Sveučilište u Splitu

Sveučilišni studijski centar za stručne studije

Programske metode i apstrakcije

Laboratorijske vježbe

Veljača, 2011.

Toma Rončević

Sadržaj:

1. Osnove rada sa Microsoft Visual Studio C/C++ okruženjem

Ovdje će biti prikazana upotreba Microsoft Visual C/C++ razvojnog sučelja, odnosno kako se kreira nova radna okolina (Workspace), otvara novi projekt, te kako se vrši pokretanje samog koda. Opisani postupci mogu varirati ovisno o verziji MS Visual Studia kojeg koristite. Svejedno, sve verzije imaju implementirane funkcionalnosti opisane u ovoj vježbi. Opisani postupak se odnosi na Microsoft Visual C++ 2008 Express Edition (MVC) koji je besplatan i možete ga naći na http://www.microsoft.com/express/download.

Prvo pokrenite MVC. Da bi kreirali novu radnu okolinu, u glavnom izborniku odaberite opciju **FILE⇨NEW⇨PROJECT.** U novootvorenom prozoru odaberite opciju ***Win32/Windows Console Application***. Pod ***Location*** treba pisati d:\temp, dok pod ***Solution Name*** treba pisati ime prezime PMA, a pod ***Name*** treba pisati Vjezba i broj vjezbe (u ovom slučaju Vjezba1). Pritisnite tipku **OK**. Na slijedećem prozoru čarobnjaka pritisnite **Next**, uključiti opciju **Empty projekt** i nakon toga pritisnuti **Finish**.

Ubuduće kada budete prelazili na novu vježbu (ili dio vježbe) sve će biti isto osim što ćete na prvom prozoru čarobnjaka odabrati ***Add to solution*** umjesto ***Create new solution*** pod stavkom ***Solution*.** Na kraju će se svi zadaci nalaziti pod različitim projektima unutar istog «rješenja» (**Solution**).

1. Zadatak

Napravite novi projekt: Vjezba1.

Sada ste kreirali prazan projekt i odgovarajuće rješenje (***Solution***). Uz desni rub ekrana se nalazi prozor radne okoline, na kojem možete pogledati što sve sadrži vaš projekt. Trenutno ne sadrži ništa jer je prazan osim tri mape u koje ćemo dodavati dijelove projekta. Budući da ćete programirati u programskom jeziku C, desnim klikom na **Source Files** dobijate opciju da dodate novu datoteku u projekt. Neka to bude .cpp datoteka sa imenom istim kao i ime vježbe. Kada je datoteka dodana, preminujte je tako da joj je nastavak .c umjesto .cpp.

Sada je sve spremno za upisivanje programskog koda. Programski kod se upisuje u .c datoteku i nećemo pisati programe koji se protežu na više .c i .h datoteka.

Da bi upisani programski kod proradio potrebno ga je prvo prevesti, zatim povezati pa tek onda pokrenuti. Prevođenje i povezivanje programa može se obaviti odabirom opcije **BUILD⇨COMPILE** ili se prevođenje i povezivanje

automatski pokreće pokretanjem programa (**BUILD⇨EXECUTE**) ako program nije preveden ili je promijenjen od zadnjeg prevođenja.

2. Zadatak

Utipkajte sljedeći programski kod.

```
#include <stdio.h>

// funkcija za zbrajanje dva broja
int zbroji(int x, int y){
   return (x+y);
}

// funkcija za mnozenje dva broja
int mnozi(int x, int y){
   return (x*y);
}

//glavni program
void main(){
   int a,b;
   int zbroj,umnozak;

   printf("Program koji zbraja i mnozi dva broja\n\n");
   printf("Upisite dva cjelobrojna broja --> ");
   scanf("%d %d",&a,&b);

   // pozivanje funkcija
   zbroj = zbroji(a,b);
   umnozak = mnozi(a,b);

   printf("\nZbroj brojeva %d i %d iznosi %d.\n\n",a,b,zbroj);
   printf("Umnozak brojeva %d i %d iznosi %d.\n\n",a,b,umnozak);
}
```

Prvo prevedite i povežite utipkani programski kod te ga zatim pokrenite. Ako ste sve točno prepisali vaš bi se program trebao izvesti.

3. Zadatak

U liniji

zbroj = zbroji(a,b);

izbrišite točku zarez na kraju linije te ponovno pokušajte prevesti i pokrenuti vaš program. Što se desilo?

4. Zadatak

U liniji

scanf("%d %d",&a,&b);

izbrišite znak & ispred varijable a i pokrenite program. Što se desilo?

5. Zadatak

Na ovom primjeru se mogu isprobati neke mogućnosti ***debuggera***. ***Debugger*** se pokreće opcijama **BUILD⇨START DEBUG⇨GO, BUILD⇨START DEBUG⇨STEP INTO** i **BUILD⇨START DEBUG⇨GO TO CURSOR.** Kada se pokrene ***debugger*** u glavnom meniu se pojavi podmeni **DEBUG**, a pri dnu ekrana se pojavi prozor na kojem možete pratiti vrijednosti pojedinih varijabli. Isprobajte opcije **STEP INTO, STEP OVER, STEP OUT** i **GO TO CURSOR** menija **DEBUG**. Uočite razliku između prve dvije opcije kada je kao slijedeću naredbu potrebno pozvati potprogram. **Zadatak: pratiti kako se mijenja vrijednost neke varijable (varijable u koju će se zapisivati rezultat funkcije zbrajanja) tijekom izvođenja programa.**

2. Varijable i standardni ulaz/izlaz

Potrebna predznanja:

- Microsoft Visual C++ okruženje
- upotreba #include pretprocesne naredbe
- main() funkcija
- deklaracija primitivnih tipova varijabli (char, int, float)
- printf i scanf funkcije
- operator pridruživanja i pretvorba tipova
- matematički i bitovni operatori
- if-else struktura

1. Zadatak

Napišite program koji za cijelobrojni broj koji unese korisnik odredi:

a) Je li paran
b) Je li negativan
c) Je li dijeljiv sa 17

2. Zadatak

Napišite program koji ima za zadatak da ispiše svaku znamenku cjelobrojnog broja koji unese korisnik ispiše u novoj liniji. Od korisnika se očekuje da unese pozitivan cijeli broj sa točno 5 znamenaka. Ispisati znamenke od najmanje značajne do najviše značajne i obrnuto.

3. Zadatak

Napisati program koji generira i ispisuje slučajni cjelobrojni broj u rasponu koji ovisi o parametrima koje zadaje korisnik (min i max). Za ograničavanje broja koristiti će se formula koja bilo koji broj *x* pretvara u broj *y* koji se nalazi u granicama [*min,max*]: $y = (x \% (max-min+1)) + min$.

Kôd za generiranje slučajnog broja:

```
// uključiti slijedeće biblioteke
#include <stdlib.h>
#include <time.h>

// pozvati samo jednom pri početku programa
srand((unsigned)time(0));
```

```
// pozivati funkciju rand() svaki put kada
// nam je potreban novi slučajni broj
trazeniBroj=rand();
```

Funkcija srand(sjeme) inicijalizira generator slučajnih brojeva pomoću proslijeđenog sjemena. Taj generator se može gledati kao jedna duga lista brojeva iz koje izvlačimo brojeve pomoću funkcije rand(). Najčešće će se funkcija srand() pozvati samo jednom negdje pri početku programa. Sjeme koje proslijedimo funkciji ćemo ili sami definirati ili uzeti rezultat funkcije time() odnosno broj sekundi koji je prošao od 1970. godine.

Za više o slučajnim brojevima pogledati MSDN help u okruženju u kojem radite.

4. Zadatak

Napišite program koji za znak koji unese korisnik odredi:

a) Je li najznačajniji bit tog znaka 1 ili 0
b) Je li najmanje značajan bit tog znaka 1 ili 0

i ispiše svih 8 bitova ASCII kôda tog znaka.

5. Zadatak

Napišite program koji će od korisnika čitati jedan pozitivan broj i razlomiti taj broj na 4 byte-a (tip unsigned char) i ispisati ih.

6. Zadatak

Napišite program koji od korisnika čita 3 broja koji predstavljaju vrijeme (sati 0-12, minute i sekunde). Kodirajte to vrijeme u jednu varijablu tipa unsigned short. Iz te varijable izvucite sate, minute i sekunde i ispišite ih na ekran.

7. Zadatak

Napišite program koji od korisnika čita jedan broj. Program će ispisati taj broj u binarnom, ternarnom (baza 3), oktalnom (baza 8) sustavu. Ne koristiti ugrađene mogućnosti ispisa broja sa različitim bazama pomoću printf!

3. Kontrola toka programa

Potrebna predznanja:

- uvjeti i logički operatori
- if-else if-else struktura
- switch-case struktura
- while, do-while i for petlje

1. Zadatak

Napraviti program koji od korisnika dobiva cijele brojeve. Program za svaki broj ispisuje:

a) Parnost / neparnost broja
b) Zbroj svih brojeva do tog broja
c) Produkt svih brojeva do tog broja

Program se zaustavlja ako korisnik unese negativan broj.

2. Zadatak

Napraviti program koji od korisnika dobiva cijele brojeve. Program za svaki broj ispisuje:

a) Zbroj svih parnih brojeva do tog broja ako je uneseni broj paran
b) Produkt svih neparnih brojeva do tog broja ako je uneseni broj neparan
c) Poruku o grešci ako je broj negativan

Progam se zaustavlja i ispisuje poruku „Kraj!“ ako korisnik unese 0.

3. Zadatak

Napraviti program koji od korisnika dobiva dva decimalna broja i računsku operaciju u obliku znaka '+', '-', '/' ili '*'. Program ispisuje odgovarajući rezultat računske operacije na ta dva broja. Zadatak riješiti pomoću *switch-case*.

4. Zadatak

Napišite program koji ispisuje prvo deset slučajnih parnih brojeva, a nakon toga deset slučajnih neparnih brojeva. Raspon slučajnih brojeva treba upisati korisnik. Za generiranje slučajnih brojeva može se koristiti kôd iz prvog poglavlja.

5. Zadatak

Napišite program koji od korisnika prima 10 brojeva i ispisuje zbroj svih unesenih brojeva.

6. Zadatak

Napišite program koji od korisnika prima brojeve sve dok korisnik ne unese 0 i ispisuje zbroj svih unesenih brojeva.

7. Zadatak

Napišite program koji od korisnika prima brojeve i računa njihov zbroj. Program ispisuje zbroj svih unesenih brojeva i završava tek kada zbroj pređe 100.

8. Zadatak

Napišite program koji od korisnika prima znakove od korisnika. Za svaki znak se mora odrediti:

a) Je li znak slovo
b) Je li znak znamenka
c) Je li znak samoglasnik (samo ako je malo slovo)
d) Je li znak suglasnik (samo ako je veliko slovo)

Program završava kada korisnik upiše znak koji nije ni znamenka ni slovo.

9. Zadatak

Napišite program koji pomoću iduće formule računa broj π.

$$\pi = 4 (1 - 1/3 + 1/5 – 1/7 + 1/9 – 1/11 \ldots)$$

Korisnik će odrediti koliko će se elemenata zbrojiti u sumu.

4. Funkcije i parametri

Potrebna predznanja:

- deklariranje, definiranje i pozivanje funkcija
- proslijeđivanje parametara funkciji
- povratna vrijednost funkcije
- rekurzija

1. Zadatak

Napišite program koji će u petlji svaki put korisnika pitati da upiše neki broj, i pozvati dvije funkcije: jednu koja će ispisivati sumu parnih i jednu koja će ispisivati produkt neparnih brojeva do tog broja. Petlju treba ponavljati sve dok korisnik ne upiše nulu. Ako korisnik upiše negativan broj treba pozvati funkciju koja će ispisati odgovarajuću poruku i ne izračunavati sumu i produkt.

2. Zadatak

Napišite program koji će u petlji svaki put korisnika pitati da upiše neki broj i redom pozvati 3 funkcije i ispisati njihove povratne vrijednosti:

a) funkciju koja će vratiti kvadrat tog broja
b) funkciju koja će tražiti unos novog broja kojeg će pribrojiti broju koji vrati prethodna funkcija i vratiti taj zbroj
c) funkciju koja će vratiti rezultat slijedeće formule: $9x \sin x + 6x \cos x$ gdje je *x* vrijednost koju vrati prethodna funkcija (za sinus i kosinus upotrijebiti funkcije iz biblioteke *math.h*)

3. Zadatak

Napisati funkciju koja od korisnika očitava koordinate u prostoru (3 realna broja). Glavni program poziva tu funkciju za dvije točke *p1* i *p2* u prostoru i izračunava udaljenost između njih po formuli:

$$d=\sqrt{(p1x-p2x)^2+(p1y-p2y)^2+(p1z-p2z)^2}.$$

4. Zadatak

Napišite dvije verzije funkcije koja vraća potenciju realnog broja $f(x,y) = x^y$:

a) Iterativnu verziju (upotrebom petlje)
b) Rekurzivnu verziju

5. Zadatak

Napisati program koji će dokazati sljedeću matematičku tvrdnju.

Za |q|<1vrijedi qk→0 vrijedi:

$$\sum_{n=1}^{\infty} q^{n-1} = \lim_{k\to\infty} s_k = \frac{1}{1-q}.$$

Neka parametar q, korisnik upiše sa tipkovnice. Sumu u prvom dijelu formule ne treba raditi do beskonačno kao na formuli, nego je treba napraviti do broja N koji odredi korisnik preko tipkovnice. Program će ispisati vrijednost koja se dobije na lijevoj i na desnoj strani jednakosti i pokazati će se da su one jednake za dovoljno veliki N.

6. Zadatak

Napisati program koji će tražiti najveći zajednički djelitelj od dva broja koja unese korisnik. Za traženje najvećeg broja koristite rekurzivnu funkciju *NZD(x,y)* napisanu po Euclidovom algoritmu:

NZD(x,y) za brojeve *x* i *y*:

Ako je *x == y* onda je njihov najveći zajednički djelitelj *x (ili y)*.

Ako je *x > y* onda je njihov najveći zajednički djelitelj *NZD(x-y,y)*

Ako je *x < y* onda je njihov najveći zajednički djelitelj *NZD(x,y-x)*

7. Zadatak

Napisati program koji će iscrtavati piramidu od zvjezdica:

```
    *
   ***
  *****
```

****

Za realizaciju programa napisati funkciju koja iscrtava jednu liniju po primljenim parametrima (parametri su koliko razmaka i koliko zvjezdica treba iscrtati u redu) i dodatnu petlju koja iscrtava traženi broj linija (po unosu korisnika) pozivajući tu funkciju s odgovarajućim parametrima.

5. Pokazivači i dinamičko alociranje memorije

Potrebna predznanja:

- deklariranje i tipovi pokazivača
- proslijeđivanje parametara funkciji preko pokazivača
- memorijski model za prikaz varijabli i pokazivača

1. Zadatak

Napisati funkciju koja zamjenjuje vrijednost dva broja. U glavnom programu učitati dva cjelobrojna broja sa tipkovnice, pozvati funkciju koja zamjenjuje međusobno njihove vrijednosti i u glavnom programu ispisati zamijenjene vrijednosti.

2. Zadatak

Ispravite i testirajte slijedeće funkcije (uz minimalne izmjene kôda u funkciji i bez izmjene kôda u main funkciji!):

a)
```
// funkcija ispisuje zbroj dvije varijable
void zbroj(int a, int *b)
{
     printf(„Zbroj %d i %d je %d\n", &a, &b, &(*a+*b));
}

void main()
{
     int a,b;
     scanf(„%d %d",&a,&b);
     zbroj(&a,&b);
}
```

b)
```
// funkcija vraća pokazivač na najveću od tri varijable
int* veci(int a, int *b, int c)
{
     if (a>=b && a>=c)
          return a;
     if(b>=a && b>=c)
          return b;
     return c;
```

```
}

void main()
{
     int a,b,c,*max;
     scanf(„%d %d %d“,&a,&b,&c);
     max = veci(&a,&b,&c);
     printf(„Največi je: %d“, *max);
}
```

c)
```
// funkcija poreda vrijednosti parametara(jedno slovo) po abecedi
void veci(char a, int b, char c)
{
     char *p1, *p2,* p3;
     // najveci
     if (a>=b && a>=c)
          p1 = *a;
     else if(b>=a && b>=c)
          p1 = b;
     else
          p1 = &c;
     // najmanji
     if(a <= b && a<= c)
          p3 = a;
     else if(b<=a && b<=c)
          p3 = *b;
     else
          p3 =& c;
     // srednji
     if(p1<=a && a<=p3)
          p2 = a;
     else if(p1<=b && b<=p3)
          p2 = b;
     else
          p2 = c;
     *a = *p1;
     *b = *p2;
     *c = *p3;
}

void main()
{
     char a,b,c;
```

```
        scanf(„%c %c %c“,&a,&b,&c);
        veci(&a,&b,&c);
        printf(„%c %c %c\n“, a,b,c);
}
```

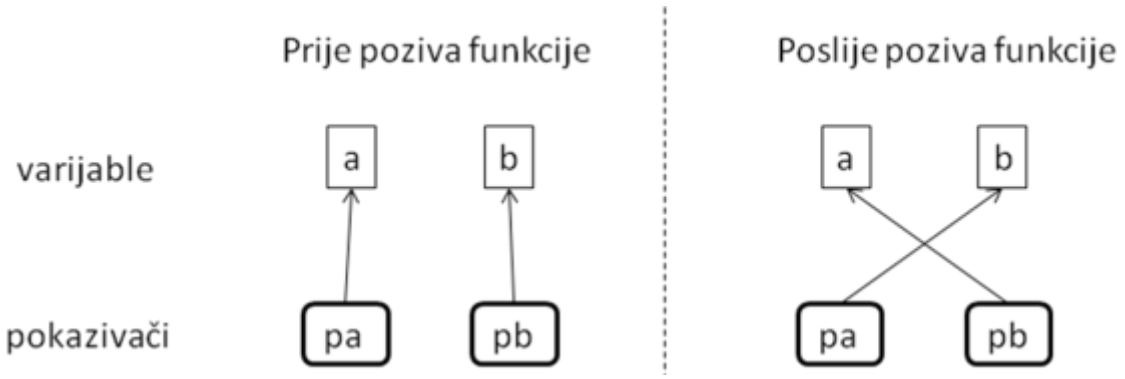
3. Zadatak

Napisati funkciju koja prima kao parametre 3 pokazivača na cjelobrojne varijable. Funkcija će stvoriti ili preusmjeriti pokazivač na varijablu koja ima najveću vrijednost. Zadatak riješiti na dva načina:

- funkcija vraća pokazivač na varijablu preko povratne vrijednosti
- funkcija koristi dvostruki pokazivač kao dodatni parametar funkcije koji preusmjerava na varijablu i nema povratne vrijednosti

4. Zadatak

Napisati funkciju koja prima dva (dvostruka) pokazivača. Funkcija zamjenjuje adrese na koje pokazuju primljeni pokazivači.



6. Nizovi

Potrebna predznanja:

- deklariranje nizova
- indeksiranje i obrada elemenata u nizovima
- upotreba #define predprocesne naredbe
- zauzimanje i oslobađanje memorije sa malloc i free funkcijama
- alociranje memorije za složene tipove podataka (stringove, nizove i strukture)
- cast operator
- NULL konstanta
- sizeof() operator

1. Zadatak

Napisati funkciju koja ispisuje srednju vrijednost niza od 10 realnih brojeva. Glavni program očitava brojeve od korisnika i poziva tu funkciju.

2. Zadatak

Napisati funkciju koja vraća srednju vrijednost niza od N realnih brojeva, gdje je N definiran globalno pomoću *#define*. Glavni program očitava brojeve od korisnika i poziva tu funkciju.

3. Zadatak

Napisati funkciju koja vraća srednju vrijednost niza od n realnih brojeva, gdje je n dodatni ulazni parametar funkcije. Glavni program očitava dužinu niza i brojeve od korisnika i poziva tu funkciju. Najveća dozvoljena dužina niza je 30 (ispisati odgovarajuću poruku o grešci ako korisnik ne poštuje to ograničenje).

4. Zadatak

Napisati funkciju koja vraća najveću vrijednost u nizu realnih brojeva. Niz je dužine *n*.

5. Zadatak

Napisati funkciju koja vraća indeks najveće vrijednosti u nizu realnih brojeva. Niz je dužine *n*.

6. Zadatak

Napisati funkciju koja provjerava da li niz od N cijelih brojeva sadrži više istih vrijednosti. Funkcija vraća 0 ako niz sadrži sve različite vrijednosti ili 1 u suprotnom.

7. Zadatak

Napisati funkciju koja će alocirati potrebnu memoriju za niz realnih brojeva dužine *n*. Funkcije kao parametar dobiva dužinu niza *n*, a vraća pokazivač na alociranu memoriju.

7. Stringovi

Potrebna predznanja:

- deklariranje, unos i ispis stringova
- funkcije gets i puts
- osnovne funkcije za rad sa stringovima (strlen, strcmp, strncmp, strcpy, strncpy, strcat)
- ASCII kodovi

1. Zadatak

Napisati funkciju koji izračunava koliko se puta zadani znak pojavljuje u nekom stringu. Funkcija će kao parametre primati string i znak.

2. Zadatak

Napisati funkciju koja prima string koji sadrži realan broj (na primjer „-23.456“) i vraća odgovarajući realan broj (tipa float). Funkcija ne koristi nikakve funkcije tipa scanf i slično. String obrađujete znak po znak.

3. Zadatak

Napisati program koji će učitavati liniju po liniju 3 recenice (u tri varijable). Nakon toga ih ispiše poredane po dužini i poredane po abecednom redoslijedu. Napisati dvije odvojene funkcije koje rade te dvije zadaće.

4. Zadatak

Napisati funkciju koja će tri riječi, proslijeđene joj kao parametri, spojiti u jednu jedinstvenu rečenicu (pomoću funkcija strcpy i strcat) i vratiti je preko četvrtog parametra.

5. Zadatak

Napisati funkciju koji izračunava koliko se puta zadana riječ pojavljuje u zadanoj rečenici kao samostalna riječ i koliko puta kao dio neke druge riječi. U glavnom programu upisati rečenicu i riječ koja se traži u rečenici, pozvati funkciju te ispisati koliko se puta riječ pojavljuje kao samostalna riječ, a koliko puta kao dio neke druge riječi.

6. Zadatak

Napisati program koji će učitavati liniju po liniju, te svaku liniju spremiti u polje pokazivača na char tip podataka. Svaku liniju treba smjestiti u sljedeći član polja, te za svaku liniju alocirati točno određeni memorijski prostor. Dobiveno polje pokazivača sortirati po dužini i ispisati.

8. Strukture

Potrebna predznanja:

- deklariranje struktura
- pristup članovima strukture preko varijable ili pokazivača strukture
- upotreba typedef ključne riječi

1. Zadatak

Napisati funkciju koja će na temelju proslijeđenih parametara alocirati i ispuniti slijedeću strukturu:

```
struct Natjecatelj {
      float najbolji_rezultati[3];
      char *ime;
      char maticni_broj[12];
      char *adresa;
      int visina;
      int tezina;
};
```

Napomena: pazite na alokaciju članova strukture.

2. Zadatak

Deklarirati strukturu koja predstavlja računalo i ima slijedeće članove:

- ime procesora
- količina memorije
- grafička kartica
- cijena

Član 'grafička kartica' je druga struktura sa članovima:

- količina memorije
- marka
- cijena

Odabrati odgovarajuće tipove za svaki član.

(***Za sve slijedeće zadatke koristiti strukture iz ovog zadatka***)

3. Zadatak

Napisati funkciju koja će od korisnika pročitati sve podatke o računalu i grafičkoj kartici i smjestiti ih u proslijeđenu strukturu. Napisati funkciju koja će ispisati proslijeđeno računalo (ispisati sve članove računala i grafičke kartice).

4. Zadatak

Napisati funkciju koja ispisuje prosiječnu cijenu računala i prosiječnu cijenu grafičke kartice u proslijeđenom nizu računala.

5. Zadatak

Napisati funkciju koja će iz proslijeđenog niza vratiti pokazivač na najjeftinije računalo.

6. Zadatak

Napisati funkciju koja će sortirati niz računala uzlazno po cijeni grafičke kartice. Za sortiranje se može upotrijebiti funkcija qsort() funkcija iz stdlib.h biblioteke ili implementirati bubble sort algoritam.

(*) Radi bržeg testiranja funkcija u zadacima potrebno je napraviti tekstualnu datoteku *racunala.txt* koja sadrži podatke (u istom redoslijedu po kojem ih čitamo u programu od korisnika, svaki podatak u novom redu) o nekoliko računala:

Pentium

512

128

Nvidia

1000

3000

Athlon

1024

...

Snimiti tu datoteku u *Debug* mapu projekta (gdje se nalazi i izvršni program vježbe). Pokrenuti konzolu (*Accessories -> Command Prompt*) i sa *cd* naredbom doći u *Debug* mapu. U toj mapi izvršiti:

Vjezba.exe < racunala.txt

Tako ćemo, umjesto da svaki put upisujemo podatke za sva računala u nizu, dobiti gotov ulaz (koji će zamijeniti standardni ulaz sa konzole) bez potrebne za izmjene u programu.

9. Rad s tekstualnim datotekama

Potrebna predznanja:

- Povezivanje s tekstualnim datotekama iz programa
- osnovne funkcije za rad s tekstualnim datotekama fopen, fclose
- dodatne funkcije za rad s datotekama fprintf, fscanf, feof, fgets, fputs, fgetc

1. Zadatak

Napišite program koji od korisnika prima liniju po liniju teksta sve dok korisnik ne upiše praznu liniju i snima ih u datoteku *linije.txt*.

2. Zadatak

Napišite program koji čita (liniju po liniju) datoteku *linije.txt* i ispisuje sadržaj datoteke na konzolu, izostavljajući velika slova.

3. Zadatak

Napišite program koji kopira datoteku znak po znak i program koji kopira datoteku liniju po liniju.

4. Zadatak

Napišite program koji sortira linije datoteke *linije.txt* po abecednom redu u novu datoteku *sortirano.txt*.

10. Rad s binarnim datotekama

Potrebna predznanja:

- Povezivanje s binarnim datotekama iz programa
- osnovne funkcije za rad s binarnim datotekama fread, fwrite, fseek

1. Zadatak

Napišite program koji snima niz od 10 struktura u datoteku. Svaka struktura predstavlja vrijeme i sastoji se od tri broja (sati, minute, sekunde). Snimanje se radi pomoću fwrite u binarnom formatu.

2. Zadatak

Napišite program koji čita niz od 10 struktura (iz prvog zadatka) iz datoteke.

3. Zadatak

Napišite funkciju koja sortira datoteku u koju je snimljen niz struktura (iz prvog zadatka) po vremenu od najmanjeg do najvećeg. Sortiranje se izvodi pmoću bubble sortiranja u istoj datoteci bez pomoći nizova struktura (zamjena dva elementa se izvodi direktno čitajući/pišući po datoteci).

5. Zadatak

Napišite program koji kriptira ili dekriptira datoteku. Program prvo pita korisnika ime datoteke koju treba kriptirati ili dekriptirati, lozinku i ime datoteke u koju se upisuje rezultat kriptiranja/dekriptiranja. Program čita tu datoteku znak po znak (binarno!). Svaki znak se kriptira tako da se izračuna novi znak koji je jednak operaciji XOR sa jednim znakom iz lozinke (pogledati sliku).

```
        N e k a k a v   t e k s t u   d a t . . .
XOR     l o z i n k a l o z i n k a l o z i n . . .
-------------------------------------------------
        ? u & % f # ” g h   j ” ! u ) g . . .
```

Slika 1 Kriptiranje teksta znak po znak pomoću XOR operatora

11. Dinamičke strukture podataka (jednostruko i dvostruko povezane liste)

Potrebna predznanja:

- deklariranje strukture za elemente liste
- proslijeđivanje liste funkciji
- model jednostruko i dvostruko povezane liste

Za slijedeće zadatke definirati strukturu elementa vezane liste:

```
struct Element {
      int broj;
      char ime[100];
      struct Element *next;
};
```

1. Zadatak

Napišite funkciju koja će listi koja sadrži dva elementa zamijeniti redoslijed elemenata.

2. Zadatak

Napišite funkciju koja će od tri proslijeđena elementa (kao tri odvojena parametra) formirati jednu vezanu listu.

3. Zadatak

Napišite dvije funkcije: jednu bez povratne vrijednosti koja dodaje element na početak liste i jednu koja vraća listu kojoj je na početak dodan element. Obje funkcije kao argumente primaju listu i element koji dodaju listi. (Obje funkcije čine istu stvar samo je razlika u načinu poziva funkciji.)

4. Zadatak

Napišite strukturu koja apstraktira natjecatelja u nekoj utrci (sadrži ime natijecatelja i postignuto vrijeme) i može se povezati u jednostruko povezanu listu. Napišite dvije funkcije koje rade s proslijeđenom listom rezultata neke utrke: funkciju koja vraća najbolje vrijeme utrke i funkciju koja vraća pobjednika utrke.

5. Zadatak

Napišite dvije funkcije (koje rade sa listama iz prethodnog zadatka): funkciju koja dodaje još jednog natijecatelja i funkciju koja diskvalificira nekog natijecatelja (funkcija iz liste briše natijecatelja po datom imenu).

6. Zadatak

Napišite strukturu koja će biti osnovni element dvostruko povezane liste. Struktura će apstaktirati nekog studenta (ime, prezime i matični broj). Napisati funkcije koje:

- Dodaje novog studenta listi
- Uklanja studenta sa liste
- Provjerava da svi studenti imaju matični broj veći od 0
- Sortiraju listu po matičnom broju

12. Dodaci

A. Dio tablice ASCII znakova koji se mogu ispisati:

Kôd	Znak	Kôd	Znak	Kôd	Znak	Kôd	Znak
32	razmak	80	P	56	8	104	h
33	!	81	Q	57	9	105	i
34	"	82	R	58	:	106	j
35	#	83	S	59	;	107	k
36	$	84	T	60	<	108	l
37	%	85	U	61	=	109	m
38	&	86	V	62	>	110	n
39	'	87	w	63	?	111	o
40	(	88	X	64	@	112	p
41	)	89	Y	65	A	113	q
42	*	90	Z	66	B	114	r
43	+	91	[	67	C	115	s
44	,	92	\	68	D	116	t
45	-	93	]	69	E	117	u
46	.	94	^	70	F	118	v
47	/	95	_	71	G	119	w
48	0	96	`	72	H	120	x
49	1	97	a	73	I	121	y
50	2	98	b	74	J	122	z
51	3	99	c	75	K	123	{
52	4	100	d	76	L	124	|
53	5	101	e	77	M	125	}
54	6	102	f	78	N	126	~
55	7	103	g	79	O	127	DEL

B. Formatiranje za printf, scanf i slične funkcije:

%d – ispis cjelobrojnog broja (*int, unsigned int*)
%f –ispis realnog broja (*float*)
%lf – ispis dugog realnof broja (*double*)
%c – ispis znaka (*char*)
%s – ispis stringa (*char**)
\n – ispis novog reda
\t – ispis tab znaka
\0 – string terminator

13. Literatura

- W. Kernighan, D. M. Ritchie, „The C Programming Language“, 2nd Edition, 1988.
- Dennis M. Ritchie, Brian W. Kernighan, „Programski jezik C“, drugo izdanje
- J. Dadić, N. Lavicki-Šatović, „Programske metode i apstrakcije“, bilješke s predavanja – 2003/2004