijednost parametra "`parametar`" je nedefinirana), a funkcija vraća kao rezultat logičku vrijednost "**`true`**" kao signal da je komanda prepoznata. Parametar "`kod_greske`" tada je nedefiniran. U suprotnom, ukoliko nije prepoznata ispravna komanda, kôd greške se smješta u parametar "`kod_greske`", parametri "`komanda`" i "`parametar`" su nedefinirani, a funkcija vraća kao rezultat logičku vrijednost "**`false`**" kao signal da nije prepoznata ispravna komanda. Eventualni razmaci ispred komande kao i nakon komande su dozvoljeni i ne trebaju uzrokovati prikavu greške. Potrebno je predvidjeti sve što bi korisnik eventualno mogao unijeti (funkcija kao i program koji je koristi ne smije da "crkne" šta god korisnik unio).

Na početku rada, robot je vidljiv, nalazi se na poziciji (0, 0) i gleda na sjever. Napisane funkcije treba demonstrirati u glavnom programu u kojem će se u petlji zahtijevati unos komande pozivom funkcije "`UnosKomande`", nakon čega će se odgovarajuća komanda izvršiti (pozivom funkcije "`IzvrsiKomandu`") ili će se prijaviti greška (pozivom funkcije "`PrijaviGresku`"). Petlja se prekida nakon što se zada komanda "`Kraj`". Tada program ispisuje poruku "Dovidjenja!" i završava sa radom. Slijedi primjer kako može izgledati dijalog između korisnika i programa:

```
Robot je vidljiv, nalazi se na poziciji (0,0) i gleda na sjever.
Unesi komandu: I7
Robot je vidljiv, nalazi se na poziciji (0,7) i gleda na sjever.
Unesi komandu: R1
Robot je vidljiv, nalazi se na poziciji (0,7) i gleda na sjeverozapad.
Unesi komandu: I40
Robot je vidljiv, nalazi se na poziciji (21,5) i gleda na sjeverozapad.
Unesi komandu: S+
Robot je nevidljiv, nalazi se na poziciji (21,5) i gleda na sjeverozapad.
Unesi komandu: R-3
Robot je nevidljiv, nalazi se na poziciji (21,5) i gleda na istok.
Unesi komandu: I25
Robot je nevidljiv, nalazi se na poziciji (-15,5) i gleda na istok.
Unesi komandu: R5
Robot je nevidljiv, nalazi se na poziciji (-15,5) i gleda na jugozapad.
Unesi komandu: I-3
Robot je nevidljiv, nalazi se na poziciji (-12,8) i gleda na jugozapad.
Unesi komandu: S-
Robot je vidljiv, nalazi se na poziciji (-12,8) i gleda na jugozapad.
Unesi komandu: I12
Robot je nevidljiv, nalazi se na poziciji (-24,-4) i gleda na jugozapad.
Unesi komandu: T
##################################################################
#      *                     *                                   #
#       *                     *                                  #
#        *         *           *                                 #
#         *       *             *                                #
#          *     *              *                                #
#           *   *               *                    *           #
#            * *                *                   *           #
#             *                 *                  *            #
#            * *                *                 *             #
#           *   *               *                *              #
#          *     *              *               *               #
#         *       *                            *                #
#        *         *                          *                 #
#       *           *                        * #                #
#      0             *                      *#                 #
#*                    *                                        #
# *                    *                                       #
#  *                    *                                      #
#   *                    *                                     #
#    *                    *                                    #
#     *                    *                                   #
##################################################################
Unesi komandu: K
Dovidjenja!
```

```
Robot je vidljiv, nalazi se na poziciji (0,0) i gleda na sjever.
Unesi komandu: R-2
Robot je vidljiv, nalazi se na poziciji (0,0) i gleda na istok.
Unesi komandu: I
Komanda trazi parametar koji nije naveden!
Unesi komandu: I  5
Robot je vidljiv, nalazi se na poziciji (5,0) i gleda na istok.
Unesi komandu: S+
Robot je nevidljiv, nalazi se na poziciji (5,0) i gleda na istok.
Unesi komandu:    R -1
Robot je nevidljiv, nalazi se na poziciji (5,0) i gleda na jugoistok.
Unesi komandu:    I4
Robot je nevidljiv, nalazi se na poziciji (9,-4) i gleda na jugoistok.
Unesi komandu: R7
Robot je nevidljiv, nalazi se na poziciji (9,-4) i gleda na jug.
Unesi komandu: M
Nerazumljiva komanda!
Unesi komandu: S-
Robot je vidljiv, nalazi se na poziciji (9,-4) i gleda na jug.
Unesi komandu: IXY2
Parametar komande nije ispravan!
Unesi komandu: R-2
Robot je vidljiv, nalazi se na poziciji (9,-4) i gleda na zapad.
Unesi komandu: I2XY
Parametar komande nije ispravan!
Unesi komandu: I0
Robot je vidljiv, nalazi se na poziciji (9,-4) i gleda na zapad.
Unesi komandu: I3
Robot je vidljiv, nalazi se na poziciji (6,-4) i gleda na zapad.
Unesi komandu: T2
Zadan je suvisan parametar nakon komande!
Unesi komandu: R3
Robot je aktivan, nalazi se na poziciji (6,-4) i gleda na jugoistok.
Unesi komandu: I5
Robot je aktivan, nalazi se na poziciji (11,-9) i gleda na jugoistok.
Unesi komandu: T
####################################################################
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                                                                  #
#                              *****                               #
#                                                                  #
#                                                                  #
#                                                                  #
#                                    ****                          #
#                                   *                              #
#                                    *                             #
#                                     *                            #
#                                      *                           #
#                                       0                          #
#                                                                  #
####################################################################
Unesi komandu: R4
Robot je aktivan, nalazi se na poziciji (11,-9) i gleda na sjeverozapad.
Unesi komandu: R0
Robot je aktivan, nalazi se na poziciji (11,-9) i gleda na sjeverozapad.
Unesi komandu: I12
Robot je aktivan, nalazi se na poziciji (-1,3) i gleda na sjeverozapad.
Unesi komandu: R5
Robot je aktivan, nalazi se na poziciji (-1,3) i gleda na istok.
Unesi komandu: KK
Zadan je suvisan parametar nakon komande!
Unesi komandu: K
Dovidjenja!
```

VAŽNA NAPOMENA: Informacije o poziciji i orijentaciji robota *ne smiju se čuvati u globalnim promjenljivim*, nego se te informacije moraju razmjenjivati između funkcija putem prenosa parametara. Također, u programu *nije dozvoljeno koristiti unos sa tastature u promjenljive tipa string ili niz znakova*, nego sva procesiranja ulaza treba vršiti direktnim čitanjima iz spremnika ulaznog toka. Nepoštovanje ovih ograničenja biće kažnjeno davanjem 0 poena na čitav zadatak!

2. U elementarnoj teoriji brojeva značajno mjesto zauzimaju tzv. *brojevi slobodni od kvadrata* (engl. *squarefree*). To su oni *prirodni* brojevi u čijoj se rastavi na proste faktore niti jedan prosti faktor ne javlja sa eksponentom većim od 1 (recimo, broj $2805 = 3 \cdot 5 \cdot 11 \cdot 17$ je slobodan od kvadrata, dok broj $1960 = 2^3 \cdot 5 \cdot 2^2$ nije). Za broj 1 se također smatra da je slobodan od kvadrata (inače, naziv "slobodan od kvadrata" potiče od toga što su ti brojevi jedinstveni po osobini da nisu djeljivi niti sa jednim kvadratom nekog prirodnog broja). Dalje, u elementarnoj teoriji brojeva je poznato da se svaki prirodan broj $n$ može na jedinstven način predstaviti u obliku $n = p\,q^2$, gdje su $p$ i $q$ prirodni brojevi, a $p$ je pored toga i slobodan od kvadrata. Na primjer, za $n = 22360800$ imamo $p = 462$ i $q = 220$, pri čemu je broj 462 slobodan od kvadrata.

   Vaš zadatak je da napravite funkciju "`RastavaBroja`" sa 3 cjelobrojna parametra (tipa "**`int`**"), koja za prirodan broj $n$ koji se zadaje kao prvi parametar nalazi prirodne brojeve $p$ i $q$ iz rastave $n = p\,q^2$ i smješta ih u drugi i treći parametar funkcije respektivno. Funkcija treba da radi i za negativne brojeve $n$, pri čemu će za $n < 0$ vrijednost $p$ također biti negativna (tako da ćemo za $n = -22360800$ imati $p = -462$ i $q = 220$), te za $n = 0$, pri čemu tada treba uzeti $p = 0$ i $q = 1$. Napisanu funkciju demonstrirajte na testnom progamu koji za cijeli broj $n$ unesen sa tastature ispisuje rastavu oblika $n = p\,q^2$, gdje su $p$ i $q$ dobijeni gore opisanim postupkom. Množenje prikažite znakom "*", a kvadriranje sufiksom "^2". Tako, dijalog između korisnika i programa treba da izgleda kao na sljedećoj slici:

```
Unesite broj: -22360800
-22360800 = -462*220^2
```

   NAPOMENA: Postoji mnogo načina da se obavi tražena rastava. Neki su elementarni u smislu da ih student može otkriti sam, a neki traže dublje poznavanje teorije brojeva (što se od studenata naravno ne očekuje). Recimo, jedan od elementarnih načina (ali ne i jedini, jer postoje i lakši načini, pokušajte ih sami otkriti) zasniva se na rastavi broja na proste faktore. Na primjer, za $n = 22360800$ možemo pisati:

   $$22360800 = 2^5 \cdot 3 \cdot 5^2 \cdot 7 \cdot 11^3 = 2 \cdot 2^4 \cdot 3 \cdot 5^2 \cdot 7 \cdot 11 \cdot 11^2 = 2 \cdot 3 \cdot 7 \cdot 11 \cdot (2^2 \cdot 5 \cdot 11)^2$$

   odakle očitavamo $p = 2 \cdot 3 \cdot 7 \cdot 11 = 462$ i $q = 2^2 \cdot 5 \cdot 11 = 220$.

3. Neka su $\boldsymbol{A}$ i $\boldsymbol{B}$ matrice formata $m \times n$ i $n \times p$. Poznato je da je proizvod ove dvije matrice matrica $\boldsymbol{C}$ čiji su elementi određeni formulom

   $$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j} = a_{i,1} b_{1,j} + a_{i,2} b_{2,j} + a_{i,3} b_{3,j} + \ldots + a_{i,n} b_{n,j},\ i = 1,2,\ldots,m,\ j = 1,2,\ldots,p$$

   Generalizirani matrični proizvod definira se na sličan način, osim što se sabiranje i množenje respektivno zamjenjuju s proizvoljne dvije binarne operacije opisane funkcijama $f$ i $g$ s dva argumenta (tako da se za $f(x,y) = x + y$ i $g(x,y) = x\,y$ dobija klasični matrični proizvod. Drugim riječima, elementi $c_{i,j}$ u generaliziranom matričnom proizvodu računaju se po formuli

   $$c_{i,j} = f(\ldots f(f(g(a_{i,1}, b_{1,j}), g(a_{i,2}, b_{2,j})), g(a_{i,3}, b_{3,j})), \ldots), g(a_{i,n}, b_{1,n})),\ i = 1,2,\ldots,m,\ j = 1,2,\ldots,p$$

   Napišite generičku funkciju "`GeneraliziraniMatricniProizvod`" koja računa generalizirani matrčini proizvod matrica koje koje joj se prenose kao parametri. Matrice su organizirani kao vektori vektôrā proizvoljnog tipa elemenata, pri čemu se tip elemenata u matricama $\boldsymbol{A}$ i $\boldsymbol{B}$ može razlikovati. Treći i četvrti parametar su funkcije $f$ i $g$, pri čemu funkcija $f$ ima dva parametra koji su onakvog tipa kakav vraća funkcija $g$ kao rezultat i koja daje rezultat istog takvog tipa, dok funkcija $g$ ima dva parametra koji su onakvih tipova kakvi su elementi matrica $\boldsymbol{A}$ i $\boldsymbol{B}$, a vraća rezultat koji može biti i nekog drugog tipa. Pri tome, da biste izbjegli vrlo složenu deklaraciju ova dva parametra kakva bi bila neophodna prema prethodnoj specifikaciji, pustite da se tip ova dva parametra automatski

određuje putem potpune dedukcije (jedini nedostatak ovog pristupa je što će se instantacija funkcije pokušati šta god da pošaljemo ovoj funkciji kao treći i četvrti parametar, koja će onda eventualno pasti u slučaju da ti parametri nisu validni). Kao rezultat, funkcija treba da vrati matricu onakvog tipa elemenata kakav vraća funkcija $f$ (odnosno $g$, s obzirom da se radi o istim tipovima), a koja predstavlja generalizirani matrični proizvod matrica koje su prenesene kao prva dva parametra. U slučaju da parametri nisu matrice propisnog formata za koji je matrično množenje moguće, funkcija treba da baci izuzetak tipa "`domain_error`" uz prateći tekst "Matrice nisu saglasne za mnozenje". U slučaju da tokom računanja generaliziranog matričnog proizvoda bilo koja od funkcija $f$ ili $g$ baci bilo kakav izuzetak, funkcija za računanje generaliziranog matričnog proizvoda treba baciti izuzetak tipa "`runtime_error`" uz prateći tekst "Neocekivani problemi pri racunanju".

Napisanu funkciju demonstrirajte u testnom programu koji traži da se sa tastature unesu brojevi $m$, $n$, i $p$, a zatim elementi matrica $\boldsymbol{A}$ i $\boldsymbol{B}$ koji će biti tipa "`string`", a za koje se pretpostavlja da predstavljaju neke riječi. Program zatim treba da izračuna generalizirani matrični proizvod unesenih matrica tako što će $f$ biti funkcija koja nadovezuje dva stringa jedan na drugi uz dodavanje znaka "+" između (tako da je npr. $f$("a", "b") = "a+b"), dok je $g$ funkcija koja nadovezuje dva stringa jedan na drugi uz dodavanje znaka "*" između (tako da je npr. $f$("a", "b") = "a*b"). Na kraju, treba ispisati elemente dobijene matrice na ekran, koristeći poravnanje svakog elementa ulijevo, pri čemu je širina ispisa prilagođena dužini najdužeg stringa u matrici uvećanoj za 1. Tako, primjer dijaloga između korisnika i programa može izgledati kao na sljedećoj slici:

```
Unesite broj redova prve matrice: 2
Unesite broj kolona prve matrice, ujedno broj redova druge matrice: 3
Unesite broj kolona druge matrice: 2

Unesite elemente prve matrice:
a11 a12 a13
a21 a22 a14
Unesite elemente druge matrice:
b11 b12
b21 b22
b31 b32

Matricni proizvod:
a11*b11+a12*b21+a13*b31 a11*b12+a12*b22+a13*b32
a21*b11+a22*b21+a23*b31 a21*b12+a22*b22+a23*b32
```

NAPOMENA: Bez obzira što testni program radi samo s matricama čiji su elementi stringovi, napisana funkcija će se testirati s više različitih matrica, različitih tipova elemenata. Ukoliko ne znate napraviti dovoljno općenitu funkciju koliko se traži u zadatku, napravite je da radi za što više specijalnih slučajeva. Dobićete nešto bodova i na takvu izvedbu.

4. Napravite generičku funkciju "`SortirajPoProizvoduRedova`" koja kao parametar kao jedini parametar neku matricu (koja može biti i grbava) organiziranu kao vektor vektôrā proizvoljnog tipa elemenata, za koje se jedino pretpostavlja da se mogu množiti i porediti po veličini (tj. da je za njih podržana operacija množenja, te operacije poređenja po veličini), kao i da se rezultati množenja, ma kakvog tipa oni bili, mogu porediti po veličini (ovo recimo isključuje hipotetičke tipove za koji bi rezultat množenja dva objekta tog tipa bio kompleksan broj, čak i ukoliko se sami objekti tog tipa mogu porediti po veličini). Funkcija treba da transformira tu matricu tako da ona postane sortirana po proizvodu redova u rastući poredak, u smislu da nakon sortiranja njen prvi red bude onaj red koji ima najmanji proizvod elemenata, i tako dalje sve do posljednjeg reda, koji je red s najvećim proizvodom elemenata. Ukoliko dva reda imaju isti proizvod elemenata, tada prije treba da dođe onaj red koji dolazi ranije u leksikografskom poretku, koji se određuje na osnovu prvog para različitih elemenata. Na primjer, vektor čiji su elementi 3, 5, 2, 7, 1 i 9 dolazi leksikografski prije vektora čiji su elementi 3, 5, 8, 4 i 6, jer je $2 < 8$). Funkcija ne vraća nikakav rezultat, nego samo modificira matricu koja joj se šalje kao parametar.

   Traženu funkciju obavezno treba realizirati uz pomoć funkcije "`sort`" iz biblioteke "`algorithm`", kojoj treba proslijediti odgovarajuću funkciju kriterija, koju treba izvesti kao imenovanu funkciju nazvanu "`Kriterij`" (šta će biti parametri ove funkcije, zaključite sami). Pri tome će ova funkcija kriterija također morati biti generička funkcija, s obzirom da tip elemenata matrice nije unaprijed

poznat. Vodite računa da ćete, kada budete slali generičku funkciju kao parametar funkciji "`sort`", morati koristiti eksplicitnu specifikaciju parametra šablona, jer je dedukcija tipa moguća samo na osnovu parametara koji se prosljeđuju funkciji, a kad funkciju kriterija prosljeđujemo kao parametar drugoj funkciji, tom prilikom ne zadajemo joj nikakve parametre.

Napisanu funkciju demonstrirajte u testnom programu u kojem se elementi grbave matrice cijelih brojeva unose red po red, pri čemu oznaku kraja reda predstavlja bilo šta što nije broj (recimo "*"). Unos se završava kada se odmah na početku reda unese nešto što nije broj. Nakon obavljenog unosa, program treba da sortira matricu na opisani način, i da ispiše njene elemente nakon obavljenog sortiranja. Konačno, nakon ispisa, program treba da traži unos elemenata neke sekvence cijelih brojeva, nakon čega postupkom binarne pretrage testira da li se unesena sekvenca pojavljuje kao neki od redova ranije unesene matrice. Ovisno od rezultata testiranja, program treba da u slučaju uspješne pretrage ispiše rečenicu "Trazena sekvenca se nalazi u *i*. redu (nakon sortiranja)", pri čemu je $i$ red matrice koji je identičan unesenoj sekvenci (nakon obavljenog sortiranja), odnosno rečenicu "Trazena sekvenca se ne nalazi u matrici" u slučaju neuspješne pretrage. Ukoliko ima više redova koji su identični unesenoj sekvenci, treba prijaviti prvi od njih. Za pretragu koristite isključivo funkciju "`lower_bound`" iz biblioteke "`algorithm`" (i nijednu drugu). Dijalog između korisnika i programa treba da izgleda poput sljedećeg:

```
Unesite elemente (* za kraj reda, * na pocetku reda za kraj unosa):
2 5 1 6 7 *
9 8 9 *
3 3 2 3 *
4 5 2 1 1 3 2 *
*

Matrica nakon sortiranja:
3 3 2 3
4 5 2 1 1 3 2
2 5 1 6 7
9 8 9

Unesite elemente sekvence koja se trazi (* za kraj reda): 2 5 1 6 7 *
Trazena sekvenca se nalazi u 3. redu (nakon sortiranja)
```

5. Date su dvije sekvence $a_1, a_2, \ldots, a_n$ i $b_1, b_2, \ldots, b_n$ iste dužine. Pretpostavlja se da se elementi prve sekvence mogu sabirati s elementima druge sekvence i da je pri tome sabiranje komutativno, tj. da ma koje $a_i$ i $b_j$ vrijedi $a_i + b_j = b_j + a_i$. Potrebno je napraviti trougaonu tablicu sabiranja, koja će sadržavati zbirove $a_i + b_j$ za $i = 1, 2, \ldots, n$ i $j = 1, 2, \ldots, i$, kao na sljedećoj slici:

$$
\begin{array}{lllllll}
a_1+b_1 & & & & & & \\
a_2+b_1 & a_2+b_2 & & & & & \\
a_3+b_1 & a_3+b_2 & a_3+b_2 & & & & \\
\ldots & \ldots & \ldots & & & & \\
a_{n-1}+b_1 & a_{n-1}+b_2 & a_{n-1}+b_3 & \ldots & a_{n-1}+b_{n-1} & & \\
a_n+b_1 & a_n+b_2 & a_n+b_3 & \ldots & a_n+b_{n-1} & a_n+b_n &
\end{array}
$$

Vaš zadatak je da napravite generičku funkciju nazvanu "`KreirajTablicuSabiranja`" koja kreira elemente ovakve tablice. Prva dva parametra funkcije su pokazivači ili iteratori koji omeđuju sekvencu $a_1, a_2, \ldots, a_n$, dok je treći parametar pokazivači ili iterator koji pokazuje na sekvencu $b_1, b_2, \ldots, b_n$, za koju se pretpostavlja da je iste dužine. Tipovi prva dva parametra moraju biti identični, ali tip trećeg parametra ne mora biti identičan tipovima prva dva parametra (npr. prva dva parametra mogu biti pokazivači na objekte tipa "**`double`**", a treći parametar iterator za dek cijelih brojeva). To, između ostalog, povlači i da elementi ove dvije sekvence ne moraju nužno biti istog tipa. Funkcija treba dinamički alocirati grbavu matricu čiji prvi red ima jedan element, drugi red dva elementa, itd. i popuniti je rezultatima sabiranja. Kreiranje treba obaviti postupkom *kontinualne alokacije*. Tip elemenata alocirane matrice treba da bude onakav kakvog je tipa rezultat sabiranja elemenata iz prve i druge sekvence (recimo, ako prva sekvenca sadrži realne, a druga kompleksne brojeve, elementi matrice trebaju biti kompleksnog tipa). Kao rezultat, funkcija treba da vrati dvojni pokazivač pomoću kojeg se može pristupiti elementima kreirane matrice. Ukoliko se dogodi da alokacija ne uspije zbog nedovoljne količine raspoložive memorije, funkcija

treba baciti izuzetak tipa "`range_error`" uz prateći tekst "Nema dovoljno memorije". Pri tome, treba paziti da ni u kom slučaju ne smije doći do curenja memorije.

Kao što je već rečeno, polazimo od pretpostavke da je sabiranje komutativno. Međutim, ukoliko funkcija ustanovi da to nije ispunjeno, odnosno da za makar jedan par indeksa $i$ i $j$ vrijedi da je $b_j + a_i \neq a_i + b_j$ (to se može, recimo, dogoditi ukoliko su elementi sekvence stringovi, s obzirom da sabiranje stringova u općem slučaju nije komutativno), funkcija treba baciti izuzetak tipa "`logic_error`" uz prateći tekst "Nije ispunjena pretpostavka o komutativnosti".

Da bi funkcija bila što univerzalnija, za parametre funkcija nemojte pretpostavljati da podržavaju nikakve druge operacije od onih operacija koje svaki iterator mora da podržava (nisu svi iteratori jednako moćni s aspekta operacija koje podržavaju, kao što će biti objašnjeno kasnije na Predavanju 7_b), a to su operacije dereferenciranja ("`*`"), dodjele ("`=`"), inkrementiranja ("`++`"), te poređenja na jednakost i različitost ("`==`" i "`!=`"). Vodite računa da će funkcija biti testirana i sa kontejnerima čiji iteratori ne podržavaju ništa više od ovih operacija (vidjećete kasnije da postoje i takvi kontejnerski tipovi). Ovim ograničenjem na korištene operatore znatno se proširuje univerzalnost funkcije, jer će se moći primijeniti i s takvim kontejnerima.

Napisanu funkciju demonstrirajte u kratkom testnom programu, koji traži unos dužine sekvenci kao i njihove elemente, od kojih prvu treba smjestiti u vektor a drugu u dek realnih brojeva, a koji zatim računa i ispisuje na ekran elemente kreirane tablice sabiranja, te na kraju oslobađa svu alociranu memoriju. Elementi svakog od redova se ispisuju razdvojeni po jednim razmakom. Dijalog između korisnika i programa treba da izgleda poput sljedećeg:

```
Duzina sekvenci: 4
Elementi prve sekvence: 5 2 8 7
Elementi druge sekvence: 1 3 6 2
Tablica sabiranja:
6
3 5
9 11 14
8 10 13 9
```

NAPOMENA: Ukoliko ne znate napisati funkciju da bude onoliko općenita koliko se traži u zadatku, napravite je da radi za što više specijalnih slučajeva. Dobićete nešto bodova i na takvu izvedbu.