Cvičenie 11

Štruktúrované premenné v prvkoch usporiadaného jednosmerného lineárneho zoznamu

Príklad 10.1 - Zoznam študentov s usporiadaným vkladaním a vyhľadávaním študentov

Zadanie
Vo vstupnom súbore *prvaci.txt* je nasledujúci zoznam študentov

```
Peter Jankovic A 12 6 1985
Jan Jankovic A 4 6 1985
Viliam Tell F 1 1 1987
Johanka Z_arku C 5 4 1238
Ferdinand Tell C 4 12 1986
0 0 0 0 0 0
```

V každom riadku tohto súboru sa nachádzajú údaje o jednom študentovi, jeho meno, priezvisko, jeho známka z matematiky, deň, mesiac a rok jeho narodenia. V poslednom riadku je 6 núl.

Vytvorte program v jazyku C++, ktorý z tohto súboru načíta všetkých študentov a vytvorí z nich lineárny zreťazený jednosmerný zoznam, ktorého jednotlivé prvky budú obsahovať údaje o jednom študentovi, umiestnené v 1 riadku súboru *prvaci.txt*. Program bude vkladať jednotlivé prvky do zoznamu **usporiadane** podľa nasledujúcich pravidiel:

1. zoznam bude usporiadaný **podľa priezvisk** študentov podľa abecedy (od A po Z)
2. ak je priezvisko študentov rovnaké, tak bude zoznam usporiadaný **podľa mien** študentov podľa abecedy (od A po Z)
3. ak budú aj mená študentov rovnaké, tak bude zoznam usporiadaný **podľa dátumu narodenia** študentov od najstaršieho študenta po najmladšieho študenta

Po načítaní posledného študenta a jeho **usporiadanom** vložení do zoznamu, program vypíše počet študentov v zozname a všetky prvky zoznamu, čiže údaje o všetkých načítaných študentoch.
Ďalej program ponúkne používateľovi voľbu vyhľadávacieho kritéria pre vyhľadávanie študentov v zozname. Keď používateľ vyberie vyhľadávacie kritérium a reťazec, ktorý sa bude vyhľadávať podľa zvoleného kritéria v príslušných vnútorných premenných jednotlivých prvkov zoznamu, program vykoná vyšpecifikované vyhľadávanie, jeho výsledky vypíše na konzolu a skončí.

Príklad výstupu programu:

```
--------vypis zoznamu----------
pocet studentov: 5

Jankovic Jan      (A) 4. 6. 1985
Jankovic Peter    (A) 12. 6. 1985
Tell Ferdinand    (C) 4. 12. 1986
Tell Viliam       (F) 1. 1. 1987
Z_arku Johanka    (C) 5. 4. 1238
-----------------------------------

podla coho chcete vyhladavat studenta? meno(0)/priezvisko(1): 1

zadajte retazec, ktory chcete vyhladat: tell

-----------------------------------
najdeni studenti:

Tell Ferdinand    (C) 4. 12. 1986
Tell Viliam       (F) 1. 1. 1987

pocet najdenych studentov: 2
-----------------------------------
```

Krátky rozbor úlohy

(Pozn.: Vo vytváranom programe využijeme viacero štruktúr a funkcií z riešenia úlohy 10.1 z cvičenia 10, takže ich opis sa môže v nasledujúcom texte opakovať. Pre úplnosť rozboru úlohy v tomto cvičení ich tu však uvádzame znovu.)

Na reprezentáciu dát si vytvoríme štruktúry *datum* a *TStudent*, ktorá nám popisujú údaje jedného študenta.

```
struct datum
{ int d, m, y;
};

struct TStudent
{ char meno[32], priezvisko[32];
  datum datum_narodenia;
  char znamka_M;
};
```

Zadanie úlohy máme riešiť pomocou lineárneho jednosmerného zoznamu. Vytvoríme si preto štruktúru *TPrvok*, ktorá bude reprezentovať jeden dátový prvok zoznamu.

```
struct TPrvok
{ TStudent student;
  TPrvok *dalsi;
};
```

Potrebujeme však ešte štruktúru, ktorá bude reprezentovať informačný záznam zoznamu.

```
struct TZoznam
{
  TPrvok *prvy, *posledny;
};
```

Vo vstupnom súbore každý riadok obsahuje údaje jedného študenta. Budeme naraz načítavať všetky údaje v riadku, pričom ich vložíme do príslušných vnútorných premenných štruktúrových premenných typu *TStudent*. Na ich načítavanie použijeme funkciu

*void **Nacitaj**(TStudent &sx, istream &vstup);*

Zaujímavý je jej druhý parameter. Je to referencia na vstupný prúd (objekt triedy *istream*). Reálnym druhým parametrom tejto funkcie môže byť vstavaný objekt *cin*, alebo vlastný dátový prúd spojený s diskovým súborom (náš prípad).

Pre výpis prvku môžeme použiť funkciu *void **VypisPrvok**(TPrvok *px)*, ktorá bude vypisovať obsah prvku, na ktorý ukazuje ukazovateľ *px*. To je ukazovateľ na štruktúrovú premennú typu *TPrvok*, ktorá obsahuje vnútorné premenné *TStudent student;* a *TPrvok *dalsi;*

Pre **usporiadané** vkladanie údajov o študentoch do prvkov zoznamu vytvoríme porovnávaciu funkciu *int **PorovnajS**(TStudent a, TStudent b)*, pomocou ktorej bude program porovnávať záznamy študentov, aby ich pomocou volania ďalšej funkcie *void **Vloz**(TZoznam &z, TStudent x)* vložil usporiadane, podľa zadaných kritérií, na správne miesto v zozname.
Vo funkcii *int **PorovnajS**(TStudent a, TStudent b)* porovnávame dvoch študentov, podľa zadaných kritérií. Funkcia vráti 1 ak je študent *a* v abecednom poradí na vyššej pozícii ako študent *b*, v opačnom prípade vráti hodnotu -1. Ak sú údaje rovnaké vráti 0.

Pri usporiadanom vkladaní údajov o študentoch do prvkov zoznamu rozoznávame 3 prípady vkladania:

- prvok obsahujúci údaje o 1 študentovi vkladáme na začiatok zoznamu,
- na koniec zoznamu,
- alebo niekde (na správne miesto) medzi existujúce prvky zoznamu.

Pri všetkých prípadoch vkladania musíme vždy zaktualizovať ukazovatele *prvy* a *posledny* uložené v štruktúrovej premennej *TZoznam &z*, aby tieto ukazovatele ukazovali na začiatok, resp. na koniec zmeneného zoznamu. Taktiež treba správne nastaviť ukazovatele *dalsi* štruktúrových premenných typu *TPrvok*, aby bol nový prvok správne pripojený do zoznamu.

Pre vyhľadávanie používateľom zadaného študenta vytvoríme funkciu
TPrvok **Najdi**(TZoznam z, char *hladane, TPrvok *start=NULL, int podla_p=1)*
, ktorá, po jej zavolaní, bude postupne prechádzať zoznamom od prvého prvku až pokiaľ hľadaný prvok nenájde. Ak bude funkcia úspešná, tak vráti ukazovateľ na prvok zoznamu, ktorý obsahuje hľadaného študenta.

Parametrami funkcie sú:

- *TZoznam z*, identifikátor zoznamu, v ktorom bude funkcia hľadať požadovaný prvok,
- *char *hladane* - ukazovateľ na reťazec obsahujúci hľadaný reťazec,
- *TPrvok *start=NULL* – ukazovateľ na prvok, od ktorého bude funkcia prehľadávať zoznam. Tento parameter sa dá využiť vtedy, ak funkcia *Najdi* nájde viacero výskytov hľadaného študenta. Tak používame tento parameter v opakovanom volaní funkcie *Najdi* vo funkcii *main*. Pri hľadaní ďalšieho študenta, teda pri ďalšom zavolaní funkcie *Najdi*, vložíme do parametra *start* ukazovateľ na posledný nájdený prvok zoznamu. Defaultovo obsahuje tento parameter hodnotu NULL, vtedy sa zoznam prehľadáva od začiatku.
- *int podla_p=1* – cez tento parameter prijme funkcia *Najdi* používateľom zvolené kritérium vyhľadávania. Ak bude *podla_p=1*, tak bude funkcia hľadať študenta alebo študentov podľa priezviska, ak bude *podla_p=1*, tak bude funkcia hľadať študenta alebo študentov podľa mena.

Nasledujúci zdrojový kód obsahuje tiež ďalšie funkcie
*void **Vypis**(TZoznam z);* (funkcia vypíše celý zoznam)
*void **ZmazPosledny**(TZoznam &z);* (funkcia zmaže zo zoznamu posledný prvok)
*void **Zmaz**(TZoznam &z);* (funkcia zmaže zoznam)

Možné riešenie

```
#include<fstream>
#include<string.h>
#include<iostream>
using namespace std;

//-----------------datove struktury-------------------//
struct datum
{ int d, m, y;
};

struct TStudent
{ char meno[32], priezvisko[32];
  datum datum_narodenia;
  char znamka_M;
};

struct TPrvok
{ TStudent student;
  TPrvok *dalsi;
};

struct TZoznam
{
  TPrvok *prvy, *posledny;
};

//------------------funkcie-----------------------------//
int JePrazdny(TZoznam z)
{
  return z.prvy == NULL;
}
```

```
int PorovnajS(TStudent a, TStudent b)
{ // podla priezviska, mena, datumu narodenia
  int porovnanie;
  porovnanie=strcmp(_strlwr(a.priezvisko), _strlwr(b.priezvisko));
  if(porovnanie!=0) return porovnanie;

  porovnanie=strcmp(_strlwr(a.meno), _strlwr(b.meno));
  if(porovnanie!=0) return porovnanie;

  long da,db;
  da=a.datum_narodenia.d+a.datum_narodenia.m*100+a.datum_narodenia.y*10000;
  db=b.datum_narodenia.d+b.datum_narodenia.m*100+b.datum_narodenia.y*10000;
  if(da>db) return 1;
  if(da<db) return -1;

  return 0;
}

void Vloz(TZoznam &z, TStudent x) // vlozi 1 prvok usporiadane do zoznamu
{
  TPrvok *novy = new TPrvok;
  novy->student=x;
  novy->dalsi = NULL;
  if(z.prvy == NULL) // prazdny zoznam
        z.prvy = z.posledny = novy;
  else if(PorovnajS(x, z.prvy->student)==-1) // (x < z.prvy->x) studenta treba vlozit na zaciatok zozn.
  {
        novy->dalsi = z.prvy;
        z.prvy = novy;
  }
  else if(PorovnajS(x, z.posledny->student)==1) // (x > z.posledny->x) studenta treba vlozit na koniec
                                                // zoznamu
  {
        z.posledny->dalsi = novy;
        z.posledny = novy;
  }
  else // inak studenta treba vlozit niekde medzi existujuce prvky zoznamu; v cykle while hladame,
  {                                                     // medzi ktore prvky ho treba vlozit
        TPrvok *p1 = z.prvy, *p2 = z.prvy->dalsi;
        while (PorovnajS(p2->student, x)==-1) // (p2->x < x)
        {
              p1 = p2; p2 = p2->dalsi;
        }
        p1->dalsi = novy;
        novy->dalsi = p2;
  }
}

TPrvok* Najdi(TZoznam z, char *hladane, TPrvok *start=NULL, int podla_p=1)
{ // ak parameter 'start' obsahuje inu hodnotu ako NULL, tak funkcia bude hladat od prvku, na ktory
  // ukazuje tento ukazovatel dalej, inak bude hladat od zaciatku zoznamu
  TPrvok *p;
  if(start==NULL)
        p = z.prvy;
  else
        p=start->dalsi;
  char hladane_pr[32];
  while (p != NULL)
  {
```

```
        if(podla_p)
          strcpy(hladane_pr, p->student.priezvisko); // aby sme si nezmenili povodne priezvisko
        else
          strcpy(hladane_pr, p->student.meno); // aby sme si nezmenili povodne meno
        if(strcmp(_strlwr(hladane_pr), _strlwr(hladane))==0)
          return p;
        p = p->dalsi;
  }
  return NULL;
}

void VypisPrvok(TPrvok *px)
{
  cout << px->student.priezvisko <<" "<<px->student.meno<<" \t("<<px->student.znamka_M<<") ";
  cout << px->student.datum_narodenia.d<<". "<<px->student.datum_narodenia.m<<". ";
  cout << px->student.datum_narodenia.y<<endl;
}

void Vypis(TZoznam z)
{
  if (JePrazdny(z)) return;
  TPrvok *p = z.prvy;
  while (p->dalsi != NULL)
  {
        VypisPrvok(p);
        p = p->dalsi;
  }
  VypisPrvok(p);
}

void ZmazPosledny(TZoznam &z)
{
  if(z.prvy == NULL) // prazdny zoznam
        return;
  TPrvok *p = z.prvy;
  if(p->dalsi == NULL) // je len jeden prvok v zozname
  {
        delete p;
        z.prvy = z.posledny = NULL;
        return;
  }
  // najdeme predposledny prvok
  while(p->dalsi->dalsi != NULL)
        p = p->dalsi;
  delete p->dalsi;
  p->dalsi = NULL;
  z.posledny = p;
}

void Zmaz(TZoznam &z)
{
  while (!JePrazdny(z))
  ZmazPosledny(z);
}

void Nacitaj(TStudent &s,istream &vstup)
{
  vstup>>s.meno>>s.priezvisko>>s.znamka_M;
  vstup>>s.datum_narodenia.d>>s.datum_narodenia.m>>s.datum_narodenia.y;
}
```

```
int main()
{
  TZoznam zoznam;
  zoznam.prvy = zoznam.posledny = NULL;
  TStudent s;
  int n=0;
  ifstream vstup;
  vstup.open("prvaci.txt");
  if(!vstup)
     cout<<"Nepodarilo sa otvorit vstupny subor";

  Nacitaj(s, vstup);
  Vloz(zoznam, s);
  while(s.datum_narodenia.d)
  {
          Nacitaj(s, vstup);
          if(s.datum_narodenia.d)
             Vloz(zoznam, s);
          n++;
  }
  vstup.close();

  cout<<"--------vypis zoznamu----------"<<endl;
  cout<<"pocet studentov: "<<n<<endl<<endl;
  Vypis(zoznam);
  cout<<"---------------------------------"<<endl;

  int kriterium;
  cout<<"\npodla coho chcete vyhladavat studenta? meno(0)/priezvisko(1): ";
  cin>>kriterium;
  cout<<endl<<"zadajte retazec, ktory chcete vyhladat: ";
  char przv[32];
  cin>>przv;
  cout<<"\n---------------------------------\nnajdeni studenti:\n"<<endl;
  TPrvok *najdeny=Najdi(zoznam, przv, NULL, kriterium); // hladame prvy vyskyt hladaneho studenta
  int pocet=0;
  while(najdeny)
  {       pocet++;
          if(najdeny)
              VypisPrvok(najdeny);
          najdeny=Najdi(zoznam, przv, najdeny, kriterium); // hladame dalsie vyskyty hladaneho
                                                           // studenta od posledneho najdeneho studenta
  }
  cout<<"\npocet najdenych studentov: "<<pocet<<endl;
  cout<<"---------------------------------"<<endl;

  Zmaz(zoznam);
  delete najdeny;
  return 0;
}
```

Príklad 11.2

Zadanie (na samostatnú prácu)
Zdrojový kód riešenia zadania príkladu 11.1 rozšírte a upravte tak, aby program vypisoval všetky prvky zoznamu, čiže údaje o všetkých načítaných študentoch **usporiadane podľa dátumu narodenia študentov** od najstaršieho študenta po najmladšieho študenta.
V zdrojovom kóde riešenia zadania príkladu 11.1 bude potrebné zmeniť a upraviť telo funkcie
int PorovnajS(TStudent a, TStudent b);
a prípadne vykonať ďalšie úpravy v zdrojovom kóde riešenia zadania príkladu 11.1.

Príklad výstupu upraveného programu, ktorý načítal dáta študentov zo súboru *prvaci.txt* uvedeného v zadaní príkladu 11.1:

```
--------vypis zoznamu----------
pocet studentov: 5

Z_arku Johanka  (C) 5. 4. 1238
Jankovic Jan    (A) 4. 6. 1985
Jankovic Peter  (A) 12. 6. 1985
Tell Ferdinand  (C) 4. 12. 1986
Tell Viliam     (F) 1. 1. 1987
-----------------------------------

podla coho chcete vyhladavat studenta? meno(0)/priezvisko(1): 1

zadajte retazec, ktory chcete vyhladat: tell

-----------------------------------
najdeni studenti:

Tell Ferdinand  (C) 4. 12. 1986
Tell Viliam     (F) 1. 1. 1987

pocet najdenych studentov: 2
-----------------------------------
```