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, 2. 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. U teoriji brojeva, susreće se pojam *multiplikativnog digitalnog korijena* (engl. *multiplicative digital root*). Multiplikativni digitalni korijen dobija se tako što se pomnože sve cifre tog broja, nakon čega se, ukoliko rezultat nije jednocifren broj, s novonastalim brojem ponavlja isti postupak, sve dok se ne dobije jednocifren broj kao rezultat. Taj rezultat je upravo traženi multiplikativni digitalni korijen. Na primjer, multiplikativni digitalni korijen broja 679 iznosi 6, jer imamo $6\cdot 7\cdot 9 = 378$, $3\cdot 7\cdot 8 = 168$, $1\cdot 6\cdot 8 = 48$, $4\cdot 8 = 32$ i, konačno, $3\cdot 2 = 6$. Multiplikativni digitalni korijen se može posmatrati i u odnosu na proizvoljnu cjelobrojnu bazu $B \geq 2$ (a ne nužno bazu 10), tako što se posmatraju cifre u bazi B (umjesto u bazi 10). Recimo, multiplikativni digitalni korijen broja 760 u bazi 7 iznosi 2. Naime, broj 760 predstavljen u bazi 7 glasi 2134 (jer je $760 = 2\cdot 7^3 + 1\cdot 7^2 + 3\cdot 7^1 + 4\cdot 7^0$), što možemo napisati kao $760 = (2134)_7$. Nakon toga, imamo $2\cdot 1\cdot 3\cdot 4 = 24 = (33)_7$, $3\cdot 3 = 9 = (12)_7$ i, konačno, $1\cdot 2 = 2$, što je jednocifren broj u bazi 7. Iz izloženog je jasno da multiplikativni digitalni korijen ma kojeg broja mora uvijek biti broj u opsegu od 0 do $B-1$, gdje je B razmatrana baza.

   Vaš zadatak započinje s kreiranjem funkcije nazvane "`MultiplikativniDigitalniKorijen`", koja kao parametre prima dva cijela broja, od kojih je prvi tipa "**`long long int`**", a drugi tipa "**`int`**" (tip prvog parametra je odabran tako da se što je god moguće više poveća opseg brojeva s kojima je moguće raditi). Prvi parametar predstavlja broj čiji se digitalni multiplikativni korijen traži, a drugi parametar je baza koja se pri tome koristi. Kao rezultat, funkcija treba da vrati traženi multiplikativni digitalni korijen. U slučaju da je broj čiji se multiplikativni digitalni korijen traži negativan, njegov znak prosto treba ignorirati. U slučaju da baza nije prirodan broj veći ili jednak od 2, funkcija treba da baci izuzetak tipa "`domain_error`" uz prateći tekst "Neispravna baza".

   Napisanu funkciju "`MultiplikativniDigitalniKorijen`" trebate iskoristiti kao potprogram unutar funkcije nazvane "`RazvrstajBrojeve`", koja razvrstava zadanu skupinu brojeva u klase prema njihovom multiplikativnom digitalnom korijenu u bazi 10. Ova funkcija prima dva parametra, pri čemu prvi parametar predstavlja vektor koji sadrži brojeve koji se razvrstavaju (elementi ovog vektora su tipa "**`long long int`**", dok će drugi parametar biti objašnjen malo kasnije. Funkcija kao rezultat treba da vrati moderni niz koji se sastoji od 10 vektora cijelih brojeva, dobijen kao rezultat razvrstavanja. Konkretnije, i-ti element ovog modernog niza (pri čemu je $0 \leq i \leq 9$) treba da bude vektor koji se sastoji od onih i samo onih elemenata ulaznog vektora (tj. vektora zadanog kao parametar) čiji multiplikativni digitalni korijen u bazi 10 iznosi i. Elementi svih tih vektora koji se dobijaju kao rezultat razvrstavanja trebaju da budu u istom međusobnom poretku u kakvom su bili u ulaznom vektoru.

   Što se tiče drugog parametra funkcije "`RazvrstajBrojeve`", on je tipa "`TretmanNegativnih`", pri čemu je "`TretmanNegativnih`" pobrojani tip s ograničenim vidokrugom, koji je deklariran na globalnom nivou pomoću deklaracije

   ```
   enum class TretmanNegativnih {IgnorirajZnak, Odbaci, PrijaviGresku};
   ```

   Ovaj parametar određuje kako će se tretirati negativni brojevi prilikom razvrstavanja. Ukoliko on ima vrijednost "`IgnorirajZnak`", negativnim brojevima se prosto ignorira znak, tj. tretiraju se kao da su pozitivni. Ukoliko je vrijednost ovog parametra "`Odbaci`", negativni brojevi se uopće ne razvrstavaju, tj. svi negativni brojevi se odbacuju prilikom razvrstavanja. Konačno, ukoliko ovaj parametar ima vrijednost "`PrijaviGresku`", nailazak na negativne brojeve prilikom razvrstavanja treba da dovede do bacanja izuzetka tipa "`domain_error`" uz prateći tekst "Nije predvidjeno razvrstavanje negativnih brojeva".

Potrebno je napisati mali testni program ("`main`" funkciju) u kojoj ćete testirati napisanu funkciju "`RazvrstajBrojeve`" na sekvenci brojeva koja se unose sa tastature. Brojevi se unose dok se ne unese bilo šta što nije broj (npr. tačka ili riječ "kraj"). Nakon sekvence, program treba da ispiše rezultate razvrstavanja, tako što će se u posebnim redovima ispisati vrijednost multiplikativnog digitalnog korijena, zatim dvotačka, te svi uneseni brojevi koji imaju taj multiplikativni digitalni korijen (u bazi 10). Redovi su poredani u rastućem poretku multiplikativnih digitalnih korijena, a ukoliko nema nijedan broj s nekim multiplikativnim digitalnim korijenom, taj red se uopće ne prikazuje. Slijedi primjer dijaloga između korisnika i programa:

```
Unesite brojeve (bilo koji ne-broj oznacava kraj): 34 142 679 98712 315 11331 467 39
31735 127 5361 222 3771 49913 11 .

Rezultati razvrstavanja po multiplikativnom digitalnom korijenu:
0: 98712 5361
1: 11
2: 34 49913
4: 39 127
5: 315 31735
6: 679 467 3771
8: 142 222
9: 11331
```

U slučaju da među unesenim brojevima ima i negativnih brojeva, program treba ispisati tekst "Nije podrzano razvrstavanje negativnih brojeva!", kao u sledećem dijalogu (obratite pažnju da ovaj tekst *nije istovjetan* tekstu bačenom u okviru izuzetka):

```
Unesite brojeve (bilo koji ne-broj oznacava kraj): 1 22 333 -4444 5555 -66666 .

Nije podrzano razvrstavanje negativnih brojeva!
```

2. Za potrebe digitalne obrade slika i općenito za potrebe pamćenja slika u računarskoj memoriji, slika se najčešće posmatra kao da je sastavljena od vrlo sitnih nedjeljivih kvadratića, nazvanih *pikseli*. Na taj način, slika se može modelirati kao matrica, pri čemu svaki od elemenata matrice odgovara po jednom pikselu, a sadrži informaciju o svjetlini ili eventualno boji odgovarajućeg piksela (na primjer, svijetlijim pikselima mogu odgovarati veći brojevi, a tamnijim pikselima manji brojevi). Međutim, uslijed grešaka u procesu fotografiranja, zna se desiti da se na slici mogu uočiti izvjesne smetnje, koje se tipično manifestiraju kao mrlje ili slične degradacije u kvalitetu slike. Zbog toga, među najvažnije postupke u digitalnoj obradi slike, spadaju postupci tzv. *filtriranja*, čiji je cilj da se smanji vizualni efekat smetnji.

   Postoji mnogo različitih tehnika filtriranja. Jedna od najjednostavnijih, ali i najlošijih tehnika je tzv. *usrednjavajući filter* (engl. *average filter* ili *mean filter*). Prema ovoj tehnici, oko svakog piksela formira se "prozor" veličine $(2\mathrm{N}+1)\times(2\mathrm{N}+1)$, gdje je N tzv. *red filtera* (prirodan broj). Tako, "prozoru" koji odgovara pikselu na poziciji (i, j) pripadaju svi pikseli kod kojih je prva koordinata u opsegu od $i-\mathrm{N}$ do $i+\mathrm{N}$, a druga koordinata u opsegu od $j-\mathrm{N}$ do $j+\mathrm{N}$. Zatim se vrijednost svakog od piksela zamjenjuje *srednjom vrijednošću* (odnosno *aritmetičkom sredinom*) vrijednosti svih piksela unutar prozora koji pripada tom pikselu. Pri tome, ukoliko se razmatra piksel koji je blizu nekog od ruba matrice, može se desiti da prozor dijelom izađe izvan slike odnosno matrice. Međutim, to nije bitno, s obzirom da se pri računu razmatraju samo oni pikseli koji se zaista nalaze unutar prozora. Drugim riječima, pri računu se uzimaju samo elementi matrice s legalnim koordinatama, tj. koordinatama koje ne izlaze izvan opsega koordinata matrice. Kao primjer, ispod lijevo je prikazana jedna slika-matrica, dok je s desne strane prikazana ista slika-matrica dobijena filtriranjem usrednjavajućim filterom reda $\mathrm{N}=1$ (tako da se filtriranje vrši usrednjavanjem unutar "prozora" veličine 3×3). U prikazu s desne strane izvršeno je zaokruživanje vrijednosti svih elemenata na 2 decimale:

$$\begin{pmatrix} 2 & 3 & 3 & 1 & 2 & 3 \\ 1 & 2 & 4 & 3 & 4 & 4 \\ 2 & 3 & 4 & 5 & 3 & 2 \\ 3 & 4 & 3 & 4 & 4 & 2 \\ 1 & 2 & 3 & 3 & 4 & 3 \end{pmatrix} \qquad \begin{pmatrix} 2 & 2.5 & 2.67 & 2.83 & 2.83 & 3.25 \\ 2.17 & 2.67 & 3.11 & 3.22 & 3 & 3 \\ 2.5 & 2.89 & 3.56 & 3.78 & 3.44 & 3.17 \\ 2.5 & 2.78 & 3.44 & 3.67 & 3.33 & 3 \\ 2.5 & 2.67 & 3.17 & 3.5 & 3.33 & 3.25 \end{pmatrix}$$

Vaš zadatak je da napravite funkciju "`UsrednjavajuciFilter`" koja kao prvi parametar prima matricu koja se filtrira, organiziranu kao vektor vektôrā realnih brojeva, a kao drugi parametar red filtriranja N (tipa "`int`"). Ukoliko je N negativan, treba baciti izuzetak tipa "`domain_error`" uz prateći tekst "Neispravan red filtriranja". Kao rezultat funkcija treba da vrati novu matricu dobijenu kao rezultat filtriranja. Matrica koja se filtrira uopće ne mora imati ispravnu strukturu matrice, odnosno može biti i "grbava". Pri tome se, naravno, u račun uzimaju u obzir samo pikseli koji stvarno postoje unutar "prozora", tj. pikseli kojima odgovaraju legalne koordinate unutar raspona matrice. U svakom slučaju, rezultirajuća matrica imaće isti oblik kao i ulazna matrica.

Napisanu funkciju trebate demonstrirati u testnom programu u kojem se prvo unose dimenzije matrice, zatim i njeni elementi, te željeni red filtriranja, Nakon toga, program treba da ispiše matricu dobijenu kao rezultat filtriranja. Elemente ispišite koristeći prikaz na tačno dvije decimale, pri čemu ćete predvidjeti širinu od ukupno 7 karaktera za ispis svakog elementa. Slijede primjeri dijaloga između korisnika i programa:

```
Unesite broj redova i kolona matrice: 5 6
Unesite elemente matrice:
2 3 3 1 2 3
1 2 4 3 4 4
2 3 4 5 3 2
3 4 3 4 4 2
1 2 3 3 4 3
Unesite red filtriranja: 1

Matrica nakon filtriranja:
   2.00   2.50   2.67   2.83   2.83   3.25
   2.17   2.67   3.11   3.22   3.00   3.00
   2.50   2.89   3.56   3.78   3.44   3.17
   2.50   2.78   3.44   3.67   3.33   3.00
   2.50   2.67   3.17   3.50   3.33   3.25
```

```
Unesite broj redova i kolona matrice: 2 2
Unesite elemente matrice: 1 2 3 4
Unesite red filtriranja: -1

GRESKA: Neispravan red filtriranja!
```

NAPOMENA: U ovom primjeru testnog programa, filtriranje se uvijek vrši nad matricama koje imaju ispravnu strukturu matrice. Međutim, funkcija za filtriranje će biti testirana i s matricama koje nisu takve, tj. s grbavim matricama.

3. Pod *spiralnom matricom* formata $M \times N$ podrazumijeva se matrica koja ima $M \cdot N$ elemenata, u kojoj se javljaju svi cijeli brojevi od k do $k + M \cdot N - 1$ (gdje je k neki cijeli broj) i to tako da se u gornjem lijevom uglu nalazi broj k, a zatim se ostali brojevi redaju u rastućem redoslijedu, ali tako da tvore oblik spirale, kao što će biti vidljivo u primjeru. Pri tome, spirala se može "razmotavati" u smjeru kazaljke na satu (tzv. *desna spiralna matrica*), ili u smjeru suprotnom od kazaljke na satu (tzv. *lijeva spiralna matrica*). Ispod je prikazan primjer lijeve (primjer lijevo) i desne (primjer desno) spiralne matrice formata 5×7:

$$\begin{pmatrix} 5 & 24 & 23 & 22 & 21 & 20 & 19 \\ 6 & 25 & 36 & 35 & 34 & 33 & 18 \\ 7 & 26 & 37 & 38 & 39 & 32 & 17 \\ 8 & 27 & 28 & 29 & 30 & 31 & 16 \\ 9 & 10 & 11 & 12 & 13 & 14 & 15 \end{pmatrix} \qquad \begin{pmatrix} -3 & -2 & -1 & 0 & 1 & 2 & 3 \\ 16 & 17 & 18 & 19 & 20 & 21 & 4 \\ 15 & 28 & 29 & 30 & 31 & 22 & 5 \\ 14 & 27 & 26 & 25 & 24 & 23 & 6 \\ 13 & 12 & 11 & 10 & 9 & 8 & 7 \end{pmatrix}$$

Napišite funkciju "`KreirajSpiralnuMatricu`" koja kao prva dva parametra prima broj redova i kolona matrice, treći parametar je početna vrijednost k (odnosno vrijednost koja će biti u gornjem lijevom uglu matrice), dok je četvrti parametrar logičkog tipa koji određuje da li će se kreirati lijeva ili desna spiralna matrica (desna se kreira ukoliko ovaj parametar ima vrijednost "`true`"). Funkcija kao rezultat treba da vrati odgovarajuću spiralnu matricu, organiziranu kao vektor vektôrā cijelih brojeva (tipa "`int`"). U slučaju da je zadani broj redova ili kolona negativan ili 0, funkcija treba da kao rezultat vrati praznu matricu (tj. matricu formata 0×0).

Pored ove funkcije, trebate napisati i funkciju "DaLiJeSpiralnaMatrica". Ova funkcija kao jedini parametar prima matricu organiziranu kao vektor vektôrā cijelih brojeva. Funkcija treba da kao rezultat vrati logičku vrijednost "**true**" ako i samo ako je ponuđena matrica spiralna matrica (bilo lijeva, bilo desna), a logičku vrijednost "**false**" u suprotnom (što uključuje i slučaj kada se funkciji ponudi grbava matrica).

Napisanu funkciju "KreirajSpiralnuMatricu" trebate demonstrirati u testnom programu koji traži od korisnika dimenzije matrice, početnu vrijednost *k*, te informaciju da li se kreira lijeva ili desna spiralna matrica a koji zatim kreira i ispisuje na ekran kreiranu spiralnu matricu. Širinu ispisa prilagodite tako da bude za 1 duža od širine najšireg elementa u matrici. Dijalog između korisnika i programa treba da izgleda poput sljedećeg:

```
Unesite broj redova i kolona matrice: 5 7
Unesite pocetnu vrijednost: -3
Unesite L za lijevu, a D za desnu spiralnu matricu: D

Kreirana desna spiralna matrica:
 -3 -2 -1  0  1  2  3
 16 17 18 19 20 21  4
 15 28 29 30 31 22  5
 14 27 26 25 24 23  6
 13 12 11 10  9  8  7
```

NAPOMENA: Funkciju "DaLiJeSpiralnaMatrica" ne trebate testirati u Vašem testnom glavnom programu, ali to ne znači da njeno funkcioniranje neće biti testirano putem autotestova!

4. U novinarstvu se često koristi *cenzurisanje teksta*, u kojem se dijelovi teksta koji su iz bilo kojeg razloga neprikladni sakrivaju od čitatelja tako što se precrtaju, zamijene nekim posebnim znacima poput zvjezdica, ili na neki drugi način (mada često smisao sakrivenih dijelova teksta može biti jasan iz ostatka teksta, odnosno konteksta). Mogu se cenzurisati individualne riječi, ali i kompletne rečenice. Obično se cenzurišu dijelovi teksta koji su nepristojni, sadrže uvrede, podstiču na nasilje ili kriminalne radnje i slično, mada se u represivnim režimima često cenzurišu i dijelovi teksta koji su u principu korektni, ali sadrže političke poruke koje vlastodršcima ne idu u prilog. Kao primjer cenzurisanja, možemo uzeti rečenicu "On je potpuni idiot, stoga ga treba dobro namlatiti!". U ovoj rečenici mogla bi se cenzurisati riječ "idiot" (izrečena u smislu uvrede, jer ovdje se očito ne misli na medicinsku dijagnozu, niti na naslov romana Dostojevskog) i "namlatiti" (podsticanje na nasilje). Cenzurisana rečenica mogla bi glasiti recimo "On je potpuni *****, stoga ga treba dobro *********!", u kojoj su sporne riječi zamijenjene nizom zvjezdica.

   Vaš zadatak je da napravite funkciju "Cenzura", koja obavlja cenzurisanje rečenice koja je zadana kao prvi parametar funkcije (tipa "string"). Drugi parametar je vektor koji sadrži spisak riječi koje nisu dozvoljene. Svaki element vektora predstavlja po jednu riječ (riječi su također tipa "string"). Funkcija treba da svaku pojavu neke od zabranjenih riječi u rečenici zamijeni nizom zvjezdica koji je dug onoliko koliko je duga i sama riječ i da vrati kao rezultat tako transformiranu rečenicu. Stoga, ukoliko se izvrši naredba poput

```
std::cout << Cenzura("On je potpuni idiot, stoga ga treba dobro namlatiti!",
  {"kreten", "idiot", "ubiti", "namlatiti", "crncuga"});
```

   na ekranu se treba ispisati tekst

```
On je potpuni *****, stoga ga treba dobro *********!
```

   Treba naglasiti da se pod pojmom riječ podrazumijeva svaki niz znakova koji se sastoji od slova i/ili cifri, a koji je s obje strane omeđen znakom koji nije ni slovo ni cifra (nego recimo razmak ili znak interpunkcije), odnosno početkom ili krajem rečenice (ukoliko nema znakova ispred ili iza riječi). Dakle, riječi u sebi ne mogu sadržavati ništa što nije slovo ili cifra. Ukoliko neka od ponuđenih "riječi" unutar vektora zabranjenih riječi nije ispravna (u smislu da sadrži bilo šta što nije slovo ili cifra), funkcija treba baciti izuzetak tipa "logic_error" uz prateći tekst "Neispravna riječ". Prilikom upoređivanja riječi, treba ignorirati razliku između malih i velikih slova. Na primjer, ukoliko je zabranjena riječ zadana kao "idiot", treba prepoznati tu zabranjenu riječ bez obzira da li je ona napisana kao "idiot", "Idiot" ili "IDIOT". Također, prepoznaju se samo kompletne

riječi, ali ne i njihovi dijelovi. Recimo, uz istu pretpostavku da je "idiot" zabranjena riječ, pojavu riječi "idiotski" u rečenici ne treba cenzurisati (tj. ne treba je zamijeniti sa "idiot***", nego je treba ostaviti neizmijenjenom).

Napisanu funkciju treba demonstrirati u kratkom testnom programu, u kojem će se dijalog između programa i korisnika odvijati na način koji je jasan iz dolje prikazanih slika (unos zabranjenih riječi uvijek se završava tačkom):

```
Unesite recenicu: On je potpuni idiot, stoga ga treba dobro namlatiti!
Unesite zabranjene rijeci (. za kraj): kreten idiot ubiti namlatiti crncuga .

Cenzurisana recenica: On je potpuni *****, stoga ga treba dobro *********!
```

```
Unesite recenicu: I corava koka gvozdena vrata otvara...
Unesite zabranjene rijeci (. za kraj): xyz 2+3=5 ##$-- .

GRESKA: Nelegalne zabranjene rijeci!
```