1. NAPISATI C PROGRAM koji u datoteku prosti.txt upisuje proste brojeve zadatog intervala kojima je zbir cifara složen broj. Interval se zadaje učitavanjem gornje i donje granice (dva prirodna broja) u komandnoj liniji. Brojeve upisivati u datoteku *prosti.txt* u opadajućem poretku. U programu nije potrebno predvideti obradu greski nastalih usled nekorektnog pozivanja programa u komandnoj liniji ili greske u otvaranju datoteke...

gcc zad1.c -o zad1

POKRETANJE PROGRAMA i TEST PRIMER: zad1 10 40
daje u izlaznoj datoteci prosti.txt IZLAZ
37 31 19 17 13
jer broj 37 je prost, a zbir cifara 3+7=10 je slozen broj
...

```
#include <stdio.h>
#include <stdlib.h>

int prost (int n);  /*testira da li je broj n prost broj */
/*Prirodni brojevi (sem 1) imaju najmanje dva delioca:jedinicu i samog sebe.
   Brojevi koji nemaju drugih delioca,sem ova dva, nazivaju se prostim */

int zbirCifara (int n);   /*vraca zbir cifara broja n */


main(int argc, char **argv)
{ int donja,gornja;  /*granice intervala */
   int i;  /*brojac u petlji */
   int pom;   /*posrednik u eventualnoj trampi granica intervala*/
   FILE *f;


    f=fopen("prosti.txt", "w");

  /* "ucitavanje" granica intervala iz niski argv[1], argv[2] u komandnoj liniji i konverzija u cele brojeve*/
  sscanf(argv[1], "%d", &donja);  /*donja=atoi(argv[1]) */
  sscanf(argv[2], "%d", &gornja);  /*gornja=atoi(argv[2]) */


  if (donja > gornja)   /*obezbedjivanje relacije: donja <=gornja */
    { pom=donja;
      donja=gornja;
      gornja=pom;
    }

  /*brojevi ce se upisivati u datoteku u opadajucem poretku gornja - > donja*/
   for(i=gornja;i>=donja; i--)
    if (prost (i) && !prost(zbirCifara(i) )  ) fprintf(f, "%d  ",i);

  fclose(f);

}


int prost(int n)  /*Ispituje se da li je broj n prost tako to se proverava da li ima delioce
medju brojevima od 2 do n/2.  Pri implementaciji se koristi tvrdjenje da je broj prost ako je jednak 2, ili ako je
neparan i ako nema delitelja medju neparnim brojevima od 3 do n/2 */
{
```

```
  int prost;  /*indikator slozenosti broja n */
  int i;    /*potencijalni delitelj broja n */
  if (n==1) return 0;
  prost= (n%2!=0) || (n==2);    /*parni brojevi razliciti od  od dva nisu prosti brojevi */

  i=3;   /*najmanji potencijalni kandidat za delitelje medju neparnim brojevima razlicitim od jedan */
  while   ( (prost) && (i<=n/2) )
   { prost=n%i != 0;
     i=i+2; /*proveravamo kandidate za delitelje samo medju neparnim brojevma */
    }
 return prost;
}


int zbirCifara (int n)
{ int  Suma=0;
   while (n>0)
  { Suma+= n%10;  /*dodavanje cifre tekuceg razreda,pocev od razreda jedinica ,
                 a iduci ka visim razredima cifara */
    n=n/10;      /*prelaz ka visem razredu */
  }
 return Suma;
}
```

2. Datoteka *ulaz.txt* u folderu *zadaci* sadrzi cele brojeve. Napisati program koji izracunava njihov zbir i upisuje ga u datoteku zbir.txt koja se takodje nalazi u folderu *zadaci*. Pretpostaviti da se izvršiva verzija programa nalazi u istom korenom katalogu gde se nalazi i podfolder *zadaci*.

ulaz.txt	izlaz.txt
12 18 11 14 15 23	93

Da li su navedena rešenja programa korektna?

```
#include <stdio.h>

int main()
{  FILE *f, *g;
   int a,s=0;
   f=fopen("zadaci//ulaz.txt", "r");
   g=fopen("zadaci//izlaz.txt", "w");
   fscanf(f, "%d", &a);
    for (; !feof (f);)
    { s+=a;
      fscanf(f, "%d", &a);
    }
    fprintf(g, "%d", s);
    fclose(f); fclose(g);
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>

int main()
{  FILE *f, *g;
   int a,s=0;
   f=fopen("ulaz.txt", "r");
   g=fopen("zadaci//izlaz.txt", "w");
    for (; !feof (f);)
    {      fscanf(f, "%d", &a);
             s+=a;
    }
    fprintf(g, "%d", s);
    fclose(f); fclose(g);
    return 0;
}
```

3. Napisati program koji sabira cele brojeve navedene u komandnoj liniji (pretpostaviti da su svi navedeni argumenti zaista celi brojevi, tj. da je ulaz sintaksno i semanticki korektan). Na primer, ako se program pokrene sa
saberi 123 456 -78 910
program treba da ispise 1411.

Da li su navedena rešenja programa korektna?

#include <stdlib.h>
#include <stdio.h>
main(int argc, char *argv[])
{
  int i, broj, s=0;
  for (i=0;i<argc;i++)
  s+=atoi(argv[i]);
  printf("%d\n", s);
 return 0;
}
	#include <stdlib.h>
	#include <stdio.h>
	
	main(int argc, char *argv[])
	{ int i, broj, s=0;
	while (--argc>0) s+=atoi (*++argv);
	
	  printf("%d\n", s);
	exit(EXIT_SUCCESS);
	  }

4. **Napisati C program koji ce odštampati na standardni izlaz izveštaj o broju redova, reči, karaktera u datoteci *PROVERA.C***
Pretpostaviti da tip long je dovoljan za sve tri karakteristike iz izveštaja. Smatrati da reč je ma koja niska znakova koja ne sadrži znak razmak, horizontalni tabulator, znak za prelaz u novi red.

```
/* word count (UNIX komanda wc) za  file  "PROVERA.C", pogledati sličan zadatak u I glavi
knjige Kernighan-Ritchie */

PROVERA.C

#include <stdio.h>
main()
{
/*iskazi f-je main su zatvoreni u zagrade */
   printf("Zdravo, svete\n"); /*poziv f-je printf da odstampa poruku*/
}

IZLAZ   6 21 152


#include <stdio.h>
#include <stdlib.h>

typedef enum {VAN, UNUTAR} STANJE;

main (){             STANJE ureci;  /*svojom vrednoscu signalizira VAN/UNUTAR  reci */
       long red_broj, rec_broj, znak_broj;   /*ukupan broj redova,reci,znakova */
       int zn;                                                /*tekuci ucitani znak */
       char ime[]="PROVERA.C" ;           /*ime datoteke koja se analizira */
       FILE *pf;

           pf = fopen(ime, "r");
           if (pf == NULL) {
                fprintf(stderr, "Otvaranje %s nije uspelo.\n", ime);exit (EXIT_FAILURE);
               }

/* analiza sadrzaja fajla  i prebrojavanje karaktera, reci, prelazaka u novi red*/
red_broj = rec_broj = znak_broj = 0;ureci = VAN;

while (( zn = fgetc(pf)) != EOF) {
             ++znak_broj;
               if (zn == ' ' || zn == '\t' || zn == '\n')
                       ureci = VAN;
               else if (ureci == VAN) {
                       ++rec_broj;
                       ureci = UNUTAR;
                }
        if (zn == '\n')++red_broj;
     }
```

```
if (ferror(pf) != 0) {fprintf(stderr, "GRESKA PRI CITANJU: %s\n", ime);     exit
(EXIT_FAILURE); }

/* prikazati rezultate analize */
printf("%s: %ld %ld %ld\n",ime, red_broj, rec_broj, znak_broj);

return (EXIT_SUCCESS);
}
```

5. **Svaki red datoteke, čije se ime unosi kao argument komandne linije, sadrži ime, prezime i korisničko ime studenta (Matematičkog fakulteta u Beogradu) na serveru ALAS. Iza imena i prezimena nalazi se po jedan blanko karakter, a korisnička imena su zapisana u formatu: dvoslovna oznaka smera, poslednje dve cifre godine upisa, četvorocifren broj indeksa. (npr. mr120007, ml110950, aa090034). Napisati program koji generiše HTML datoteku generacija12.htm koja sadrži tabelu sa podacima o studentima koji su upisani 2012. godine na Matematički fakultet. Zaglavlje tabele treba da sadrži polja: ime, prezime, username. Pretpostavka da ulazna datoteka je bez sintaksnih i semantičkih greški.**

ULAZ
Marko Markovic ml090888
Branka Brankovic mr120012
Ana Petrovic mm100088
Petar Andric mv120700
Mika Lazic mn120701
Sanja Petrovic aa090500

IZLAZ generacija12.htm

Ime	Prezime	Username
Branka	Brankovic	mr120012
Petar	Andric	mv120700
Mika	Lazic	mn120701

IDEJA i DISKUSIJA: Ucitava se rec po rec datoteke i nakon svake tri ucitane reci, proverava se vrednost niske username i njeni karakteri na drugoj i trecoj poziciji (jer nulta i prva pozicija cuvaju oznake smera, a druga i treca pozicija oznaku godine upisa u obliku poslednje dve cifre godine).
Kako niske ime, prezime, username za svakog studenta ne sadrze beline, a kako separatori među niskama su blanko, a separator među podacima o studentima je znak za prelaz u novi red, to možemo koristiti bibliotečki funkciju formatiranog ulaza fscanf (tj. nije nužno da implementiramo korisničku funkciju za učitavanje reči). Uočite da implementacija korisničke funkciju za učitavanje reči sa ulaza je bilo potrebno kad god su te reči bile specifične forme (npr. tagovi - niske između uglastih zagrada ili alfanumeričke niske-niske koje sadrže samo slova ili cifre ili tagovi bez atributa - niske između uglastih zagrada i/ili belina,...)

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>

#define MAXIME 20
#define MAXU 9

int main(int argc, char ** argv)
{
  FILE *ulaz, *izlaz;
  char ime[MAXIME], prezime[MAXIME], username[MAXU];

        ulaz=fopen(argv[1], "r");
        izlaz=fopen("generacija12.htm", "w");
```

```
        /*formiranje HTML zaglavlja i tela u izlaznoj datoteci generacija12.htm */
        fprintf(izlaz,"<HTML><HEAD><TITLE>Generacija upisana 2012/13</TITLE></HEAD><BODY>");
        fprintf(izlaz, "<TABLE><TR><TH>Ime</TH><TH>Prezime</TH><TH>Username</TH>");


  while(!feof(ulaz))
  {
    /* Ucitavanje imena iz ulazne datoteke*/
    fscanf(ulaz, "%s", ime);

   /* Ucitava se prezime iz datoteke */
       fscanf(ulaz, "%s", prezime);

    /* Ucitava se username*/
         fscanf(ulaz, "%s", username);

      /* provera da li ucitan potpun sadrzaj za sve tri niske */
              if(strlen(ime) && strlen(prezime) && strlen(username))
                          /* provera da li korisnicko ime sadrzi cifre 12 na pozicijama godina upisa*/
                          if((username[2]=='1') && (username[3]=='2') )
                                      /* upisi podataka u tabelu datoteke generacija12.htm */
        fprintf(izlaz,"<TR><TD>%s</TD> <TD>%s</TD> <TD>%s</TD></TR>", ime, prezime, username );
  }




  /*Upis: tagovi za kraj tabele i i kraj HTML datoteke*/
  fprintf(izlaz,"</TABLE></BODY></HTML>");

  fclose(ulaz);
  fclose(izlaz);
}
```

6. **Napisati program koji ce ispisati argumente komandne linije uz opcije: [-n| -t| -q] , gde opcija –n ukazuje da se ocekuje ispis u novom redu, opcija –t da se ocekuje ispis razdvojen tabulatorom, opcija –q da se ocekuje ispis u kom je svaki argument ogradjen dvostrukim navodnicima**

Test primeri:

ULAZ
zad6 -n 123 456 -78 910

IZLAZ
123
456
-78
910

ULAZ
zad6 -tq 123 456 -78 910
IZLAZ
"123" "456" "-78" "910"

ULAZ
zad6 -qt 123 456 -78 910
IZLAZ
"123" "456" "-78" "910"

ULAZ
zad6 -qtn 123 456 -78 910
IZLAZ
"123"
"456"
"-78"
"910"

```
#include <stdlib.h>
#include <stdio.h>

main(int argc, char *argv[])
{ char znak;   /*znak iz komandne linije koji ucestvuje u testu opcionih argumenata */
   char ispisFormat[4]="";  /*za sada prazna niska ispisnog formata, ciji sadrzaj zavisi od priustva opcionih
argumenata */
   int i, j;                /*brojaci u petljama i indeksi */
   int start=0;             /*signal pojave dvostrukog navodnika ispred parametara komandne linije */
   int  uNovomRedu=0;       /*signalizacija pojave opcije -n */
   int  saNavodnicima=0;    /*signalizacija pojave opcije -q */
   int saTabom=0;           /*signalizacija pojave opcije -t */

/*pretraga flegova  -n, -q, -t  u komandnoj liniji */

   /*prosledjivanje argumenata komandne linije */
while (--argc>0 && (*++argv)[0]=='-' )
   /*prosledjivanje opcija -n, -q, -t*/
     while (znak=*++argv[0] )
         switch (znak )
    {  case 'n':  uNovomRedu=1; break;
       case 'q':  saNavodnicima=1;      start=1;      break;
       case 't':  saTabom=1;  break;
       default :  printf(" Nekorektna opcija  %c u komandi  %s\n\n",znak, argv[0]);
    printf(" Usage:  %s  [-n  -q  -t ]   strings\n\n",argv[0]);
    exit(EXIT_FAILURE);
   }


/*popunjavanje niske ispisFormat  obracajuci paznju na redosled dodela ,jer ce se ispisFormat
"ispisivati"  izmedju svaka dva argumenta  komandne linije  */
j=0;
if (saNavodnicima) ispisFormat[j++]='\"'  ;
if (saTabom) ispisFormat[j++]='\t'  ;
if (uNovomRedu) ispisFormat[j++]='\n'  ;
/*okoncavanje niske , ako je uopste menjana */
if (j)  ispisFormat[j]='\0';

/*ispis u zeljenom obliku,pri cemu deo petlje zaduzen inicijalizaciju je prazan jer je vazno pristupiti tekucoj nisci
*argv.
Uociti da argc se smanjuje u svakoj iteraciji   */
for(; argc>0; argc--, argv++)
   printf("%s%s%s",   start? "\""  :  ""  ,  *argv, ispisFormat );
printf("\n");
```

```
return 0;
}
```

7. **NAPISATI C PROGRAM koji na izracunava i na standardni izlaz ispisuje sumu brojeva koji se zadaju u komandnoj liniji. Celi brojevi se zadaju posle imena programa i opcije -*d*. Realni brojevi se zadaju posle imena programa i opcije -*f*. Predvideti obradu greski pri unosu.**

RESENJE:

Dakle, program mora da radi sa sledecom vrstom poziva iz komandne linije:

- naziv (izvrsive verzije) programa
- znak horizontalna crta
- opcija *d* ili *f* (bez razmaka izmedđu horizontalne crte i opcije)
- niz stringova koji sadrze cele brojeva ako je uneta opcija -i ili niz stringova koji sadrze realne brojeve ako je uneta opcija -f

U slucaju da korisnik u komandnoj liniji zada samo naziv programa, dobro bi bilo da program obavestiti korisnika da nije uneo dovoljno argumenata za pokretanje programa i da program prekine dalji rad.
U slucaju da korisnik u komandnoj liniji izostavi znak horizontalne crte, dobro bi bilo da program program prekine dalji rad i obavestiti korisnika da nije uneo opcije na korektan nacin.
U slucaju da korisnik u komandnoj liniji nakon znaka horizontalne crte unese znak koji nije slovo d ili f, dobro bi bilo da program program prekine dalji rad i obavestiti korisnika da nije uneo opcije na korektan nacin.
U slucaju da korisnik u komandnoj liniji unese opciju -d, onda konvertovati argumente komandne linije u cele brojeve f-jom atoi.
U slucaju da korisnik u komandnoj liniji unese opciju -f, onda konvertovati argumente komandne linije u cele brojeve f-jom atof.

```
#include <stdio.h>
#include <stdlib.h>

typedef enum {GR1, GR2, GR3} kodGreske;

char *greske[]=
{ "Nekorektno pokretanje programa. Morate navesti i opcije i konkretne brojeve\n",
  "Nekorektno zadavanje opcija -d ili -f\n",
  "Nekorektno slovo u opciji. MORATE uneti -d ili -f\n"
};

void ispisGreske(kodGreske broj);

main(int argc, char*argv[])
{
   int i; /*brojac u ciklusu*/
   int s1=0; /*suma celih brojeva */
   double s2=0; /*suma realnih brojeva*/

if (argc<2) ispisGreske(GR1);

if(*argv[1]!='-') ispisGreske(GR2);

/*analiza opcije, tj. slova iza - */
switch(*(argv[1]+1))
{
  case 'd': for (i=2;i<argc;i++)  s1+=atoi(argv[i]);
            printf("\n Suma CELIH brojeva iz komandne linije: %d\n ", s1);
            break;
```

```
 case 'f': for (i=2;i<argc;i++)  s2+=atof(argv[i]);
            printf("\n Suma REALNIH brojeva iz komandne linije: %f\n ", s2);
            break;
 default: ispisGreske(GR3);  break;
 }

}


void ispisGreske(kodGreske j)
{  fprintf(stderr,  greske[j]);
   exit(EXIT_FAILURE);
}
```

8.

NAPISATI C PROGRAM koji ispisuje na standardni izlaz spojeni sadržaj nekoliko datoteka čiji nazivi su dati kao parametri komandne linije. Ako se u komandnoj liniji ne nalazi naziv niti jedne datoteke, onda ispisati tekući sadržaj sa standardnog ulaza.

IDEJA:
U ovom zadatku potrebno je otvoriti datoteke čiji nazivi su sadržani u niskama argv[1], argv[2],..., argv[argc-1]
Ako otvaranje pojedinačne datoteka bude uspešno (fopen(...) != NULL), onda pozvati f-ju pisi(f,stdout) koja sadrzaj datoteke na koji pokazuje f ispisuje na stdout (standardni izlaz, tj. ekran dok ga ne preusmerimo komandom operativnog sistema)
Ako otvaranje datoteke ne uspe, potrebno je ispisati poruku o gresci na stderr(standardni tok za poruke o grešci) i prekinuti dalje izvršvanje programa (exit(1);)
U f-ju pisi(f,stdout) čita se karakter po karakter (sve do EOF) iz datoteke na koju pokazuje f (getc(f)) i ispisuje na standardni izlaz (putc(c,stdout)).
Nakon završetka ispisa sadržaja datoteke, datoteka se zatovori. (fclose(f))

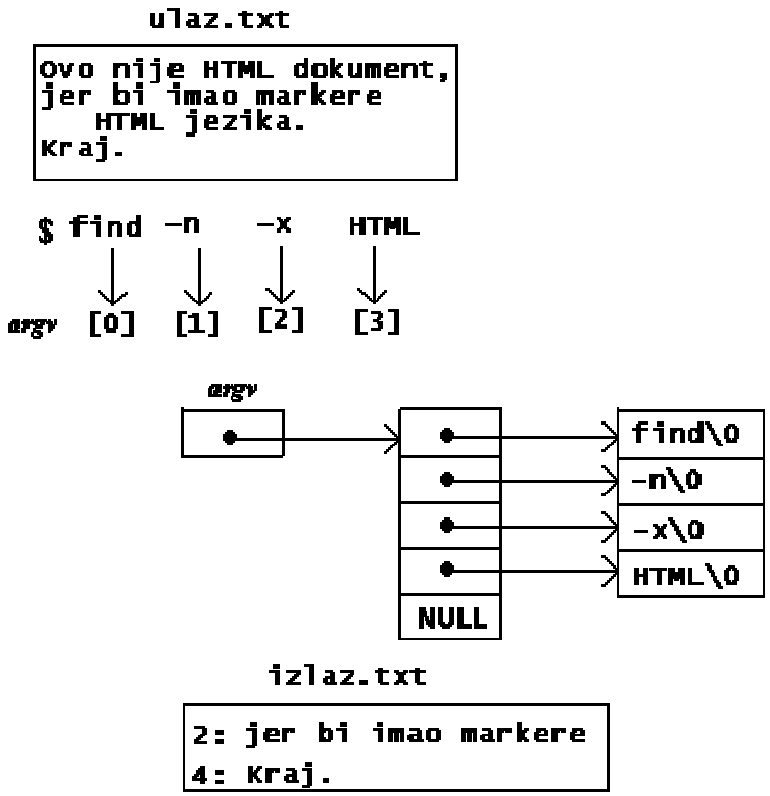
```
#include <stdio.h>

void pisi(FILE *ulaz,FILE *izlaz);    /*prepisuje ulaznu datoteku u izlaznu datoteku */
main(int argc, char *argv[ ])
{ FILE *f;                            /* za citanje */
   int i;                                   /*brojac u petlji */
   char *imeprog=argv[0]; /*ime programa; nije obavezno pri ispisu*/
   if(argc==1) pisi(stdin,stdout);  /* kada se prenosi tekst sa glavnog ulaza,jer u komandnoj
liniji nema imena ostalih datoteka */
   else
/*inace ispisati "spojeni" sadrzaj (ako je moguce) onih  datoteka cija se imena cuvaju kao
argv[1]...argv[argc-1] */

 for(i=1; i < argc;i++)    /*pokusaj otvranja datoteke za citanje */
          if(  (f=fopen(argv[i],"r") ) ==NULL)
       {
           fprintf(stderr, "%s:  Ne mogu otvoriti fajl sa imenom %s\n",imeprog, argv[i]);
           exit(1);        }
       else
            /*ispis sadrzaja upravo otvorene datoteke na standardni izlaz */
             { pisi(f,stdout);  fclose(f);}
     if (ferror(stdout) )
        { fprintf(stderr, "%s: greska pisanja u stdout\n",imeprog);   exit(2);}
      exit ( 0 );
}
```

```
void pisi(FILE *ulaz,FILE *izlaz)   /*kopira ulaz na izlaz */
  { int c;
            while(   (c=getc(ulaz)  )   !=   EOF)         putc(c,izlaz);
}
```

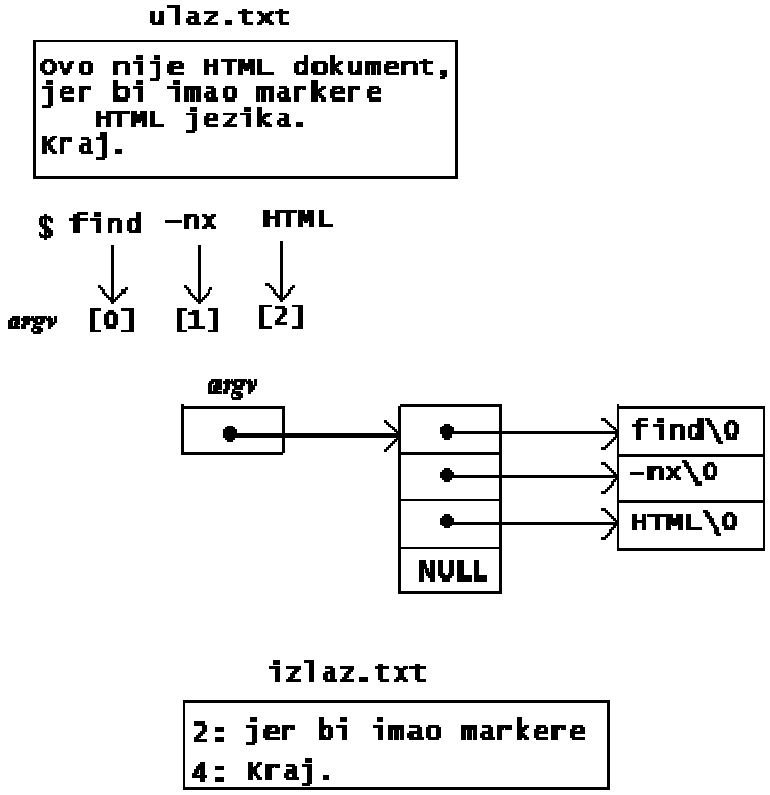
9. /* find_str [-n|-x] pattern */

Argumenti komandne linije su: poziv programa, opcioni argumenti (-n, -x), *obrazac*.
NAPISATI C PROGRAM koji čita linije sa standardnog ulaza do markera kraja i štampa sve linije koje sadrže obrazac(sem ako se među argumentima komandne linije ne nalazi opcija *-x*). Ukoliko se među opcionim argumentima nalazi *-n* ispisuje redni broj linije ispred sadržaja linije. Predvideti mogućnost pojavljivanja oba opciona argumenta u formi -nx, -xn. Pretpostaviti da ukupan broj linija je strogo manji od LONG_MAX.

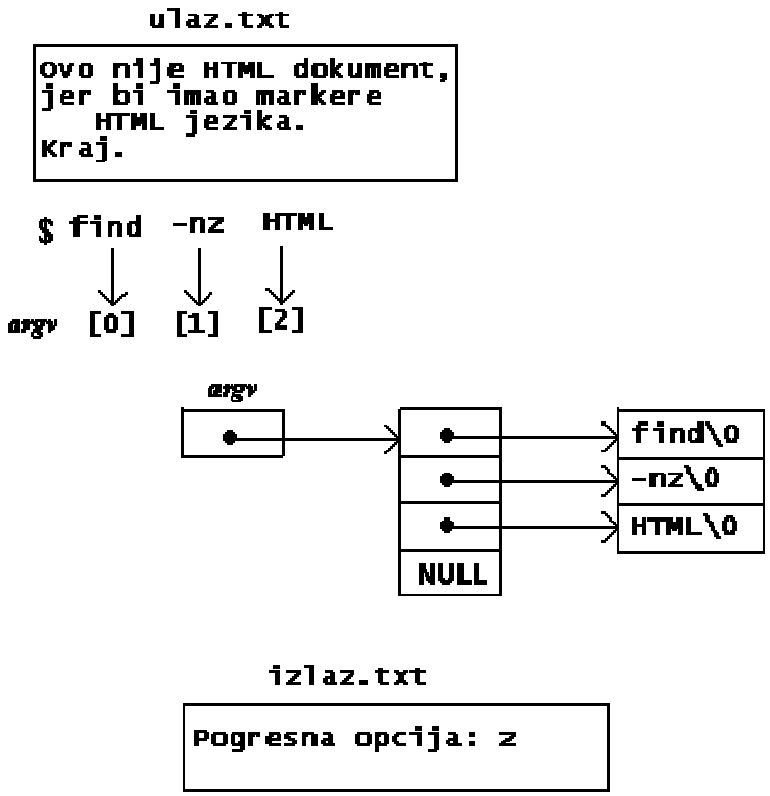
Primer 1: ispisati sve linije teksta sa ulaza koje ne sadrže (opcija: -x) nisku HTML. Uz sadrzaj ispisati i redne brojeve linija. (opcija -n)



Primer 2: isto kao primer 1, ali ilustruje prisustvo i kombinaciju oba opciona argumenta



Primer 3: poziv sa nekorektnom opcijom -z koja je zadata kao argument komandne linije




```
 #include <stdio.h>
#include <string.h>
#define MAXLEN 1000  /*max duzina linije */
```

```
/*ucitava liniju limitirane duzine  i vraca njenu duzinu */
int getline( char line[], int lim);

main(int argc, char *argv[])
{
   char line [MAXLEN] ;   /*tekuca linija */
   long rednibr=0;   /*numeracija linija pod pretpostavkom da je tip long dovoljan */
   int znak;  /*opcija iz poziva */
   int i, j;                /*brojaci u petljama i indeksi */
   int except=0;  /*indikator nailaska na opciju "izuzev" */
   int number=0;  /*zahtev za numeraciju */
   int found=0;      /*rezultat koji se vraca okruzenju */

   /*prosledjivanje argumenata komandne linije */
for (i=1; i< argc && argv[i][0] == '-' ; i++)
   /*prosledjivanje opcija za numeraciju ili "izuzev "*/
  for (j=1; znak=argv[i][j]; j++)
    switch(znak)
        {
            /*"izuzev"  opcija*/
          case 'x' :  except=1;
                      break;
           /* opcija numeracije */
            case 'n' : number=1;
                        break;
            default:  printf("Pogresna opcija: %c\n", znak);
                       argc=0;
                       found=-1;
                       break;
          }


        if (argc != 1) {  printf("Pogresan format poziva! \n");
                            if (found != -1) found=1; }
        else
            /*unos linija do markera kraja */
     while (getline(line,MAXLEN) > 0)
            {
              rednibr++;
             /*zavisno od pokazivaca na prvo pojavljivanje *argv u line i vrednosti opcije "izuzev" stampaju se linije
                koje sadrze pattern ili ne */
              if (  (strstr(line, *argv)  != NULL)  != except )
                {
                    if (number) printf("%ld: ", rednibr);  /*ispisu linije neka prethodi numeracija, ako je tako zahtevano u
komandnoj liniji */
                    printf("%s", line);   /*samo ispis sadrzaja linije */
                  }
  }
            return found;
        }

/* getline: ucitava liniju i vraca njenu duzinu */
int getline( char line[], int granica)
{int znak, indeks;
for( indeks=0;indeks<granica-1 && (znak=getchar())!=EOF && znak!='\n'; ++indeks)
line[indeks]=znak;
if( znak=='\n' ) line[indeks++]=znak;
```

```
line[indeks]='\0';
return indeks;
}
```

10. NAPISATI C PROGRAM koji utvrdjuje koliko se puta data rec pojavljuje u datoj tekstualnoj datoteci. Naziv datoteke kao i rec zadaju se kao argumenti komandne linije.