Gymnázium Brno, Vídeňská, příspěvková organizace


Pololetní projekt
pro Programování

# Databáze knih

17. dubna 2024

David Horák, 4.G

# Databáze knih

## Prohlášení

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

.....
David Horák
17. dubna 2024

# Obsah

# Kapitola 1

# Zadání

Vytvořte databázi libovolného podmětu v jazyce C, se kterou bude váš program pracovat. Databáze musí obsahovat alespoň 20 záznamů. Každý záznam musí mít alespoň 3 atributy, z nichž alespoň 1 musí být číselný.

Program by měl umožnit uživateli následující: Vložit nový záznam, opravit existující záznam, vymazat záznam, vytisknout celou databázi na obrazovku, seřadit databázi, vyhledat záznam v databázi a vytvořit souhrn dat v databázi na základě zvoleného témata (napři. průměrná výška lidí, celkový počet lidí, atd.).

# Kapitola 2

# Návrh řešení

## 2.1 Struktura databáze

Jelikož není v zadání specifikován formát databáze a počítáme s tím, že s databází bude pracovat pouze náš program, můžeme si vytvořit vlastní formát, který by vyhovoval naším potřebám. Zde se hodí obyčejný textový soubor. Do něj budeme ukládat jednotlivé záznamy na samostatný řádek, jednotlivé atributy záznamů budou oddělíme čárkou.

## 2.2 Struktura programu

Zadání požaduje velice málo dat v databázi (alespoň 20 prvků se třemi atributy). Můžeme počítat s tím, že smíme celou databázi načíst do paměti počítače uživatele. Poté uživatel pomocí několika funkcí může manipulovat s touto databází, než ji zase uloží do dříve zmíněného souboru.

## 2.3 Ovládání programu

Program má umožnit uživateli přístup k několika funkcím pro manipulaci databáze. Na tohle vytvoříme menu, ve kterém budou jednotlivé položky. Položky tohoto menu by obsahovali název funkce a korespondující písmeno pro zpuštění téhleté funkce.

## 2.4 Nedostatky návrhu

Tento návrh má několik nedostatků. Atributy záznamu nemohou mít v sobě čárky (kvůli rozdělení atributů v souboru) a databáze je také teoreticky omezena velikostí paměti/swap souboru uživatelského počítače.

# Kapitola 3

# Řešení

## 3.1 Databáze

Jako podmět mé databáze jsem si vybral knihy. Každý záznam má následující atributy: Název, autor, žánr, rok vydání a počet stran. Seznam knih byl náhodně vybrán z těch, na které jsem si vzpomněl.

Informace o atributách knihy jsem našel na databazeknih.cz(CITACE). Rozhodl jsem napsat všechny textové atributy v angličtině, jelikož program neumí zacházet s unicode písmenama.

## 3.2 Struktura kódu

### Základní datové struktury

Nejzákladnější datové struktury tohoto programu jsou definovány v `databazeK.h`. Struktura `KNIHA` má 6 atributů. Tři 255 písmen dlouhé řetězce (název, autor a žánr), 2 celá čísla (počet stran a rok vydání) a bool `jeSmazano` který říká, jestli je struktura smazaná.

Struktura `KDATABAZE` má dva atributy: Ukazatel `prvky` na strukturu `KNIHA` (pracuje s ním jako s polem) a celé číslo `pocetZaznamu`, který ukládá aktuální počet prvků na který ukazatel může ukazovat.

### Pomocné funkce

Pomocné funkce využívá většina ostatních funkcí, hlavně pro zisk vstupů od uživatele. Jestli je požadovaný vstup velmi malý, tak tyhle funkce také vyčistí vstup. Jsou 4 pomocné funkce:

- Funkce `dostanString` získa od uživatele 255 písmen dlouhý řetězec. Pokud uživatel zadá řetězec, který obsahuje čárky, přemění čárky na mezery (kvůli formátu databáze).
- Funkce `dostanInt` získa od uživatele jedno celé číslo. Pokud uživatel zadá více čísel nebo jiný špatný vstup, funkce se ho zeptá znova, dokud nezadá správně.
- Funkce `ziskVolby` získa od uživatele jedno písmeno. Jestli je písmeno malé, vrátí velké.
- Funkce `stiskneteEnter` donutí uživatele stisknout enter. Je potřebný, jelikož menu vždy první vymaže celou konzoli. Tahle funkce zabrání vrácení k menu hned po konci volání funkce.

## Menu

Menu programu je definováno v `menicko.c` a `menicko.h`. Pro vytvoření menu se zavolá funkce `menuVolba`, která má následující parametry: ukazatel na strukturu `KDATABAZE`, řetězec `nadpisek`, řetěžec `podpisek`, bool `neulozeneZmeny` a pole struktur `MPOLOZKA`.

Struktura `MPOLOZKA` je jedna položka menu. Skládá se z řetězce `text`, písmena `pismeno` a ukazatelem na funkci `funkce`.

když se zavolá funkce `menuVolba`, funkce vymaže konzoli a vytiskne `nadpisek` a `podpisek`, případně hvězdičku jestli bool `neulozeneZmeny` se rovná 1. poté vytiskné jednotlivé položky a zavolá `ziskVolby`. Po zisku písmena je porovná s písmenami položek menu, jestli se rovná, zavolá funkci té položky. Jestli získané písmeno je Q, přestane cyklus a skončí funkce. Jestli se ničemu nerovná nic se nestane a cyklus zase začne.

## Hlavní funkce

Hlavní funkce, které mají jeden parametr ukazatele na `KDATABAZE` a jsou v nabídce menu jsou následující:

- Funkce `otevriDatabazi` se optá uživatele, zda chce otevřít existující databázi nebo vytvořit novou. Přijme cestu na soubor od uživatele a podle dřívějšího rozhodnutí se pokusí ho otevřít s jiným oprávněním. Jestli to nepujde, zeptá se o novou cestu. Cesta se uloží do globální proměnné `cesta` se kterým budou pracovat jiné funkce. Jestliže už soubor existuje, načte z něj pomocí druhotné funkce `nacteniDatabaze` databázi.
- Funkce `tiskDatabaze` vytiskne všechny položky v poli knih v databázi.
- Funkce `Vyhledavani` přijme od uživatele řetězec s čísly 1 až 5, který určuje jakými atributy chce uživatel hledat (každé číslo koresponduje jednému atributu). Poté načte uživatelem vybrané hodnoty do vlastní `KNIHA` struktury . Pak lineárně vyhledává rovnající se knihy v databázi (samozřejmě jen těmi hodnotami, co specifikoval uživatel).
- Funkce `novyZaznam` realokuje pole `KNIHA` položek v `KDATABAZI` o jeden víc, načte od uživatele nové hodnoty a zvětší číslo textttpocetZaznamu o jeden.
- Funkce `opravaZaznamu` přijme číslo od uživatele, které koresponduje s položkou pole knih. Jestliže kniha existuje, načte od uživatele nové hodnoty knihy.
- Funkce `smazaniObnovaZaznamu` přijme číslo knihy u které ma změnit bool `jeSmazano` na opačné znaménko. Položky, u kterých `jeSmazano` se rovná 1, nebudou při uložení do souboru napsány.
- Funkce `seradDatabazi` přijme od uživatele řetězec čísel 1 až 5, které korespondují k jednotlivým atributům knih, podle kterých bude řadit (čísla napsaná dřív budou mít prioritu). Také se zeptá uživatele, zda chce seřadit soubor vzestupně nebo sestupně. Poté zavolá funkce, které jsou definované v `sortDatabaze.h` a `sortDatabaze.c`. Algoritmus k seřazení je quicksort. Funkce pro porovnání, zda je položka menší nebo větší k té, se kterou ji chceme porovnávat bere jako parametr řetězec daných atributů.
- Funkce `souhrn` vytiskne na obrazovku celkový počet knih a stran a průmerný počet stran v databázi.
- Funkce `ulozZmeny` se zeptá uživatele jestli chce uložit danou databázi do stejného souboru nebo do jiného. Jestli si vybere do jiného, funkce se ho zeptá o novou cestu. Poté vypíše všechny položky databáze do souboru, pokud nejsou označené jako smazané. Nakonec znovu načte z toho souboru prvky zpět do pole.

### Sekundární funkce

Funkce, které volají hlavní funkce. Narozdíl od vedlejších funkcí mají většinou využití jenom u jedné funkce.

- Funkce `QSsortDat`, `QSsortRek`, `QSswap`, `QssortRozdel` a `JeVetsi` jsou všechny podfunkce funkce `seradDatabazi` a jsou definovány v souboru `sortDatabaze.h` a `sortDatabaze.c`. Jedná se o algoritmus quicksort.
- Funkce `tiskMenu` je podfunkce pro funkci `menuVolba`. Vytiskne položky menu.
- Funkce `vyhledat` a `rovnaSe` jsou podfunkce funkce `Vyhledavani`. Funkce `Vyhledavani` hlavně se stará o správný vstup od uživatele, `vyhledat` prochází databázi a `rovnaSe` vrací 0 pokud se dvě položky rovnají.
- Funkce `tiskZaznamu` vytiskne jeden záznam na obrazovku. Tuhle funkci používá hlavně funkce `tiskDatabaze`.
- Funkce `nacteniDatabaze` načte databázi ze souboru do paměti. Používá ji funkce `otevriDatabazi` a `ulozZmeny`. Vrací jiné číslo než 0 když nastala chyba.

## 3.3 Použité technologie a programovací jazyky

Program byl napsán v jazyce C jak bylo v zadání. Byl použit IDE Code::Blocks a tím i programy gcc, gdb a jiné. Pro manuální úpravy databáze a psaní dokumentace byl použit textový editor Neovim. Pro správu verzí byl použit git a projekt byl uchován na platformě github.

## 3.4 Známé omezení a problémy

Některé známé problémy byly již zmíněny v kapitole návrhu. Jednotlivé textové položky databáze nesmí obsahovat v sobě žádné čárky a unicode písmena. Teoreticky je program omezen velikostí paměti počítače. Také nemohou textové položky záznamu obsahovat víc jak 255 písmen.