Izrada jednostavnih programa u C-u i ispravljanje pogrešaka

Prilikom pisanja programa pogreške se mogu podijeliti u dvije skupine: **sintaktičke pogreške** i **logičke pogreške**.

Sintaktičke pogreške uglavnom nastaju pogrešnim unošenjem kôda u računalo ili nepoznavanjem leksičke strukture programskog jezika. Najčešća pogreška prilikom programiranja u programskom jeziku C je izostavljanje znaka **';'** poslije svake naredbe. Osim toga česte su i pogreške krivog upisivanja naziva funkcija (bilo korisničkih, bilo sistemskih – na primjer **`prinf`** umjesto **`printf`**).

Pojavom logičkih pogrešaka javlja se situacija kod koje **program radi, ali ne radi ispravno**. Npr: ako se program za zbrajanje dva broja prevede i poveže bez pogreške, a unošenjem podataka dobije se pogrešan rezultat, napravljena je logička pogreška.

Ispravljanje sintaktičkih pogrešaka puno je jednostavnije od ispravljanja logičkih pogrešaka. U najvećem broju slučajeva pojavu sintaktičke pogreške javlja prevodilac, a ostale slučajeve dojavi povezivač. Sintaktičke pogreške ispravljaju se navođenjem ispravne jezične sintakse, navođenjem ispravnih naziva funkcija i sl. Logičke pogreške teže se ispravljaju, a moguće ih je otkriti ulaskom u način rada za ispravljanje pogrešaka (**debugger**) ili ispisivanjem vrijednosti varijabli unutar programa na određenim **kritičnim** mjestima.

Ispravljanje pomoću debuggera može se obaviti na slijedeći način:

- postaviti prekidnu točku prije dijela izvornog koda za koji se sumnja da sadrži logičku pogrešku;
- pokrenuti program uz izvođenje dok ne stane u prekidnoj točki;
- izvoditi redak po redak izvornog koda uz ispisivanje "sumnjivih" varijabli.

Ispravljanje ispisivanjem varijabli na određenim kritičnim dijelovima programa može se obaviti na sljedeći način:

- na kritična mjesta u programu postavimo naredbu (funkciju) za ispisivanje – **`printf`** – koja će ispisati vrijednosti sumnjivih varijabli;
- prema potrebi nakon ispisivanja vrijednosti postavi se naredba koja će zaustaviti izvođenje programa dok se ne pritisne bilo koja tipka – **`getch`**[1];
- izvođenje redak po redak nije moguće bez debuggera pa zbog toga treba prva dva navoda postaviti na sva kritična mjesta unutar programa.

Ispravljanje pogrešaka ispisivanjem vrijednosti varijabli može se upotrijebiti u bilo kojem slučaju, dok je upotreba debuggera uvjetovana postojanjem samog debuggera. U okviru Visual C++ ili Visual Studio .NET razvojnog okruženja uključen je i debugger pa je prema tome preporučljiva upotreba raznih pomagala koja debugger pruža.

[1] Funkcija getch nije dio ANSI standarda te se nalazi u biblioteci funkcija conio.h

ZADACI:

1. Pokrenite Visual Studio .NET razvojno okruženje. Otvorite novi projekt (*prema uputama iz prethodnih vježbi*).

- U svom projektu otvorite novu programsku datoteku i upišite slijedeći program koji bi trebao računati rješenja kvadratne jednadžbe, međutim sadrži neke pogreške (sintaktičke i logičke):

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

main() {
  float a,b,c;      /* koeficijenti kvadratne jednadžbe */
  float x1, x2;     /* rješenja kvadratne jednadžbe */
  float d;

  printf("Upisite koeficijente kvadratne jednadzbe (ax^2 + bx + c = 0):");
  printf("\na: ");
  scanf("%f",&a);
  printf("\nb: ")
  scanf("%f",&b);
  printf("\nc: ");
  scanf("%f",&c);

  if (a==0){
    printf("Jednadžba nije kvadratna!\n");
    exit(1); /* prekidam izvođenje programa */
  }
  /* računam diskriminantu */
  d = b*b - 4*c;
  if (d<0) {
    printf("Kvadratna jednadzba nema realna rjesenja!\n");
  }
  else if (d == 0) {
    x1 = -b/(2*a);
    printf("Kvadratna jednadzba ima samo jedno rjesenje: %6.2f\n", x1);

  else {
    x1 = sqrt(d) - b)/2*a;
    x2 = -(sqrt(d) + b)/2*a;
    printf("Rjesenja kvadratne jednadzbe su: %6.2f, %6.2f\n", x1, x2);
  }
  return 0;
}
```

- Pokrenite upisani program. Prilikom prevođenja prevodilac je dojavio nekoliko pogrešaka (sintaktičke pogreške). Popis tih pogrešaka možete pogledati u prozoru poruka (*TaskList*). Dvostrukim klikom lijeve tipke miša na neku od poruka o pogrešci kursor se pomiče na mjesto u programskom kodu gdje je prevodilac uočio pogrešku. Ispravite pogreške i ponovno pokrenite vaš program.
- Provjerite kako radi vaš program. Zadajte mu neku jednostavnu jednadžbu koju je lako provjeriti (npr. $x^2 – 3x +2 = 0$). Uvrstite dobivena rješenja u zadanu jednadžbu i provjerite da li ju zadovoljavaju. Pokušajte sada riješiti jednadžbu $2x^2 – 6x + 4 = 0$ koja bi morala imati ista rješenja. Sada izgleda da nešto nije u redu. Probajte debugger-om otkriti u čemu je stvar. Postavite prekidne točke na neka mjesta unutar programa i provjerite vrijednosti koje poprimaju razne varijable u programu (na

primjer provjerite vrijednost koju poprimi diskriminanta kvadratne jednadžbe). Nakon uklanjanja tzv. logičke pogreške ponovno prevedite i povežite program pa pokušajte sada izvesti program s istim koeficijentima kvadratne jednadžbe.

2. Unutar vaše radne okoline (unutar postojećeg *Solution-a*) otvorite novi projekt (*File->New Project*). Odaberite opciju *Add to solution*. Ako pogledate karticu *Solution Explorer* u prozoru projekta, vidjet ćete da u vašoj radnoj okolini imate dva projekta, od kojih je jedan ispisan masnim slovima. Taj je projekt trenutno aktivan. Možete pokrenuti samo onaj projekt koji je aktivan u tom trenutku (samo jedan projekt u nekom trenutku može biti aktivan). Aktivni projekt možete postaviti klikom na naziv projekta unutar *Solution Explorera* i odabirom *Set as StartUp Project* iz izbornika *Project* ili iz pop-up izbornika koji se pojavi desnim klikom miša na naziv projekta .

- U novom projektu otvorite novu programsku datoteku i u nju upišite slijedeći program koji računa sjecište dva pravca:

```
#include <stdio.h>

main() {
  float a1, b1, a2, b2;     /* koeficijenti eksplicitnih jednadžbi pravca*/
  float x, y;               /* koordinate sjecišta pravca */


  printf("Upisite jednadzbu prvog pravca: y = a1x + b1");
  printf("\na1: ");
  scanf("%f", &a1);
  printf("\nb1: ");
  scanf("%f", &b1);

  printf("\nUpisite jednadzbu drugog pravca: y = a2x + b2);
  printf("\na2: ");
  scanf("%f", &a2):
  printf("\nb2: ");
  scanf("%f", &b2);

  if (a1 == a2) {
    if (b1 == b2) {
      printf("\nPravci su identicni!!\n");
    }
    else {
      printf("\nPravci se ne sijeku!!\n");
    }
  }
  else {
    x = (b2 - b1)/(a2 - a1);
    y = a1*x + b2;
    printf("\nPravci se sjeku u tocki (%1.2f, %1.2f)\n", x, y);
  }
  return 0;
}
```

- uklonite sve sintaktičke i logičke pogreške iz tog programa
- testirajte program s različitim skupovima ulaznih podataka
- zadajte barem jedan skup podataka koji definira "skoro paralelne" pravce (sjecište im je negdje daleko, npr. a1=2, a2=2.00000005). Analizirajte dobivene rezultate i preuredite kriterij provjere paralelnosti tako da korisnik bude upozoren na probleme s preciznošću izračunavanja sjecišta.

Zadaci za vježbu

- Napišite u C-u program u kojem se neka zadana dužina `d` dijeli u omjeru 1:2:3. Ispisati originalnu dužinu i njene dijelove `d1`, `d2`, `d3`.
- Napišite u C-u program u kojem treba učitati neki peteroznamenkasti cijeli broj. Ukoliko broj nije peteroznamenkast uz odgovarajuću poruku završiti program. Za ispravno učitani broj treba ispisati učitani broj i u novom retku sve znamenke tog broja razmaknute sa po jednim razmakom.
- Napišite u C-u program u kojem treba učitati neki četveroznamenkasti cijeli broj. Ukoliko broj nije četveroznamenkast uz odgovarajuću poruku završiti program. Za ispravno učitani broj treba ispisati učitani broj i u novom retku dvije srednje znamenke tog broja.
- Napišite program koji će u smjeru kretanja kazaljki na satu zamijeniti numeričke vrijednosti triju varijabli x, y, z (npr. ako su pročitani x = 5, y = 2, z = 3 nakon izmjene bi morali imati vrijednosti x = 2, y = 3, z = 5).