Gymnázium Brno, Vídeňská, příspěvková organizace

Ročníkový projekt
pro Programování

Tréninkový deník

15. března 2025 Roman Grmela, 6.G

Abstrakt

Tato práce se zabývá vývojem tréninkového deníku, aplikací pro zaznamenávání a správu tréninkových jednotek. Umožňuje evidenci aktivit, jejich délky a intenzity, nabízí statistiky a práci se soubory. Cílem je usnadnit sledování tréninkového pokroku. Dokumentace popisuje návrh, implementaci a funkce aplikace.

Klíčová slova

dynamická datová struktura, modul, paměť, soubor, funkce

Citace

Roman Grmela: *Tréninkový deník*, Ročníkový projekt, Brno, Gymnázium Brno, Vídeňská, příspěvková organizace, 2025.

Tréninkový deník

Prohlášení

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

.....

Roman Grmela

16. března 2025

Obsah

Kapitola 1 Zadání

Zadání této práce je volné a umožňuje realizaci libovolného programátorského projektu, který zahrnuje implementaci algoritmů a programovací logiku. Není omezeno konkrétním jazykem ani technologií, což dává prostor pro kreativitu a volbu optimálních nástrojů. Důležitým aspektem je však vytvoření funkční aplikace s jasně definovaným účelem a praktickým využitím. V rámci této práce jsem se rozhodl vytvořit tréninkový deník – aplikace pro správu a evidenci tréninkových aktivit.

Kapitola 2 Analýza a návrh

2.1 Specifikace zadání a cíle

Vytvoř aplikaci pro tréninkový deník, která umožní uživateli evidovat tréninkové aktivity, včetně data, typu tréninku, délky a dalších parametrů. Navrhni uživatelské rozhraní pro zadávání tréninků, které bude intuitivní a přehledné, aby uživatelé snadno zadávali potřebné údaje. Aplikace by měla umožnit uživateli přidávat nové tréninkové záznamy, upravovat existující záznamy a mazat nepotřebné záznamy, čímž zajistí flexibilitu a efektivní správu tréninkových aktivit.

Zajisti validaci vstupních dat, aby uživatel mohl zadávat pouze platné hodnoty, například datum ve správném formátu nebo délku tréninku v minutách. Aplikace by měla upozornit uživatele na chyby při zadávání a zabránit ukládání neplatných údajů. Dále vytvoř funkci pro zobrazení statistik a analýzu tréninkových dat, která bude uživateli umožňovat zobrazení informací, jako je průměrná délka tréninků, celkový čas strávený trénováním nebo vyhodnocení dosažených cílů, čímž přispěje k lepšímu přehledu o pokroku.

Připrav si několik testovacích souborů a otestuj funkčnost aplikace v různých scénářích, aby bylo zajištěno, že všechny funkce fungují správně. Otestuj přidávání, úpravy a mazání tréninkových záznamů, validaci vstupních dat a zobrazení statistik. Navrhni aplikaci tak, aby byla dobře rozšiřitelná, s ohledem na možné přidání nových funkcí, jako je například sledování různých typů tréninků nebo propojení s dalšími aplikacemi pro sledování fyzické aktivity. Dbej na modularitu a přehlednost kódu pro zajištění snadné údržby a vylepšení aplikace v budoucnu.

Cílem aplikace je nabídnout uživatelům nástroj pro efektivní správu tréninkových aktivit, který bude snadno použitelný a přehledný. Umožní sledování pokroku, poskytování statistik a analýzu výkonu, čímž usnadní plánování a optimalizaci tréninků. Aplikace zjednoduší zaznamenávání tréninků, čímž ušetří čas. Bude flexibilní pro různé uživatele a umožní snadné rozšíření o nové funkce, nebo tréninkové plány. Hlavním cílem je také vyzkoušet práci s dynamicky alokovanou pamětí a ukazateli jako dynamickou datovou strukturu.

2.2 Analýza problémů

Při vývoji aplikace realizující tréninkový deník budu muset řešit následující problémy. Jelikož budu tvořit program, který zpracuje vstupní údaje ze souboru, kde budou uloženy data ve vhodném formátu, vyplývá z toho několik problémů. Jakým způsobem bude uživatel zadávat soubory se vstupy? Jak přesně má vypadat formát souboru? Co je správný vstup, a co ne? Co se stane, když zadá uživatel vstup chybně a co když bude soubor ve špatném formátu? S prací se soubory se pojí také bezpečnostní otázky, jako je ochrana dat před neúmyslným poškozením nebo ztrátou. Důležité bude zajistit pravidelné zálohování souborů.

Při vývoji aplikace pro tréninkový deník by měl být kladen důraz na efektivitu a optimalizaci výkonu, zejména s ohledem na práci s rostoucími objemy dat. Aplikace bude muset efektivně zpracovávat a ukládat stále více tréninkových záznamů bez ztráty výkonu nebo zpomalení. Jedním z klíčových aspektů bude optimalizace způsobu uložení a načítání dat, například použitím vhodných datových struktur pro rychlý přístup k informacím. Dále bude nutné provádět pravidelnou správu paměti, aby se zabránilo jejímu přetížení nebo neefektivnímu využívání.

Dále bude důležitým zaměřením uživatelská zkušenost a rozhraní. Aplikace bude muset efektivně reagovat na různé vstupy od uživatelů, včetně chyb při zadávání nebo nedostatečných infor-

mací, a to prostřednictvím jasných a srozumitelných chybových hlášení, která uživateli pomohou opravit jeho vstupy. Dále bude kladeno důraz na to, aby uživatelské rozhraní bylo intuitivní, přehledné a snadno ovladatelné, a to i pro uživatele, kteří nemají zkušenosti s podobnými aplikacemi.

Bude vhodné navrhnout aplikaci tak, aby byla dobře strukturovaná a snadno rozšiřitelná o nové funkce nebo moduly v budoucnosti. K tomu bude důležité využít principy modularity, které umožní oddělit jednotlivé části aplikace, jako je správa tréninkových záznamů, statistiky nebo validace dat. Tímto způsobem bude možné přidávat nové funkce bez nutnosti zásadních změn v celkové struktuře aplikace.

Další problém vyplývající ze zadání je efektivní správa paměti, zejména s ohledem na možnost práce s velkými objemy dat. Při rostoucím počtu tréninkových záznamů je nezbytné zajistit, aby paměť byla správně uvolňována a aby nedocházelo k jejímu úniku, což by mohlo vést ke zpomalení aplikace nebo k selhání při dlouhodobém používání. Správné řízení paměti bude klíčové pro zachování výkonu aplikace a zajištění její stability.

2.3 Návrh řešení

2.3.1 Grafické rozhraní, programovací jazyk

Při rozhodování o výběru programovacího jazyka pro tento projekt je důležité zvážit několik faktorů, včetně požadavků na funkčnost aplikace, dostupnosti knihoven pro specifické úkoly, a také na to, co chci jako vývojář získat z realizace tohoto projektu. Jedním z častých jazyků pro vývoj aplikací s grafickým uživatelským rozhraním (GUI) je například Java. Java poskytuje mnoho nástrojů pro tvorbu grafických rozhraní, jako je JavaFX nebo Swing, které umožňují rychlý vývoj moderního a intuitivního GUI. Java také zjednodušuje správu paměti díky automatickému garbage collectoru a nabízí širokou podporu pro práci s databázemi a síťovými aplikacemi.

Nicméně, přestože Java nabízí rychlý vývoj s GUI, navrhuji implementaci v jazyce C. Tento jazyk mi poskytuje větší kontrolu nad tím, jak aplikace funguje na nižší úrovni, zejména co se týče správy paměti. Můj hlavní cíl je zaměřit se na práci s dynamickými datovými strukturami (DDS), jako jsou zásobníky nebo fronty, a mít možnost ovládat alokaci a dealokaci paměti pomocí ukazatelů. V C mám možnost více experimentovat s těmito nízkoúrovňovými koncepty a lépe pochopit, jak fungují v reálných aplikacích.

Měl jsem už zkušenosti s DDS, a to mě na programování baví, protože umožňuje detailní práci s pamětí a datovými strukturami. Tento projekt pro mě představuje ideální příležitost, jak si tyto dovednosti vyzkoušet v praxi. Zatímco GUI není pro tento konkrétní projekt zásadní, zaměřím se na funkční část aplikace a efektivitu algoritmů. Výběr jazyka C mi umožní nejen efektivně spravovat paměť, ale i vyvinout algoritmy pro zpracování tréninkových dat, které se budou přímo vztahovat k požadavkům zadání.

2.3.2 Tréninkový deník jako DDS

Tréninkový deník plánuji implementovat jako dynamickou datovou strukturu (DDS seznam), která bude umožňovat efektivní práci s tréninkovými záznamy. U této struktury bude vytvořena zastřešující struktura pro uchovávání informací, která bude obsahovat ukazatele na jednotlivé prvky. Tyto ukazatele budou propojeny a umožní dynamické přidávání nových prvků za běhu programu. Tento přístup je ideální i pro práci s rozsáhlými daty, protože DDS mění svou velikost v průběhu životního cyklu aplikace.

Dynamická datová struktura je pro tento typ aplikace velmi vhodná, protože umožňuje ukládání dat, jejichž počet není předem známý. V případě tréninkového deníku, kde počet záznamů může růst podle toho, jak uživatel zaznamenává nové tréninky, je tento flexibilní přístup nezbytný. Díky DDS nebudu muset předem alokovat velký blok paměti a aplikace bude schopná efektivně reagovat na měnící se požadavky bez přetížení paměti nebo zpomalení výkonu. To vše umožní aplikaci snadnou škálovatelnost i při rostoucím objemu tréninkových záznamů.

2.3.3 Moduly

Dále plánuji strukturovat aplikaci do několika modulů pro zajištění přehlednosti a snadné rozšiřitelnosti. Prvním modulem bude hlavičkový soubor, který bude obsahovat deklarace funkcí, jež může uživatel použít. Tento modul bude sloužit jako rozhraní mezi aplikací a funkcemi, které umožní interakci s tréninkovým deníkem, jako například přidávání, úprava, mazání záznamů a generování statistik.

Druhým modulem bude zdrojový soubor, který by mohl obsahovat implementaci těchto funkcí. V tomto modulu budou detailně popsány a implementovány všechny operace, které aplikace vykonává nad tréninkovými záznamy, včetně dynamického přidávání záznamů do DDS, jejich úpravy a mazání. Kromě toho zde budou zahrnuty i další skrytější funkce, které zajišťují správu paměti, optimalizaci výkonu a validaci vstupních dat.

Poslední částí aplikace bude hlavní soubor (main), kde budou využity funkce z předchozích modulů. Zde také implementuji různé algoritmy, které budou operovat nad tréninkovým deníkem, například analýzu statistik nebo porovnání záznamů. Tento modul bude sloužit jako centrální bod pro zpracování všech dat a jejich výstup uživateli. Tímto způsobem bude aplikace dobře organizovaná a rozšiřitelná o nové moduly nebo funkce, které budou potřeba v budoucnu.

2.3.4 Práce se soubory a validace vstupních dat

Navrhuji, aby uživatel měl možnost zadat název souboru, který obsahuje tréninkové záznamy, včetně přípony (např. "data.txt"). Tento soubor bude obsahovat textová data, která budou následně zpracována aplikací. Uživatel bude moci tento název souboru zadat ručně prostřednictvím uživatelského rozhraní aplikace. Pro snazší práci by bylo vhodné umožnit zadání cesty k souboru, aby uživatel nemusel soubor umisťovat do stejného adresáře, kde běží aplikace.

Pro řešení problémů s chybnými údaji, které uživatel může zadat (např. špatně formátovaný název souboru, nesprávná přípona souboru, nebo neexistující soubor), navrhuji implementovat funkce pro validaci všech informací, které uživatel zadá. Tyto funkce by měly kontrolovat, zda název souboru obsahuje správnou příponu (např. ".txt"), zda soubor existuje v dané cestě a zda není prázdný. Pokud bude některý z těchto kontrolních bodů porušen, aplikace by měla uživateli zobrazit informativní chybovou hlášku a umožnit mu opravit chybu.

S prací se soubory jsou spojeny i bezpečnostní otázky, jako je ochrana dat před neúmyslným poškozením nebo ztrátou. Abychom minimalizovali riziko ztráty dat, plánujeme implementovat funkci pro pravidelné zálohování souborů. Uživatel by měl mít možnost nastavit, jak často chce zálohovat soubor, případně by bylo možné zálohování provádět automaticky při každém úspěšném zápisu dat do souboru. To zajistí, že při případném pádu aplikace nebo neúmyslném poškození souboru, bude možné obnovit poslední platnou verzi dat.

Kapitola 3 Popis řešení

3.1 Vývojové prostředí

Pro vývoj aplikace pro tréninkový deník jsem použil integrované vývojové prostředí **Code::Blocks**, které je dobře známé z hodin programování. Code::Blocks je vhodné pro jazyk C a poskytuje přehledné a intuitivní rozhraní pro psaní, kompilování a ladění kódu. Nabízí užitečné funkce jako zvýraznění syntaxe, automatické doplňování kódu a integrovaný debugger, což usnadňuje práci a zrychluje vývoj. Pro tento projekt bylo ideální, protože umožňuje efektivní práci s jazykem C a poskytuje všechny nástroje potřebné pro rychlý vývoj.

3.2 Struktury – nové datové typy

V této podkapitole popíši struktury nových datových typů, které jsem použil pro reprezentaci tréninkové jednotky a deníku. Jsou definovány pomocí struktur, které uchovávají informace o tréninkových aktivitách a stavu deníku.

3.2.1 Struktura TJednotka

První struktura se jmenuje TJednotka a obsahuje následující pole:

- datum (char[11]): Uchovává datum tréninkové jednotky ve formátu YYYY-MM-DD.
- aktivita (char[15]): Uchovává typ tréninkové aktivity (např. plavání, posilovna, apod.).
- delkaTrvani (int): Uchovává délku trvání tréninku v minutách.
- narocnost (int): Uchovává hodnotu RPE (Rating of Perceived Exertion), což je osobní měřítko náročnosti tréninku na škále 1-10.
- dalsi (TJednotka*): Ukazatel na další prvek v seznamu, což umožňuje vytváření dynamické datové struktury pro uchovávání více tréninkových jednotek (pomocí propojeného seznamu).

Tato struktura bude klíčovým prvkem pro práci s dynamickými datovými strukturami v aplikaci. Díky použití ukazatelů je možné efektivně přidávat, upravovat a mazat jednotlivé tréninkové jednotky, čímž aplikace získá flexibilitu při zpracování a správě tréninkových záznamů.

3.2.2 Struktura TDenik

Další struktura, kterou jsem použil, je TDenik. Tato struktura reprezentuje samotný tréninkový deník, který obsahuje seznam všech tréninkových jednotek. Struktura TDenik obsahuje následující proměnné:

- prvni (TJednotka*): Ukazatel na první tréninkovou jednotku v deníku. Tento ukazatel umožňuje přístup k prvnímu záznamu v seznamu.
- posledni (TJednotka*): Ukazatel na poslední tréninkovou jednotku v deníku. Tento ukazatel usnadňuje přidávání nových jednotek na konec seznamu bez nutnosti procházet celý seznam.
- pocetJednotek (int): Uchovává počet tréninkových jednotek v deníku. Tento údaj umožňuje efektivní práci s dynamickým seznamem, zejména při přidávání nebo mazání jednotek.

Struktura TDenik spolu se strukturou TJednotka tvoří základ pro správu tréninkových záznamů. Pomocí propojených seznamů (ukazatelů) je možné dynamicky přidávat, mazat a upravovat tréninkové jednotky. Tato struktura zajišťuje, že aplikace bude efektivně pracovat s tréninkovými záznamy a umožní jejich správu v dynamickém prostředí.

3.3 Moduly

Pro implementaci aplikace pro tréninkový deník jsem zvolil tři moduly, které se zaměřují na různé aspekty programu. Tyto moduly jsou: main.c, TDenik.h a Tdenik.c.

Modul main.c je zodpovědný za volání již hotových funkcí a práci s tréninkovým deníkem. Jeho hlavní funkcí je spojit všechny části aplikace a zajistit její správný chod. Tento soubor slouží jako hlavní součást aplikace, která vykonává operace podle požadavků uživatele.

TDenik.h je hlavičkový soubor, který deklaruje funkce používané v rámci programu. Tento soubor poskytuje definice funkcí pro manipulaci s tréninkovým deníkem, jako je přidávání nebo mazání tréninkových jednotek. Jeho úkolem je zajistit, aby všechny potřebné funkce byly dostupné v jiných modulech.

Modul TDenik.c obsahuje implementace funkcí deklarovaných v TDenik.h a také další pomocné skryté funkce, které podporují práci s tréninkovými jednotkami a jejich správu v deníku. Tento soubor je základem pro manipulaci s daty tréninkového deníku a obsahuje veškerou logiku pro přidávání, mazání nebo úpravu tréninkových záznamů.

Volba těchto modulů je běžná při práci s dynamickými datovými typy, jako jsou zásobník, fronta a seznam. Rozdělení kódu do modulů umožňuje lepší strukturalizaci programu a usnadňuje jeho rozšiřování a údržbu. Každý modul se specializuje na konkrétní úkol, což podporuje přehlednost a efektivitu při vývoji. Nyní se podíváme na jednotlivé moduly podrobněji

3.3.1 Modul main.c

Tento modul je hlavní součástí aplikace pro tréninkový deník. Je zodpovědný za interakci s uživatelem, správu deníku a volání funkcí definovaných v dalších modulech, jako je „TDenik.h“ a „Tdenik.c“. Hlavním cílem tohoto modulu je poskytnout textové uživatelské rozhraní, které umožňuje uživatelské akce, jako je přidávání tréninkových jednotek, mazání jednotek, zobrazení statistik a uložení dat do souboru.

Po začátku programu je inicializován tréninkový deník pomocí funkce „denikInicializuj()“. Pokud dojde k chybě při alokaci paměti pro deník, program se ukončí s chybovým hlášením. V opačném případě pokračuje načítáním nebo vytvářením prázdného deníku. Uživatel je vyzván, aby zvolil, zda chce pokračovat s prázdným deníkem, nebo zda chce načíst data ze souboru. Pokud uživatel zvolí načtení dat ze souboru, program se pokusí načíst soubor, který obsahuje tréninkové jednotky, a ověřit jejich platnost.

Po načtení souboru nebo inicializaci prázdného deníku program přistupuje k hlavní smyčce, která poskytuje uživatelské menu. Menu obsahuje několik možností, jako je zobrazení nápovědy, výpis deníku, přidání nové tréninkové jednotky, odstranění jednotky, zobrazení statistik a uložení deníku do souboru. Program čeká na uživatelský vstup a podle volby provádí odpovídající akci.

Při volbě přidání tréninkové jednotky program požaduje od uživatele datum, aktivitu, délku trvání a náročnost tréninku. Poté provede kontrolu zadaných hodnot a, pokud jsou správné, přidá novou tréninkovou jednotku do deníku. Při volbě odstranění jednotky program požaduje datum tréninku, který má být odstraněn, a následně jednotku z deníku vymaže. Pokud uživatel zvolí možnost zobrazení statistik, program provede analýzu všech tréninkových jednotek v deníku a vypíše souhrnné statistiky, jako je celkový počet tréninků, průměrná délka tréninků, průměrná náročnost, nejdelší trénink, nejtežší trénink a statistiky podle aktivit.

Pokud uživatel zvolí možnost uložit deník do souboru, program požaduje název souboru a uloží stav deníku do tohoto souboru. Před ukončením programu je vždy prováděno uvolnění alokované paměti deníku, což je zajištěno voláním funkce „denikUvolni()“.

Tento modul je příkladem jednoduchého a efektivního způsobu, jak spravovat a interagovat s dynamickými datovými strukturami, jako je propojený seznam tréninkových jednotek. Program využívá funkce z dalších modulů k efektivní manipulaci s těmito datovými strukturami, přičemž uživatel má možnost snadno provádět různé akce, jako je přidávání, mazání a analýza tréninkových jednotek. Modul „main.c“ tak tvoří základní rozhraní pro uživatele a interakci s tréninkovým deníkem.

3.3.2 Modul Tdenik.h

Modul „TDenik.h“ je hlavičkový soubor, který definuje struktury a funkce pro správu tréninkového deníku. Tento modul je zodpovědný za definici datových struktur, které uchovávají informace o jednotlivých tréninkových jednotkách a celkový deník. Díky těmto funkcím může program efektivně manipulovat s tréninkovými jednotkami a udržovat tréninkový deník aktuální a organizovaný.

V modulu jsou definovány dvě hlavní struktury. „TJednotka“ představuje jednotlivou tréninkovou jednotku a obsahuje informace jako datum, typ aktivity, délku trvání tréninku, úroveň náročnosti (RPE) a ukazatel na další jednotku v seznamu. „TDenik“ je struktura, která reprezentuje celý tréninkový deník. Udržuje ukazatele na první a poslední jednotku v seznamu a také počet tréninkových jednotek, které jsou v deníku uloženy.

Mezi funkcemi definovanými v tomto modulu jsou klíčové funkce pro správu tréninkového deníku. Funkce „denikInicializuj“ inicializuje nový tréninkový deník a vrací ukazatel na něj. Funkce „denikUvolni“ slouží k uvolnění paměti alokované pro deník a všechny tréninkové jednotky v něm obsažené. Funkce „denikJePrazdny“ kontroluje, zda je deník prázdný, tedy zda neobsahuje žádné tréninkové jednotky.

Další důležitou funkcí je „denikPridatJednotku“, která přidává novou tréninkovou jednotku na konec seznamu v deníku. Před přidáním jednotky do seznamu je však provedena kontrola, zda je jednotka validní. Funkce „denikOdstranitJednotku“ umožňuje odstranit tréninkovou jednotku na základě zadaného data. K dispozici je i funkce „denikTiskni“, která vytiskne všechny tréninkové jednotky v deníku na obrazovku.

Pokud uživatel chce uložit svůj deník do souboru, může použít funkci „denikTiskniDoSouboru“, která zapíše všechny tréninkové jednotky do souboru. Naopak pro načítání tréninkových jednotek z externího souboru je určena funkce „denikNactiZeSouboru“, která načte data a přidá je do deníku.

Tento modul poskytuje základní funkcionalitu pro práci s tréninkovými deníky, včetně přidávání, mazání, tisknutí a načítání tréninkových jednotek. Je navržen tak, aby usnadnil správu tréninkového deníku a umožnil jeho uchování mezi jednotlivými spuštěními aplikace.

3.3.3 Modul Tdenik.c - implementace funkcí

V této části dokumentace se zaměřím na popis implementace hlavního modulu „TDenik.c“, který poskytuje konkrétní implementaci funkcí pro správu tréninkového deníku definovaných v modulu „Tdenik.h“.

Funkce „**denikInicializuj**“ je zodpovědná za inicializaci nové instance tréninkového deníku. Při jejím zavolání dojde k alokaci paměti pro novou strukturu „TDenik“. Po úspěšné alokaci jsou ukazatele na první a poslední tréninkovou jednotku v deníku nastaveny na NULL, což znamená, že deník je na začátku prázdný. Počet tréninkových jednotek je nastaven na 0, což signalizuje, že v deníku nejsou žádné jednotky.

Funkce „**denikUvolni**“ je nezbytná pro správnou správu paměti, která byla alokována na haldě pro tréninkový deník a všechny jeho tréninkové jednotky. Tato funkce prochází všechny tréninkové jednotky uložené v deníku a postupně uvolňuje jejich paměť pomocí funkce free. Jakmile jsou uvolněny všechny tréninkové jednotky, funkce nakonec uvolní i paměť alokovanou pro samotnou strukturu deníku. Tímto způsobem je zajištěna prevence úniku paměti, což je klíčové pro stabilitu aplikace.

Funkce „**denikTiskni**“ prochází všechny tréninkové jednotky v tréninkovém deníku a vypisuje jejich informace na obrazovku. Výstup obsahuje datum, aktivitu, délku trvání tréninku a úroveň náročnosti (RPE). Před samotným výpisem jednotek je vytištěna hlavička pro lepší přehlednost. Pokud je deník prázdný, funkce informuje uživatele a neprovádí žádný výstup. Funkce navíc používá systémový příkaz cls pro vyčištění obrazovky, což je platformně závislé a může fungovat pouze na některých operačních systémech, například na Windows. Podobnou funkci má funkce „**denikTiskniDoSouboru**“, která navíc očekává jako parametr název souboru, do kterého aktuální stav deníku uložit a data tak zálohovat.

Funkce „**denikNactiZeSouboru**“ slouží k načítání tréninkových jednotek ze souboru a jejich následnému přidání do tréninkového deníku. Otevře soubor se zadaným názvem a načte tréninkové jednotky v předem definovaném formátu, který obsahuje datum (ve formátu YYYY-MM-DD), název aktivity (max. 14 znaků), délku trvání tréninku (v minutách) a úroveň náročnosti (hodnota mezi 1 a 10). Po načtení každé jednotky je provedena její validace pomocí funkce „_otestujJednotku“. Pokud je jednotka platná, vytvoří se nová tréninková jednotka pomocí funkce „_vytvorJednotku“ a přidá se do deníku pomocí funkce „denikPridatJednotku“. Pokud se jednotku nepodaří vytvořit nebo je neplatná, je vypsána chybová hláška. Po dokončení načítání a přidání jednotek je soubor automaticky uzavřen. Funkce využívá interní funkce pro validaci, vytváření jednotek a přidávání jednotek do deníku.

Funkce „**denikOdstranitJednotku**“ slouží k odstranění tréninkové jednotky z deníku na základě zadaného data a názvu aktivity. Funkce prochází všechny tréninkové jednotky v deníku a hledá jednotky, které odpovídají zadanému datu. Po zobrazení nalezených jednotek vyzve uživatele k zadání názvu aktivity, kterou chce odstranit. Pokud uživatel zadá název aktivity, který odpovídá některé z nalezených jednotek, funkce ji odstraní ze seznamu. Pokud žádná jednotka pro zadané datum nebo aktivitu neexistuje, funkce vypíše chybovou hlášku. Pokud uživatel zadá neplatnou aktivitu, funkce mu umožní opravit vstup nebo zrušit operaci. Funkce kontroluje, zda je deník prázdný, a zda existují tréninkové jednotky pro dané datum. Po odstranění jednotky je seznam aktualizován a paměť pro odstraněnou jednotku je uvolněna.

Funkce „**_vytvorJednotku**“ vytváří nové tréninkové jednotky. Nejdříve alokuje paměť pro novou strukturu typu „TJednotka“, což je objekt reprezentující tréninkovou jednotku, která bude obsahovat informace o konkrétní aktivitě. Po alokaci paměti funkce nastaví atributy nové tréninkové jednotky podle parametrů, které jsou předány při jejím volání. Na závěr funkce nastaví ukazatel dalsi na NULL, protože nová jednotka bude na konci seznamu. Pokud se během alokace paměti pro novou tréninkovou jednotku objeví chyba (např. není dostatek paměti), funkce vrátí NULL. Pokud je alokace úspěšná, funkce vrátí ukazatel na nově vytvořenou tréninkovou jednotku. P. S. Tato funkce se volá pouze v případě, že jsou atributy ověřeny a validovány.

Funkce „**denikPridatJednotku**“ přidává novou tréninkovou jednotku do deníku a zajišťuje správnou organizaci záznamů. Před samotným přidáním již proběhla validace atributů jednotky, což znamená, že všechna data jsou správná a relevantní. Funkce nejprve ověří, zda je deník prázdný. Pokud ano, nová jednotka se stane prvním i posledním záznamem. V případě, že deník již obsahuje jiné jednotky, nová jednotka je připojena na jeho konec a aktualizuje se ukazatel na poslední záznam. Nakonec se zvýší počet evidovaných jednotek, čímž se udržuje konzistence struktury deníku.

Funkce „**_otestujJednotku“** prověřuje, zda jsou všechny atributy tréninkové jednotky správně zadané a odpovídají stanoveným pravidlům. Nejprve se zkontroluje formát data, aby odpovídal požadovanému tvaru. Následuje ověření aktivity, která musí patřit mezi předem definované možnosti. Dále se posuzuje délka trvání, která nesmí být nulová nebo záporná. Nakonec proběhne kontrola náročnosti, která musí spadat do platného rozmezí hodnot. Pokud jakýkoli z těchto parametrů nevyhovuje, funkce vypíše odpovídající chybovou hlášku a vrátí hodnotu false. V případě úspěšného splnění všech podmínek vrací true, čímž potvrzuje, že tréninková jednotka je validní a může být dále zpracována – tj. přidána do dynamického seznamu.

Přišlo mi vhodné implementovat také pomocnou funkci „**denikJePrazdny“**, která slouží k rychlé kontrole, zda tréninkový deník obsahuje nějaké záznamy. Funkce nejprve ověří, zda ukazatel na deník není NULL, a následně zkontroluje, zda první prvek seznamu existuje. Pokud je deník prázdný, funkce vrátí true, což může být užitečné při rozhodování o dalších operacích, například při pokusu o odstranění nebo výpis jednotek. Naopak pokud deník obsahuje alespoň jednu tréninkovou jednotku, funkce vrátí false, což znamená, že lze s daty dál pracovat.

3.4 Dokumentační komentáře

Při dokumentování kódu jsem dbal na přehlednost a srozumitelnost jednotlivých funkcí pomocí dokumentačních komentářů. Každou funkci jsem opatřil strukturovaným komentářem ve formátu Doxygen, který umožňuje automatickou generaci dokumentace. Každý komentář začíná klíčovým slovem @brief, kde jsem stručně popsal účel funkce. Následně jsem doplnil podrobnější vysvětlení jejího chování a přidal popisy jednotlivých parametrů pomocí @param. Pokud funkce vrací hodnotu, uvedl jsem také @return, aby bylo jasné, co funkce vrací a za jakých podmínek. Díky tomuto přístupu jsem zajistil, že kód je snadno čitelný a pochopitelný i pro ostatní vývojáře. Navíc to umožňuje generování dokumentace bez nutnosti ručního vypisování popisů jednotlivých funkcí.

3.5 Možné rozšíření

Díky modulárnímu návrhu a dobře strukturovanému kódu je aplikace snadno rozšiřitelná. Dynamický seznam, který uchovává tréninkové jednotky, obsahuje všechny potřebné funkce pro správu dat, a proto není nutné zásadně měnit jeho implementaci. Možná rozšíření se tak nabízejí především na úrovni hlavního modulu, kde by bylo možné přidat další algoritmy pro zpracování dat. Například by šlo implementovat řazení tréninkových jednotek podle určitého kritéria (délka trvání, náročnost, datum) nebo přidat pokročilejší filtry pro vyhledávání konkrétních aktivit.

Dalším krokem by mohlo být vylepšení uživatelského rozhraní. Aktuální verze je textová, což je pro základní správu deníku dostačující, ale pokud bych chtěl aplikaci posunout dál, dalo by se zvážit grafické rozhraní. Nejjednodušší by bylo využít Javu a její objektový přístup, který práci s GUI

značně usnadňuje. V C by bylo řešení složitější, protože bych musel ručně spravovat okna, vstupy a výstupy.

Nicméně mým hlavním cílem bylo procvičit si práci s pamětí na větším projektu a práci s ukazateli, což je v C klíčové. V jazycích jako Java se o správu paměti stará garbage collector, což usnadňuje vývoj, ale zároveň skrývá důležité principy správy dynamických struktur. Proto jsem zvolil přístup, který mi umožnil prohloubit pochopení těchto základních konceptů.

Kapitola 4 Závěr

V této práci jsem se zaměřil na vytvoření tréninkového deníku pro správu tréninkových jednotek sportovce, přičemž jsem použil dynamický seznam (DDS), což mi umožnilo efektivně spravovat jednotlivé tréninkové jednotky. Cílem bylo nejen prohloubit mé znalosti v práci s dynamickou pamětí a ukázat, jak fungují datové struktury na nízké úrovni, ale také si osvojit různé techniky, které se používají při návrhu a implementaci software.

V průběhu vývoje se mi podařilo implementovat funkce pro ověření platnosti dat, přidávání tréninkových jednotek do deníku, jejich odstranění, validaci atributů a mnoho dalších důležitých operací, které jsou součástí každého dobře navrženého tréninkového deníku.

Měl jsem v plánu do projektu přidat více algoritmů pro manipulaci s tréninkovým deníkem, například třídění jednotek podle různých kritérií, ale z časových důvodů jsem musel některé z těchto funkcí redukovat. Bylo to z důvodu mého nasazení v jiných aktivitách, jako je příprava na koncerty v klavíru, příprava na MČR v plavání, kde jsem měl každodenní tréninky, a samozřejmě příprava na maturitu z dalších předmětů. I přesto jsem spokojený s tím, co jsem naprogramoval a doufám, že aplikace bude především pro sportovce přínosná.

Na závěr bych chtěl poděkovat za možnost pracovat na tomto projektu. Byla to cenná zkušenost, která mi pomohla nejen prohloubit moje technické dovednosti, ale i porozumění tomu, jak efektivně navrhovat a implementovat software. S tímto dobrým základem věřím, že další pokročilé jazyky a techniky pro mě nebudou problém.

Citace použité literatrury

Dynamická alokace paměti. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: https://cs.wikipedia.org/wiki/Dynamick%C3%A1_alokace_pam%C4%9Bti. [cit. 2025-03-16].

HEROUT, Pavel. *Učebnice jazyka C*. 2021. Kopp, 2021. ISBN 978-80-7232-383-8.

You will never ask about pointers again after watching this video. Online. Https://www.youtube.-com/watch?v=2ybLD6_2gKM. 2023. Dostupné z: https://www.youtube.com/watch?v=2ybLD6_2gKM. [cit. 2025-03-16].

C typedef. Online. Https://www.geeksforgeeks.org/. 2025. Dostupné z: https://www.geeksforgeeks.org/typedef-in-c/. [cit. 2025-03-16].