Gymnázium Brno, Vídeňská, příspěvková organizace

Ročníkový projekt
pro Programování

STATISTIKY PLAVCŮ

19. května 2023

Roman Grmela, 4.G

Obsah

Klíčová slova

soubor, pole struktur, ročníkový projekt, funkce, program

Citace

Roman Grmela: *STATISTIKY PLAVCŮ*, Ročníkový projekt, Brno, Gymnázium Brno, Vídeňská, příspěvková organizace, 2023.

Abstrakt

Tato práce se zabývá detailním zpracováním dat o plavcích obsažených v textovém souboru v programovacím jazyce C. Práce obsahuje implementaci funkce pro vyhledávání, mazání, třídění, řazení dat a mnoho dalších. Můžeme to brát jako „statistiku“, ze které lze vyčíst při promyšleném užití mnoho zajímavých informací. Hlavním cílem projektu je usnadnění a zefektivnění práce všem plavcům při vyhledávání informací o ostatních soupeřích a poskytnout tak uživateli relevantní, přehledné a přesné výsledky.

Prohlášení

Prohlašuji, že jsem tuto práci vytvořil samostatně a poctivě jsem uvedl všechny zdroje, ze kterých jsem vycházel.

.....

Roman Grmela

5. června 2023

Kapitola 1 Zadání

Vytvoř ročníkovou práci v programovacím jazyce C v předmětu informatika a výpočetní technika. Nejprve si vyber téma a následně ho vypracuj. Měl bys pracovat se soubory, poli struktur, knihovnami stdbool.h a string.h. Nutná součást projektu bude vyhledávání dat, filtrace, úprava, ale také mazání dat. Na konci vypracuj zálohu souboru.

Kapitola 2 Analýza a návrh

2.1 Specifikace zadání

Vytvoř ročníkovou práci z předmětu informatika a výpočetní technika. Nejprve si vyber téma na ročníkovou práci a vytvoř ji v jazyce C. Měl bys pracovat se soubory, poli struktur, knihovnami stdbool.h a string.h. Nutná součást projektu bude vyhledávání dat, filtrace, úprava a mazání dat. Na konci vypracuj také zálohu souboru.

Důležité součásti projektu budou: načtení dat ze souboru do pole struktur (soubor je vyplněný aspoň 11 položkami, každá položka má 4 vlastnosti a jedna z toho je číselná), zápis dat z pole struktur do souboru (dvě funkce). Při načítání do pole kontroluj, zda nepřesáhneš velikost pole. Pokud přesáhneš, další položky z pole nenačítej a vypiš hlášku, že pole nemá dostatečnou kapacitu.

Další funkce pro řazení pole aspoň podle dvou vlastností (jedna číselná, jedna textová) - výpis na obrazovku. Zkontroluj, že ti ostatní funkce pracují bez chyby. Měla by tam být funkce pro souhrn (počet, min, max, sum) a jednoduchý filtr.

Vyhledávání podle určité položky je další důležitá součást projektu. Také doplňte mazání položky. Mělo by fungovat přidání položky a úprava položky.

2.2 Analýza problému

Uživatel si po spuštění programu vybere jeden ze dvou souborů, které jsou vypsány na obrazovce a se kterým z nich má v plánu pracovat. Program se bude věnovat textovým souborům/dokumentům. Správným vstupem bude tedy soubor s příponou „.txt“ (např. plavci1.txt nebo plavci2.txt). Při špatném zadání souboru program vypíše hlášku: „Soubor se nepodařilo otevřít“ a program tímto končí. V opačném případě program pokračuje.

Po vybrání souboru se uživateli ukáže menu, ze kterého si vypsáním čísla (pouze nabízeného) vybere, co se má s daty v poli struktur stát. Při zadání špatného čísla se vypíše hláška: „Netrefil ses, zkus to znovu.“ Ovšem program tímto nekončí. Uživatel dostane další šanci pro zadání čísla z nabídky. Ke každému číslu je přiřazen písemný úkon. Všechny dílčí úkony jsou přehledně zpracovány funkcí „proved“ – do této funkce posílám volbu a postupně se v této funkci volají ostatní funkce pro provedení zvolené akce. Uživatel si takto vybírá, dokud nestiskne volbu UKONČENÍ PROGRAMU – pod číslem 0. V tento moment se vybraný soubor po všech změnách uživatelem uloží pod názvem „plavci_zaloha.“

Kapitola 3 Popis řešení

3.1 Prostředí

Celý projekt je vypracován v programovacím jazyce C. Práci jsem tvořil v aplikaci Code::Blocks na mém PC. Toto prostředí je mi známé i z praktických vyučovacích hodin IVT, bylo to pro mě tedy pohodlné. Pokaždé, když jsem dodělal určitou část projektu, jsem si můj rozpracovaný projekt uložil do Repl.it. Má práce vyžaduje textový soubor, jenž by měl být připraven ve složce projektu. Je to nutnost, protože celý program pracuje pouze s textovými soubory.

3.2 Struktura – nový datový typ

V souboru se nachází minimálně jedenáct položek složených z: jména plavce (TEXT), příjmení plavce (TEXT), ročníku plavce (INTEGER), klubu plavce (TEXT), osobního rekordu na 50m volný způsob za závodní sezónu 2022 (REAL) a můžeme si všimnout také položky obsazeno (BOOL) – ta bude, např. při přidávání záznamu, pro výsledek klíčová. Z těchto položek vytvořím nový datový typ: TPLAVEC.

3.3 Funkce Main

Po spuštění programu a otevření souboru vytvořím pole struktur o maximální velikosti MAX (stanovenou na hodnotu 100) a všechny buňky pole označím jako neobsazené. Poté vytvořím proměnnou „n“, která bude značit aktuální velikost pole (počet obsazených buněk pole). Pomocí této proměnné zavolám funkci pro načtení hodnot do pole. Načítám takto, dokud je místo v poli a dokud je co načítat ze vstupního souboru. Pro přehlednost se na obrazovku vypíše počet úspěšně načtených struktur. Následně uživatel zadává volbu, kde je vždy u konkrétního čísla uvedeno, co se má stát. Tímto způsobem uživatel zadává volbu, dokud nezadá 0 – konec programu. Pokud bude zadáno jiné číslo, než navrhuje funkce menu, vypíše se hláška, že se uživatel netrefil. Program ovšem tímto nekončí, je možnost zadat číslo, které je navrhováno. Po každém úkonu se obrazovka pro přehlednost maže. Když bude zadáno číslo 0, program tím končí

Obrázek 3.1: Logo Gymnázia Brno, Vídeňská.

a vytvoří se záloha souboru po všech operacích uživatelem. Poslední věcí ve funkci „main“ je zavření všech souborů.

3.4 Ostatní funkce

3.4.1 Funkce init

Tato funkce se postará o inicializaci pole struktur. Pouze se postará o to, aby byly všechny buňky pole označeny jako neobsazené. Očekává následující vstupy: pole a jeho aktuální velikost. Tato funkce nic nevrací, pouze inicializuje pole struktur.

3.4.2 Funkce pro načtení a výpis

Funkce pro načtení očekává jako vstupní parametry soubor a pole a vrací počet úspěšně načtených hodnot do pole. Samozřejmě je zde kontrolováno, zda nepřesáhneme maximální velikost pole.

Funkce pro výpis vypíše všechny obsazené buňky pole. Většinou je do této funkce posílán parametr „stdout“, díky čemuž se data vypíší na obrazovku. Na konci programu se do zmíněné funkce pošle výstupní soubor pro vytvoření zálohy.

3.4.3 Řadící funkce

Při řazení dat v souboru jsem mohl využít mnoho algoritmů. Nelze jednoznačně tvrdit, který je nejlepší. Každý má své klady i zápory. Nakonec jsem při řazení dat podle rychlosti a podle příjmení použil algoritmy vnitřního řazení a to konkrétně SelectionSort a InsertSort, protože mi přišly kvůli menšímu množství dat v souboru optimální. Jejich implementace byla rovněž jednoduchá. Při většímu množství dat by se dalo polemizovat o volbě nejlepšího algoritmu. Určitě by byl brán v potaz také QuickSort. Ten je totiž v průměru pro náhodná data suverénně nejrychlejší. Oba algoritmy jsem upravil pro moje pole struktur. Funkce pro seřazení podle příjmení počítá s tím, že se v souboru vyskytují i plavci se stejným příjmením a seřadí to v případě nutnosti i podle jména.

3.4.4 Funkce typu bool

Do projektu jsem také naprogramoval funkce datového typu bool, které otestují, zda jsou data podle nějakého klíče seřazena. Tohle jsem provedl z důvodu 100% jistoty o správnosti řazení pole struktur. Je zde také funkce typu bool, která zjistí, zda je plavec nadprůměrný. Zjistí to tak, že spočítá průměrný osobní rekord a následně porovná průměrný rekord a rekord zadaného plavce. Samozřejmě se zde počítá i s tím, že předtím uživatel nějaké plavce smazal. Ti se pak do průměrného času nepočítají. U všeho se berou v potaz pouze obsazené buňky pole struktur.

3.4.5 Funkce na filtrace

Dále, např. u funkce, která vypíše plavce z klubu „KomBr“ jsem použil knihovnu programu <string.h> a také funkci „strcmp“. Je zde i pomocná proměnná, která odhalí počet plavců z komety. Pokud je tato proměnná prázdná, vypíše se, že zde nikdo z komety není. Další funkce, jež se věnuje filtraci dat vypíše počet plavců, kteří mají osobní rekord pod 23 vteřin. A zároveň vypíše i ty konkrétní plavce na obrazovku.

3.4.6 Funkce na vyhledávání

Nezbytnou pomocnou funkcí programu je funkce „najdi plavce“, která vyhledá pozici plavce podle jména a příjmení. Právě tato je v programu využita vícekrát a pro konečný výsledek nezbytná. Např. při přidání nové osoby se zavolá tato funkce, a pokud se takový plavec v seznamu už nachází, nedovolí program uživateli přidat stejnou osobu znovu. Při úpravě již existujícího plavce je zmíněná funkce rovněž volána (mnoho funkcí v programu spolu souvisí).

3.4.7 Srovnávací funkce

Pro srovnání plavce se v mé ročníkové práci vyskytují 2 velmi podobné funkce. Ta první vypíše rychlejší plavce než plavec zadaný. A ta druhá vypíše ty pomalejší.

3.4.8 Funkce pro mazání

Při funkci, která se věnuje mazání osoby jsem použil logické mazání. Ve struktuře se nachází položka „obsazeno“. Při mazání pouze nastavím obsazeno na „false“. Všechny ostatní funkce jsou tomuto přizpůsobené. V mé práci si můžeme všimnout dvou funkcí, které plavce mažou. Ta první je jednoznačná – pouze smaže plavce podle zadaného jména a příjmení. Ovšem pak je tu funkce, která maže plavce podle klubu. Tak se zde vyskytuje následující komplikace: V souboru mohou být plavci, kteří závodí za stejný klub. A tak abychom smazali plavce ze zadaného klubu, kterého opravdu chceme, program vždy vypisuje po jednom plavce, kteří za daný klub trénují. Pokaždé je zde na výběr, jestli toho plavce chceme opravdu smazat, a pokud ano, tak jestli budeme mít zájem smazat i dalšího plavce z toho klubu. Bude-li uživatel stále zadávat, že chce mazat dál, ale žádný plavec se tam už vyskytovat nebude, mazání tímto končí. Vždy se pracuje pouze s buňkami pole, které jsou obsazené. Ty buňky, které obsazeny „nejsou“ se nijak do výsledných operací nepromítnou.

3.4.9 Funkce pro uložení nových dat

Funkce, která ukládá novou strukturu do pole určitě testuje, jestli jsme nepřesáhli maximální velikost pole. Pokud by se tato velikost přesáhla, uživateli bude tato možnost odepřena.

3.5 Chybně zadané hodnoty uživatelem

Na začátku se program zeptá: „S jakým souborem chceš pracovat?“ Pokud uživatel nezadá soubor, který je možno použít, program končí. V opačném případě pokračuje a zobrazí se menu. Ve funkci menu je každé číselné hodnotě přiřazena akce, která se provede. Zadá-li uživatel číslo, které menu neumožňuje, vypíše se hláška, že se uživatel netrefil.

Po správně zadané hodnotě z menu dávají některé možnosti uživateli prostor k napsání vlastního textu. K tomu se pojí několik problémů. Co když uživatel zadá např. plavce, který v souboru není? Tyto problémy jsem se snažil co nejlépe vyřešit.
Vyřešil jsem to následovně. Chce-li uživatel přidat osobu, která se už v seznamu vyskytuje, vypíše se hláška, že se zde už zadaný plavec nachází a operace se ukončí. Stejně jsem si poradil i s mazáním podle klubu. Pokud se žádný plavec ze zadaného klubu v souboru nenachází, ukončí se větev switche a uživatel pokračuje s úpravou vstupního souboru.

3.6 Usnadnění pro čtenáře

Kód je rozdělen do logických bloků, přičemž všechny funkce jsou nad funkcí main. Každá volaná funkce se nachází nad danou funkcí, která ji volá.
Pro zjednodušení pochopení pro čtenáře kódu jsem ke každé funkci přidal popis, co dělá. Rovněž názvy všech funkcí jsou stručné a výstižné. V celém projektu můžeme narazit na mnoho komentářů pro upřesnění apod. Např. V prvním komentáři vidíme zkrácenou verzi souboru pro ušetření času čtenáři. Ovšem je zde i hodně upřesňujících informacích třeba o jednotlivých proměnných (např. u proměnné „kde“ je poznámka: na jakém indexu leží nejrychlejší plavec?). Tímto způsobem velmi urychlím práci všem, kteří do kódu nahlédnou.

3.7 Zdroje

Ve funkci main jsem se v ročníkové práci inspiroval řešením mého učitele informatiky Mgr. Šatného při „cvičení 14 - vyhledávání“, kdy se posílá hodnota volby do funkce „proved“. Zmíněné řešení mi přišlo nejpřehlednější. V projektu jsem také použit funkci „init“, která byla použita ve „cvičení 16“, při realizaci hashovací tabulky. Při řazení jsem využil řadící algoritmy, které jsem měl uložené z předešlých hodin IVT praxe a následně je upravil tak, aby šly použít na moje pole struktur. Když jsem si nemohl na něco vzpomenout (např. jak vypsat řetězec na určitý počet míst), použil jsem prezentace Davida Martínka. Čerpal jsem rovněž z literatury Učebnice jazyka C od Pavla Herouta.

3.8 Citace

Učebnice jazyka C (1. díl). České Budějovice: Kopp. ISBN 80-7232-220-6., Pavel Herout

MARTÍNEK, David. *Prezentace Jazyk C - funkce*.

MARTÍNEK, David. *Prezentace Jazyk C - strukturované datové typy*.

Kapitola 4 Závěr

4.1 Závěrečný komentář

Tento projekt slouží jako zdroj informací a statistik pro plavce, které zajímají jejich časy, srovnání s ostatními atd.

Jak kód, tak finální grafiku programu jsem se snažil co nejvíce zpřehlednit, aby bylo pro všechny jednodušší se v programu orientovat. Všechny části zadání jsem splnil a ve finální verzi je má práce funkční. Myslím, že se mi kvalitně povedlo vše, co jsem v projektu měl v plánu zrealizovat. Na mnoho částí jsem obzvlášť hrdý. Za sebe mohu tvrdit, že část, jež provádí mazání plavce podle klubu je vytvořena perfektně. Tuhle část jsem mnohokrát předělával. Pak jsem se vcítil do role uživatele a předělal to tak, aby to bylo připraveno na všechny možnosti mazání. Snažil jsem se maximálně vzít v úvahu také chybně zadané hodnoty či názvy uživatelem. (např. když uživatel zadá plavce, jenž se v datech nevyskytuje, vypíše se hláška, že se zde plavec nenachází).

V práci jsem zrealizoval mnoho svých nápadů, ale je určitě hodně věcí, které by se daly do projektu přidat a práci tak vylepšit. Mám na mysli např. přidání do struktury datum zaplavání osobního rekordu či město, kde plavec podal osobní maximum a podle těchto dat plavce setřídit do skupin. (např. kdo si zaplaval rekord na Mistrovství ČR konaném v Plzni 11. 11. 2022.) Ještě mě napadlo, že bych do projektu zadal hodnotu limitu pro výběr do reprezentace a vypsat např. plavce, kteří limit mají, či seřadit podle toho, kdo je k limitu nejblíže. V mé práci se nijak nebere v potaz, když by chtěl uživatel přidat novou osobu a zadal by více znaků, než je velikost řetězce. Tomuto problému bych se také mohl v budoucnu věnovat. V práci je sice opatřeno, když uživatel zadá jiné číslo, než nabízí funkce menu, ale není zde vyřešen následující problém. Tím je právě zadání např. písmene místo čísla. S touto situací se program vypořádat neumí. Takže je stále možné moji práci jistým způsobem vylepšit.

Chci říct, že mě velmi mile překvapilo, jak můj projekt na konci vypadá. Pracoval jsem na tom průběžně a s upřímnou radostí. Využil jsem snad všechny znalosti z jazyka C, které jsem za své studium nasbíral. Za zmínku stojí také mé obrovské zlepšení v programování za posledního půl roku, které mi tvoření tohoto projektu poskytlo. Povedlo se mi zpracovat vše, co se po mě v zadání očekávalo a přidal také hodně dalších věcí navíc. Svá očekávání jsem tedy splnil.

Po několika měsících tvrdé práce můžu říct, že jsem se svou prací velmi spokojen.

Na úplný závěr chci poděkovat, že jste si našel/našla čas na přečtení tohoto textu. Doufám že Vám přinesl něco nového a užitečného. Pokud máte jakékoliv dotazy nebo chcete diskutovat o tomto tématu více, neváhejte mě kontaktovat prostřednictvím mailu (romasg1@seznam.cz).

4.2 Příloha

4.2.1 Ukázka kódu

V dokumentaci bych rád ukázal část kódu, za kterou jsem nejvíc pyšný. Jedná se konkrétně o mazání položky podle klubu. Jelikož se v souboru může vyskytovat více plavců ze stejného klubu, vyřešil jsem to tak, že se budou postupně ukazovat na obrazovce plavci, kteří jsou právě z daného klubu. Uživatel si vybírá, zda chce plavce smazat – a pokud ano, tak jestli chce smazat i dalšího.

```
case 13:
        printf("Zadej klub plavce, ktereho chces odstranit: ");
        scanf("%14s", klub);
        printf("--------------------------------------------------------------------------\n");
        index=najdiPodleKlubu(pole, n, klub, 0);
       if(index == -1){
        printf("Zadny plavec z klubu %s zde neni...\n", klub);
        break;
       }

       printf("Prvni plavec z klubu %s je -----> %s %s ", pole[index].klub, pole[index].jmeno,
pole[index].prijmeni);


        volba2=2;
        while(index<n && index != -1 && volba2 != 1 && (volba2 == 2 || volba2 == 3)/*kdyby
nahodou nekdo zadal jinou volbu nez 1, 2 nebo 3...*/){
        printf("\n\nJsi si jisty, ze chces smazat plavce %s %s z klubu %s?\n\n", pole[index].jmeno,
pole[index].prijmeni, pole[index].klub);
        printf("1.....ANO, SMAZAT A KONEC\n2.....ANO, SMAZAT A CHCI SMAZAT
DALSIHO\n3.....NE, NEMAZAT A PREJIT NA DALSIHO PLAVCE\n\nVolba: ");
        scanf("%d", &volba2);
        printf("--------------------------------------------------------------------------\n");

        switch(volba2){
        case 1:
        vymazPlavce(pole, n, index);
        printf("Byla odstranena osoba %s %s\n", pole[index].jmeno, pole[index].prijmeni);
        printf("--------------------------------------------------------------------------\n");
        printf("Koncim s mazanim...\n");
```

```
        break;

      case 2:
        vymazPlavce(pole, n, index);
      printf("Byla odstranena osoba %s %s\n", pole[index].jmeno, pole[index].prijmeni);
      printf("--------------------------------------------------------------------------\n");

      index=najdiPodleKlubu(pole, n, klub, index+1);
      if(index == -1){
      printf("Dalsi plavec z klubu %s zde neni...\n", klub);
        break;
        }

        printf("Dalsi plavec z klubu %s je -----> %s %s ", pole[index].klub, pole[index].jmeno,
pole[index].prijmeni);
        break;


      case 3:
        index=najdiPodleKlubu(pole, n, klub, index+1);
      if(index == -1){
      printf("Dalsi plavec z klubu %s zde neni...\n", klub);
        break;
        }

        printf("Dalsi plavec z klubu %s je -----> %s %s ", pole[index].klub, pole[index].jmeno,
pole[index].prijmeni);
        break;


      default:
        printf("Smula, netrefil ses!\n");
        break;
```

4.2.2 Testovaná data

Pro úplnost bych rád přiložil data ze vstupního souboru na kterých jsem celý program testoval.

```
Daniel Gracik 2004 SCPAP 22.47
Krystof Smehlik 2006 PKKBr 24.78
Radim Svarc 2002 KomBr 22.13
Martin Molis 2006 KomBr 25.58
Roman Grmela 2006 KomBr 23.82
Miroslav Knedla 2005 Zlin 22.56
```

Jan Bruna 2006 SlPl 23.43
Ondrej Grus 2006 BiJa 25.21
Krystof Bursa 2006 ZlPK 23.45
Matej Masa 2004 KomBr 24.19
Vaclav Siroky 2000 KomBr 24.56
Ondrej Slavik 2006 KPSOs 22.65
Samuel Hubscher 2007 KPSOs 24.62
Ondrej Havrlant 2006 KomBr 25.01
Jan Reka 2006 KomBr 25.24
Jakub Krischke 2006 SlzPK 22.39