Zadaća 3.

Ova zadaća nosi ukupno 4.5 poena, pri čemu zadaci nose redom 0.7, 0.9, 0.7, 0.8, 1 i 0.4 poena. Svi zadaci se mogu uraditi na osnovu gradiva sa prvih 8 predavanja i pretpostavljenog predznanja iz predmeta "Osnove računarstva". Rok za predaju ove zadaće je nedjelja, 21. V 2023. (do kraja dana).

1. U raznim oblastima nauke i tehnike često se dešava da je vrijednost neke funkcije f poznata samo na nekom konačnom skupu tačaka x_i, $i = 1, 2, \ldots, n$ (tj. poznate su samo vrijednosti $y_i = f(x_i)$, $i = 1, 2, \ldots, n$), a zanima nas njena vrijednost $f(x)$ u nekoj tački x koja *ne pripada ovom skupu*. Ovaj problem poznat je kao *problem interpolacije*. Jedan od najjednostavnijih načina za rješavanje ovog problema je da se pretpostavi da je grafik funkcije izlomljena linija koja se dobija spajanjem dužima tačaka (x_i, y_i), $i = 1, 2, \ldots, n$ (tzv. *čvorova interpolacije*) sortiranih u rastući poredak po x koordinati pravim linijama. Na taj način se dobija tzv. *linearna interpolacija*. Međutim, linearna interpolacija ima prilično malu tačnost i daje neprirodno izlomljen grafik. Bolje bi bilo kada bi grafik funkcije bio "gladak" i da pri tom ne pravi nikakva neprirodna "izvijanja" ili oscilacije u dijelovima između čvorova interpolacije. Ovim problemom bavili su se brojni poznati naučnici, uključujući *Lagrangea*, *Newtona* i *Gaussa*, i pri tome postigli relativno zadovoljavajuće rezultate. mada ponekad praćene priličnim problemima. Jedna jednostavna tehnika, koja doduše daje "gladak" grafik, ali koji ponekad nažalost ima vrlo neprirodne "oscilacije" u dijelovima između čvorova interpolacije (ta pojava se naziva *Rungeov fenomen*, po njemačkom matematičaru *Carlu Rungeu* koji ju je uočio i otkrio pod kojim uvjetima do nje dolazi), zasniva se na ideji da se konstruira polinom $\mathrm{P}(x)$ stepena $n-1$ koji prolazi kroz sve tačke (x_i, y_i), $i = 1, 2, \ldots, n$, a da se nakon toga umjesto nepoznate vrijednosti $f(x)$ koristi vrijednost $\mathrm{P}(x)$, odnosno uzima se da je $f(x) \approx \mathrm{P}(x)$. Ovo je tzv. *polinomska interpolacija*. Postoje razni načini da se odredi ovaj polinom. Najdirektniji je da se pretpostavi da je $\mathrm{P}(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$, pri čemu su $a_0, a_1, \ldots, a_{n-1}$ neki nepoznati koeficijenti, koji se zatim odrede rješavajući sistem n linearnih jednačina sa n nepoznatih oblika $\mathrm{P}(x_i) = y_i$ (odnosno $a_0 + a_1 x_i + a_2 {x_i}^2 + \ldots + a_{n-1} {x_i}^{n-1} = y_i$) za sve $i = 1, 2, \ldots, n$. Međutim, ovo je dosta naporan i neefikasan postupak. Srećom, postoji način da se vrijednost $\mathrm{P}(x)$ odredi, bez ikakve potrebe da se eksplicitno nađu koeficijenti polinoma $\mathrm{P}(x)$. Naime, za tu svrhu može se koristiti tzv. *Lagrangeova formula* (nazvana po *J. L. Lagrangeu*, koji ju je objavio 1795. godine, mada je za nju znao *E. Waring* 16 godina prije Lagrangea), koja glasi ovako:

$$f(x) \approx \mathrm{P}(x) = \sum_{i=1}^{n} \left(y_i \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \right)$$

Polinomska interpolacija zasnovana na ovoj formuli naziva se *Lagrangeova interpolacija*. Vaš zadatak je da napravite funkciju za podršku za Lagrangeovu interpolaciju. Ova funkcija se treba zvati "`LagrangeovaInterpolacija`", a imaće dvije verzije. Prva verzija ima jedan parametar, koji je vektor uređenih parova (tj. objekata tipa "`std::pair`") realnih brojeva, koji predstavljaju čvorove interpolacije. Kao rezultat, ova funkcija treba vratiti novu funkciju sa jednim realnim argumentom x koja obavlja traženu Lagrangeovu interpolaciju za zadanu vrijednost x (to je zapravo polinom P iz prethodnog teorijskog razmatranja). Na primjer, ukoliko se izvrši naredba

```
auto P = LagrangeovaInterpolacija({{1, 3}, {2, 5}, {4, 4}, {5, 2}, {7, 1}}));
```

tada će "`P`" biti funkcija (tačnije, polinom) dobijen Lagrangeovom interpolacijom na osnovu čvorova $(1, 3)$, $(2, 5)$, $(4, 4)$, $(5, 2)$ i $(7, 1)$. Želimo li sada odrediti vrijednost dobijenu interpolacijom u tački $x = 2.5$ (usput, ta vrijednost iznosi $\mathrm{P}(2.5) = 5.33594$), možemo izvršiti recimo naredbu

```
std::cout << P(2.5);
```

U slučaju da među čvorovima interpolacije postoje čvorovi koji imaju identične x-koordinate (što nije dozvoljeno), funkcija "`LagrangeovaInterpolacija`" treba baciti izuzetak tipa "`domain_error`" uz prateći tekst "Neispravni cvorovi". Interesantno je uočiti, da za razliku od mnogih drugih formula za interpolaciju, ova formula ima smisla čak i ako je $x < x_{min}$ ili $x > x_{max}$, gdje su x_{min} i x_{max} najmanja odnosno najveća vrijednost među vrijednostima x_i, $i = 1, 2, \ldots, n$ (inače, u tom slučaju, govorimo o *ekstrapolaciji*, a ne o interpolaciji, koja je mnogo manje pouzdana od interpolacije).

Druga verzija funkcije "`LagrangeovaInterpolacija`", s četiri parametra, služi za aproksimaciju neke već postojeće funkcije uz pomoć Lagrangeove interpolacije. Prvi parametar ove funkcije je neka realna funkcija jednog realnog argumenta (ili bilo šta što se može pozvati s jednim realnin argumentom i daje rezultat koji se može interpretirati kao realan broj) i on predstavlja funkciju koju aproksimiramo. Drugi, treći i četvrti parametar ćemo nazvati redom x_{min}, x_{max} i Δx. Ova funkcija treba prvo da kreira čvorove interpolacije na intervalu $[x_{min}, x_{max}]$ sa korakom Δx (tj. u tačkama x_{min}, $x_{min} + \Delta x$, $x_{min} + 2\,\Delta x$, itd.) uzimajući u tim tačkama vrijednosti funkcije zadane prvim parametrom, a zatim treba da iskoristi te čvorove da konstruira novu funkciju dobijenu iz tih čvorova Lagrangeovom interpolacijom i da vrati tako konstruisanu funkciju kao rezultat (vodite računa da ukoliko je $x_{max} - x_{min}$ tačno djeljivo sa Δx, tačka x_{max} također treba uključena u čvorove interpolacije, što se možda neće desiti ukoliko budete nepažljivi s realnom aritmetikom). U slučaju da je $x_{min} > x_{max}$ ili da je $\Delta x \leq 0$, treba baciti izuzetak tipa "`domain_error`" sa pratećim tekstom "Nekorektni parametri".

Napišite i kratki testni program ("`main`" funkciju) u kojem ćete demonstrirati napisane funkcije, pri čemu ćete drugu funkciju demonstrirati na problemu aproksimacije jedne konkretne funkcije zadane kao $f(x) = x^2 + \sin x + \ln(x + 1)$. Slijede primjeri kako trebaju izgledati dijalozi između programa i korisnika (objašnjenja će uslijediti nakon prikaza dijaloga):

```
Odaberite opciju (1 - unos cvorova, 2 - aproksimacija): 1
Unesite broj cvorova: 5
Unesite cvorove kao parove x y: 1 3
2 5
4 4
5 2
7 1
Unesite argument (ili "kraj" za kraj): 2.5
f(2.5) = 5.33594
Unesite argument (ili "kraj" za kraj): 7
f(1) = 1
Unesite argument (ili "kraj" za kraj): 1.5
f(1.5) = 4.21788
Unesite argument (ili "kraj" za kraj): 0.3
f(0.3) = 0.643802 [ekstrapolacija]
Unesite argument (ili "kraj" za kraj): 5.32
f(5.32) = 1.38346
Unesite argument (ili "kraj" za kraj): kraj
```

```
Odaberite opciju (1 - unos cvorova, 2 - aproksimacija): 2
Unesite krajeve intervala i korak: 0 1 0.2
Unesite argument (ili "kraj" za kraj): 0.4
f(0.4) = 0.885891 P(0.4) = 0.883891
Unesite argument (ili "kraj" za kraj): 0.13
f(0.13) = 0.268752 P(0.13) = 0.268737
Unesite argument (ili "kraj" za kraj): 0.91
f(0.91) = 2.26471 P(0.91) = 2.26469
Unesite argument (ili "kraj" za kraj): 1.5
f(1.5) = 4.16379 P(1.5) = 4.17145 [ekstrapolacija]
Unesite argument (ili "kraj" za kraj): kraj
```

Kao što je vidljivo, nakon nekoliko početnih pitanja, ulazi se u petlju u kojoj se stalno traži vrijednost argumenta x i izračunava vrijednost dobijena interpolacijom (ili ekstrapolacijom) za taj argument (u drugom dijalogu prikazuje se i tačna vrijednost funkcije i aproksimativna vrijednost dobijena interpolacijom odnosno ekstraploacijom). U slučaju da je u pitanju bila ekstrapolacija, ta činjenica se prikazuje korisniku (u formi odgovarajućeg teksta u uglastim zagradama). Petlja se prekida čim se unese nešto što nije broj (to ne mora nužno biti riječ "kraj" kako je gore prikazano), a time se ujedno završava i program.

Možete pretpostaviti da će pri testiranju na svim drugim mjestima gdje se očekuju brojevi zaista biti uneseni brojevi, tako da ne morate provjeravati validnost unosa. Ukoliko se zadaju takvi početni podaci da funkcija "`LagrangeovaInterpolacija`" baci izuzetak, treba ispisati odgovarajući tekst izuzetka i odmah prekinuti program.

2. U većini programa za obradu teksta postoji opcija za kreiranje *indeksa pojmova*, koji predstavlja popis pojmova koji se nalaze u tekstu, pri čemu se uz svaki pojam navode i pozicije na kojima se taj pojam javlja u tekstu. U ovom zadatku ćete trebati uraditi nešto slično tome. Tekst koji se analizira nalazi se u objektu tipa "`Knjiga`", pri čemu je ovaj tip definiran pomoću deklaracije

   ```
   typedef std::map<std::string, std::vector<std::string>> Knjiga;
   ```

   Dakle, "`Knjiga`" je mapa čija su ključna polja tipa "`string`", a pridružene vrijednosti vektori stringova. Ključna polja predstavljaju oznake poglavlja knjige (oni su stringovnog tipa, jer oznake poglavlja ne moraju nužno biti brojčane). Odgovarajuće pridružene vrijednosti su vektori stringova, pri čemu svaki string u vektoru predstavlja sadržaj jedne stranice razmatranog poglavlja knjige. Na primjer, neka je "`k`" neki objekat tipa "`Knjiga`". Tada je "`k["XIV"][5]`" sadržaj šeste stranice XIV-og poglavlja knjige (šeste a ne pete, jer indeksacija počinje od nule).

   Prvo ćete napraviti funkciju "`KreirajIndeksPojmova`" koja kao parametar prima neki objekat tipa "`Knjiga`", koji predstavlja tekst koji se analizira. Funkcija kao rezultat treba vratiti mapu koja predstavlja traženi indeks pojmova. Ključna polja ove mape su tipa "`string`", a pridružene vrijednosti su skupovi čiji su elementi uređene trojke (tipa "`tuple`"), čija je prva koordinata tipa "`string`", a druga i treća su cijeli brojevi. Vrijednosti ključnih polja predstavljaju različite riječi pronađene u analiziranom tekstu, dok su odgovarajuće pridružene vrijednosti skupovi čiji elementi opisuju pozicije na kojima se odgovarajuća riječ nalazi unutar razmatranog teksta. Svaka pozicija opisana je kao uređena trojka, pri čemu prva koordinata predstavlja oznaku poglavlja, druga koordinata redni broj stranice (uz numeraciju koja počinje od jedinice, a ne od nule), dok treća koordinata predstavlja poziciju razmatrane riječi unutar stranice (tj. indeks od kojeg počinje riječ). Na primjer, pretpostavimo radi jednostavnosti da tekst ima samo jedno poglavlje koje se zove "I", u kojem ima samo jedna stranica (koja naravno ima indeks 0), čiji je sadržaj string "abc qwe stsda abc abc dhi qwe hrkw dhi". Funkcija tada treba da vrati mapu u kojoj se nalazi 5 parova (toliko ukupno ima različitih riječi u tekstu), čija su ključna polja stringovi "abc", "dhi", "hrkw", "qwe" i "stsda" (riječi koje se nalaze u tekstu), dok su odgovarajuće pridružene vrijednosti skupovi {("I", 1, 0), ("I", 1, 14), ("I", 1, 18)}, {("I", 1, 22), ("I", 1, 35)}, {("I", 1, 30)}, {("I", 1, 4), ("I", 1, 26)} i {("I", 1, 8)}. Jasno je zbog čega su u svim trojkama prve dvije koordinate "I" odnosno 1, a što se tiče treće koordinate, riječ "abc" se nalazi na pozicijama 0, 14 i 18 u razmatranom stringu, riječ "dhi" nalazi se na pozicijama 22 i 35, itd.

   Sljedeća funkcija koju treba napraviti je "`PretraziIndeksPojmova`". Ova funkcija kao parametar prima neku riječ (tipa "`string`") i mapu koja predstavlja indeks pojmova, a koja kao rezultat vraća odgovarajući skup pozicija za datu riječ pronađen u datom indeksu pojmova, ili baca izuzetak tipa "`logic_error`" sa propratnim tekstom "Pojam nije nadjen" u slučaju da data riječ nije nađena u indeksu pojmova.

   Konačno, posljednja funkcija koju treba napraviti je "`IspisiIndeksPojmova`". Ova funkcija kao parametar prima mapu koja predstavlja indeks pojmova, a ispisuje njen kompletan sadržaj na ekranu u obliku tako da se u svakom redu ispisuje prvo pojam, zatim dvotačka praćena razmakom, i na kraju, spisak pozicija međusobno razdvojenih zarezom iza kojeg slijedi razmak. Svaka pozicija ispisuje se kao oznaka poglavlja, broj stranice i pozicije unutar stranice, pri čemu se između tih podataka nalazi kosa crta (bez razmaka ispred i iza nje). Na primjer, za indeks pojmova kreiran na osnovu teksta iz prethodnog primjera, ova funkcija bi trebala proizvesti ispis poput sljedećeg:

   ```
   abc: I/1/0, I/1/14, I/1/18
   dhi: I/1/22, I/1/35
   hrkw: I/1/30
   qwe: I/1/4, I/1/26
   stsda: I/1/8
   ```

   Stringovi koji predstavljaju sadržaje stranica mogu sadržavati ma kakve znakove (slova, cifre i znake interpunkcije), pri čemu se ne pravi razlika između malih i velikih slova, tako da "ABC", "Abc", "abc", "aBc" i "aBC" predstavljaju istu riječ. Pri tome se u mapi uvijek bilježi riječ koja se sastoji samo od malih slova (tako da će sve ove riječi zapravo biti evidentirane kao riječ "abc"). Kao riječ unutar teksta smatra se svaka neprekinuta sekvenca znakova koji su slova ili cifre koja je sa obe strane omeđena razmakom ili znakom interpunkcije (tačnije, znakom koji nije niti slovo niti cifra), osim eventualno na početku ili kraju stringa, kad ne mora biti razmaka ili znaka

interpunkcije ispred odnosno iza riječi. Na primjer, u stringu "pqr, ab/123 (qwe) tt2 " riječi su "pqr", "ab", "123", "qwe" i "tt2".

Napisane funkcije demonstrirajte u glavnom programu u kojem se prvo sa tastature unosi tekst za analizu (na način koji će biti vidljiv iz primjera koji slijedi), nakon čega se na ekranu ispisuje kreirani indeks pojmova. Potom program ulazi u petlju u kojoj se za svaku riječ unesenu sa tastature ispisuje pozicije na kojima se riječ nalazi u analiziranom tekstu, pri čemu su pozicije međusobno razdvojene razmacima (koristeći pri tome kreirani indeks pojmova), ili informaciju da unesena riječ nije nađena (u vidu teksta "Unesena rijec nije nadjena!". Program treba da prekine rad kada korisnik unese znak "." (tačka sa tastature). Također, isti znak se koristi i kao indikator da ne želimo unositi novo poglavlje ili novu stranicu. Dijalog između korisnika i programa trebao bi izgledati poput sljedećeg:

```
Unesite naziv poglavlja: I
Unesite sadržaj stranice 1: abc qwe stsda abc abc dhi qwe hrkw dhi
Unesite sadržaj stranice 2: .
Unesite naziv poglavlja: .

Kreirani indeks pojmova:
abc: I/1/0, I/1/14, I/1/18
dhi: I/1/22, I/1/35
hrkw: I/1/30
qwe: I/1/4, I/1/26
stsda: I/1/8

Unesite rijec: abc
I/1/0 I/1/14 I/1/18
Unesite rijec: hrkw
I/1/30
Unesite rijec: xyzzy
Unesena rijec nije nadjena!
Unesite rijec: .
```

3. Pod matričnim harmonijskim polinomom n-tog reda neke matrice $\boldsymbol{A}$ podrazumijeva se matrica definirana izrazom

$$\mathcal{H}_n \boldsymbol{A} = \boldsymbol{A} + \frac{1}{2}\boldsymbol{A}^2 + \frac{1}{3}\boldsymbol{A}^3 + \ldots + \frac{1}{n}\boldsymbol{A}^n = \sum_{k=1}^{n} \frac{1}{k}\boldsymbol{A}^k$$

gdje je $\boldsymbol{A}^k$ klasični k-ti stepen matrice, tj. produkt matrice $\boldsymbol{A}$ k puta sa samom sobom (razumije se da zbog toga matrica $\boldsymbol{A}$ mora biti kvadratna matrica). Dopunite program za rad sa generičkom strukturom "`Matrica`" dat na Predavanju 8_b sa dvije nove funkcije nazvane "`ProduktMatrica`" i "`MatricniHarmonijskiPolinom`". Funkcija "`ProduktMatrica`" prima dvije matrice kao parametre (matrice su definirane kao odgovarajuće strukture) i vraća njihov produkt kao rezultat, odnosno baca izuzetak ukoliko se matrice ne mogu množiti. Funkcija "`MatricniHarmonijskiPolinom`" prima matricu $\boldsymbol{A}$ kao prvi parametar, a kao drugi parametar prirodan broj n, dok kao rezultat vraća vrijednost izraza $\mathcal{H}_n \boldsymbol{A}$ (za potrebe realizacije ove funkcije, trebate koristiti već napisanu funkciju "`ProduktMatrica`"). U slučaju da matrica $\boldsymbol{A}$ nije kvadratna, funkcija treba baciti izuzetak tipa "`domain_error`" uz prateći tekst "Matrica nije kvadratna", s obzirom da je izraz za $\mathcal{H}_n \boldsymbol{A}$ definiran samo za kvadratne matrice. Izuzetak (također tipa tipa "`domain_error`", samo uz prateći tekst "Pogresan parametar n") treba baciti i ukoliko je n negativan.

Pored ove dvije funkcije, treba također proširiti funkciju "`IspisiMatricu`" sa dodatna dva parametra nazvana "`preciznost`" i "`treba_brisati`". Parametar "`preciznost`" je cijeli broj koji određuje preciznost ispisa, odnosno broj tačnih cifara pri ispisu (njega zapravo treba proslijediti funkciji "`cout.precision`" ili manipulatoru "`setprecision`"). Ovaj parametar treba da ima podrazumijevanu vrijednost 6. Drugi parametar "`treba_brisati`" je tipa "**`bool`**". Ukoliko ovaj parametar ima vrijednost "**`true`**", funkcija treba da po obavljenom ispisu oslobodi prostor zauzet matricom koja joj je proslijeđena kao parametar, u suprotnom ne treba da radi ništa po tom pitanju. Ovim se omogućava da možemo zadavati pozive poput

```
IspisiMatricu(ZbirMatrica(a, b), 10, 5, true);
```

tako da se oslobađanje memorije koju je zauzela pomoćna matrica koja predstavlja zbir matrica može obaviti bez korištenja pomoćne promjenljive (kao što smo morali u izvornom programu prikazanom na predavanjima). Pri tome, definirajte da parametar "`treba_brisati`" ima podrazumijevanu vrijednost "`false`", tako da ga ne treba navoditi ukoliko nam brisanje ne treba.

Napisane funkcije testirajte u glavnom programu koji sa tastature prvo traži da se unese dimenzija matrice (dovoljno je unijeti samo jednu dimenziju, s obzirom da će matrica uvijek biti kvadratna), zatim elementi matrice (pozivom funkcije "`UnesiMatricu`", zadajući "A" kao ime matrice), te red polinoma *n*. Nakon obavljenog unosa, program treba da ispiše matricu dobijenu izračunavanjem matričnog harmonijskog polinoma koristeći širinu ispisa od 10 mjesta i preciznošću od 6 tačnih cifara. Dijalog između korisnika i programa treba da izgleda poput sljedećeg:

```
Unesite dimenziju kvadratne matrice: 2
Unesite elemente matrice A:
A(1,1) = 1
A(1,2) = 2
A(2,1) = 3
A(2,2) = 4
Unesite red polinoma: 6
Matricni harmonijski polinom:
   1237.55    1804.1
   2706.15    3943.7
```

U slučaju da dođe do bacanja izuzetka, samo ispišite odgovarajući tekst izuzetka i prekinite program. Također, dobro pazite da nigdje ne dođe do curenja memorije, ni pod kakvim okolnostima (u suštini, to je i *osnovni cilj zadatka*).

Napomena: U ovom programu je *izuzetno lako* napraviti curenje memorije (konkretnije, u funkciji "`MatricniHarmonijskiPolinom`"). Dobro razmislite šta radite, i ne oslanjate se nasumice ni na kakve "testere curenja memorije" i slična pomagala, jer u suprotnom nećete ovladati tehnikama upravljanja memorijom (to i jeste glavni cilj ovog zadatka). Kad shvatite koliko treba razumijevanja da se u ovom zadatku izbjegne curenje memorije, tek tada ćete *zaista cijeniti destruktore*, koji ovakve probleme rješavaju praktično automatski.

Napomena 2: Zadaci slični ovom zadatku su se javljali kao zadaci za zadaću u prethodnim generacijama. Naravno, možete "pođoniti" neki od tih zadataka (uz odgovarajuće modifikacije), ili Vam ga može uraditi neko drugi, za pare ili ne (kao što uostalom nekima od Vas drugi rade i ostale zadatke). Ali da znate, što bi rekao Balašević, *neko to od gore vidi sve*, i prije ili kasnije to će Vam se od glavu razbiti. Izuzetno je važno da barem ovaj zadatak *zaista uradite sami* (naravno, samostalno iste trebali uraditi i sve ostale zadatke, ali je izuzetno važno da baš ovaj zadatak ne prepišete ni po koju cijenu). Ukoliko ne želite samostalno da uradite ovaj zadatak, *radije ga nemojte ni raditi, niti predavati*.

4. U plemenu Wabambe poglavica se bira razbrajalicom. Svi punoljetni članovi plemena osim bivšeg poglavice poredaju se u krug, a bivši poglavica saopštava plemenu neki prirodni broj M. Nakon toga, počinje razbrajanje. Razbrajanje se vrši tako što se svaki M-ti član plemena po redu odstranjuje iz kruga, dok u krugu ne ostane samo jedan čovjek. Taj čovjek će postati novi poglavica. Na primjer, ukoliko u krugu ima 11 ljudi (sa rednim brojevima $1 - 11$) i ukoliko je $M = 4$, redoslijed ispadanja je 4, 8, 1, 6, 11, 7, 3, 2, 5 i 10 (nacrtajte sliku), tako da na kraju ostaje samo čovjek sa rednim brojem 9, koji će postati novi poglavica.

   Mandat poglavice traje tačno godinu dana. Za vrijeme trajanja mandata poglavica ima apsolutnu moć u plemenu i uživa sve privilegije koje se mogu zamisliti. Po isteku mandata, poglavicu njegovi saplemenici skuhaju i pojedu. Svi članovi plemena doživljavaju izbor za poglavicu kao veliku čast. Međutim, Mbebe Mgogo, koji upravo ove godine postaje punoljetan, nikako ne želi postati poglavica. Ideja o neograničenim privilegijama zapravo i ne zvuči loše, ali mu se ideja da bude skuhan i pojeden nakon isteka mandata baš i ne sviđa osobito. Zbog toga, Mbebe Mgogo po svaku cijenu želi da izbjegne da bude izabran za poglavicu. On je uspio podmititi postojećeg poglavicu da mu oda koji će broj M biti saopćen. Međutim, Mbebe ne zna koliki je tačan broj punoljetnih osoba u plemenu, ali zna da taj broj nije manji od N_1 niti veći od N_2 gdje su N_1 i N_2 neki prirodni brojevi.

Potrebno je napraviti program koji će Mbebeu pomoći da ne postane poglavica. U programu treba implementirati dvije funkcije, "`Poglavica`" i "`SigurnoMjesto`". Funkcija "`Poglavica`" prima kao parametre broj punoljetnih članova plemena N i korak razbrajanja M i kao rezultat daje redni broj čovjeka koji postaje poglavica. Funkcija treba biti zasnovana na tipu "`list`", koji se često primjenjuje upravo za rješavanje problema srodnih opisanom problemu. Naime, ova funkcija će prvo napuniti listu rednim brojevima od 1 do N. Nakon što se formira lista, vrši se kretanje kroz nju, pri čemu se nakon svakih M napravljenih koraka iz liste odstranjuje onaj element na kojem se trenutno nalazimo (čime se odgovarajuća osoba efektivno odstranjuje iz kruga), nakon čega se prelazi na sljedeći element. Pri tome, kad god se dostigne kraj liste, vraćamo se ponovo na početak ("kružna" lista). Postupak se ponavlja dok u listi ne ostane samo jedan broj, koji upravo predstavlja redni broj novog poglavice (i njega treba vratiti kao rezultat iz funkcije). Da bi se olakšalo testiranje, funkcija "`Poglavica`" treba da ima i treći parametar logičkog tipa sa podrazumijevanom vrijednošću "**`false`**", za potrebe testiranja. Ukoliko je njegova vrijednost "**`true`**", na ekranu se ispisuje redni broj svake odstranjene osobe u redoslijedu odstranjivanja, u suprotnom se ne vrši nikakav ispis. Pri tome, brojevi trebaju biti međusobno razdvojeni zarezom.

Funkcija "`SigurnoMjesto`" kao parametre prima M, N_1 i N_2 a kao rezultat daje redni broj sigurnog mjesta u krugu, tj. mjesta na koje treba stati da se sigurno ne bi postalo poglavica za ma kakav N u opsegu od N_1 do N_2 (ukoliko takvih mjesta ima više, funkcija vraća redni broj onog sigurnog mjesta sa najmanjim rednim brojem). Ukoliko sigurno mjesto za zadane vrijednosti M, N_1 i N_2 ne postoji, funkcija kao rezultat vraća 0. Rad ove funkcije zasniva se na pozivanju funkcije "`Poglavica`" za sve vrijednosti N u opsegu od N_1 do N_2 i bilježenja ko će biti izabran za poglavicu u svim tim slučajevima (mjesto je sigurno ukoliko čovjek koji stoji na tom mjestu neće biti poglavica ni za kakvo N u opsegu od N_1 do N_2). Napisane funkcije demonstrirajte u testnom programu koji za zadane vrijednosti M, N_1 i N_2 ispisuje na koju poziciju Mbebe Mgogo treba da stane da garantirano neće postati poglavica, ili informaciju da takva garancija ne postoji za zadane ulazne podatke.

5. Riješite ponovo prethodni zadatak, ali tako što ćete za realizaciju funkckije "`Poglavica`" umjesto bibliotečki definiranog tipa podataka "`list`" koristiti ručno kreiranu povezanu listu čvorova, bez upotrebe ikakvih bibliotečki definiranih tipova. To ćete izvesti ovako. Prvo ćete definirati čvornu strukturu "`Clan`" koja predstavlja jedan element liste (tj. jednog člana plemena) sa poljima "`redni_broj`", koje sadrži redni broj člana plemena u krugu (tipa "**`int`**"), i "`sljedeci`", koje je pokazivač koji pokazuje na sljedeću osobu u krugu. Funkcija "`Poglavica`" zatim treba kreirati povezanu listu ljudi (tj. čvorova tipa "`Clan`") sa rednim brojevima redom od 1 do N, pri čemu svaki član pokazuje na člana sa narednim rednim brojem, osim posljednjeg člana koji pokazuje nazad na prvog člana, čime se zapravo kreira krug članova (takve povezane liste u kojima posljednji čvor u listi pokazuje nazad na prvi čvor nazivaju se *kružne* ili *cirkularne liste*). Nakon što se formira tražena lista, vrši se kretanje kroz listu, polazeći od prvog člana, pri čemu se nakon svakih M napravljenih koraka odstranjuje onaj član iz liste na kojem se trenutno nalazimo. Odstranjivanje se izvodi tako što se pokazivač "`sljedeci`" člana koji prethodi članu na kojem se trenutno nalazimo preusmjerava tako da ne pokazuje više na člana na kojem se trenutno nalazimo nego na člana koji slijedi iza njega (čime se član efektivno odstranjuje iz kruga), nakon čega se prelazi na sljedećeg člana. Tom prilikom je pomoću operatora "**`delete`**" potrebno izbrisati čvor koji odgovara odstranjenom članu, da ne zauzima više memoriju. Postupak se ponavlja dok se ne eliminira svih N osoba, pri čemu je posljednja odstranjena osoba upravo novi poglavica (redni broj te osobe treba vratiti kao rezultat iz funkcije). U svim ostalim detaljima (osim u načinu realizacije funkcije "`Poglavica`"), ovaj program treba biti identičan programu iz prethodnog zadatka.

6. Riješite ponovo prethodni zadatak, ali tako što će će se svugdje umjesto običnih koristiti pametni pokazivači. Vrijede iste napomene kao i u prethodnom zadatku. Suštinska razlika je što Vam ovaj put neće trebati operator "**`delete`**", jer će svaki čvor automatski nestati čim se isključi iz lanca, s obzirom da tada niko neće pokazivati na njega. Međutim, morate paziti da kada ostane samo jedan čvor, njegovo polje "`sljedeci`" će pokazivati na njega samog (tj. upravo na taj jedini preostali čvor). Ovu vezu *obavezno morate raskinuti da biste obrisali taj posljednji čvor*, inače će doći do curenja memorije (jer će taj čvor nastaviti da "čuva samog sebe" čak i kad nestanu drugi pokazivači koji su na njega pokazivali). Prosto rečeno, kružnu listu *morate na nekom mjestu raskinuti prije nego što program završi sa radom*.

NAPOMENA: Sve eventualne nedorečenosti u postavkama zadataka biće razriješene putem javnih autotestova.