Termín odevzdání:	**26.12.2022 11:59:59**
Pozdní odevzdání s penalizací:	**31.12.2022 23:59:59** (Penále za pozdní odevzdání: 100.0000 %)
Hodnocení:	**5.5000**
Max. hodnocení:	**5.0000** (bez bonusů)
Odevzdaná řešení:	1 / 20 Volné pokusy + 10 Penalizované pokusy (-10 % penalizace za každé odevzdání)
Nápovědy:	0 / 2 Volné nápovědy + 2 Penalizované nápovědy (-10 % penalizace za každou nápovědu)

Úkolem je realizovat sadu funkcí (ne celý program, jen funkce) pro práci se spojovými seznamy. Funkce umožní seznam snadno vytvářet, mazat a řadit. Úloha je rozšířením jednoduššího zadání, přidává možnost vlastního komparátoru. Doporučujeme nejprve vyřešit zadání jednodušší a to pak rozšiřovat.

Pří řešení problému se setkáte s následujícími deklaracemi:

`TITEM`

Tato struktura reprezentuje prvek jednosměrně zřetězeného spojového seznamu. Deklarace struktury je pevně daná v testovacím prostředí, Vaše implementace bude tuto nezměněnou deklaraci používat (nemůže/nesmí ji měnit). Ve struktuře jsou následující složky:

- `m_Next` ukazatel na další prvek ve spojovém seznamu, hodnota `NULL` pro poslední prvek,
- `m_Name` řetězec s názvem prvku seznamu,
- `m_Secret` tajná data přičleněná k této položce seznamu. Vaše implementace s touto složkou nepotřebuje pracovat a nesmí ji měnit (výjimkou je inicializace ve funkci `newItem`).

```
TITEM * newItem ( const char * name, TITEM * next )
```

Tato funkce vytvoří nový záznam `TITEM`. Paměť pro strukturu budete alokovat dynamicky. Funkce navíc zkopíruje jméno a odkaz na další prvek z parametrů do složek `m_Next` a `m_Name`. Složku `m_Secret` funkce vyplní nulovými bajty ('\0'). Návratovou hodnotou funkce je ukazatel na takto vytvořenou a inicializovanou struktury `TITEM`. Implementace funkce je Vaším úkolem.

```
void freeList ( TITEM * l )
```

Funkce slouží k pohodlnému výmazu spojového seznamu. Parametrem je počátek mazaného spojového seznamu `l`. Funkce zajistí uvolnění všech prostředků, které seznam zabíral. Implementace funkce je Vaším úkolem.

```
TITEM * sortListCmp ( TITEM * l, int ascending, int (*cmp)(const TITEM *, const TITEM *) )
```

Funkce slouží k seřazení prvků ve spojovém seznamu. Parametrem je počátek řazeného spojového seznamu `l`, požadované seřazení `ascending` a komparátor `cmp`. Funkce zajistí přeskupení prvků v zadaném seznamu tak, aby pořadí vyhovovalo požadovanému uspořádání. Funkce **nesmí** prvky původního seznamu uvolňovat (a např. vrátit jejich nově vytvořenou kopii). Naopak, **musí** přepojit odkazy existujících prvků a vrátit ukazatel na první prvek takto vzniklého seznamu.

Kritériem pro řazení je výsledek po zavolání komparátoru `cmp`. Vždy, když je potřeba určit vzájemné uspořádání dvou prvků, zavolá řadící algoritmus funkci předanou parametrem `cmp` Volání komparátoru pracuje stejně jako v knihovní funkci `qsort` - vrací zápornou / nulovou / kladnou hodnotu, pokud byl první parametr předaný komparátoru menší / stejný / větší než parametr druhý. Řazení dále ovlivní směr uspořádání - buď je vzestupné (parametr `ascending` není nula) nebo sestupné (parametr `ascending` je roven nule). Funkce musí zajistit, že řazení je **stabilní**.

Odevzdávejte zdrojový soubor, který obsahuje implementaci požadovaných funkcí. Do zdrojového souboru přidejte i další Vaše podpůrné funkce, které jsou z implementovaných funkcí volané. Funkce budou volané z testovacího prostředí, je proto důležité přesně dodržet zadané rozhraní funkce. Za základ pro implementaci použijte kód z přiloženého archivu. Ukázka obsahuje testovací funkci `main`, uvedené hodnoty jsou použité při základním testu. Všimněte si, že vkládání hlavičkových souborů a funkce `main` jsou zabalené v bloku podmíněného překladu (`#ifdef/#endif`). Prosím, ponechte bloky podmíněného překladu i v odevzdávaném zdrojovém souboru. Podmíněný překlad Vám zjednoduší práci. Při kompilaci na Vašem počítači můžete program normálně spouštět a testovat. Při kompilaci na Progtestu funkce `main` a vkládání hlavičkových souborů "zmizí", tedy nebude kolidovat s hlavičkovými soubory a funkcí `main` testovacího prostředí.

Poznámky:

- Zkopírujte si ukázku z přiloženého archivu a použijte ji jako základ Vašeho řešení.
- Do funkce `main` si můžete doplnit i další Vaše testy, případně ji můžete libovolně změnit. Důležité je zachovat podmíněný překlad.
- Testovací prostředí omezuje použití některých funkcí, konkrétně neprojde volání knihovních funkcí `qsort` a `qsort_r` (ty stejně nejsou určené pro spojové seznamy).
- V povinných testech jsou zadávané relativně krátké spojové seznamy, efektivita použitého řadicího algoritmu není kritická.
- První bonus kontroluje rychlost implementovaného řadicího algoritmu. Vstupem je dlouhý seznam, kvadratické algoritmy řazení nemají šanci seznam seřadit ve vymezeném časovém limitu.
- Druhý bonus kontroluje rychlost a paměťové nároky implementovaného řadicího algoritmu. Vstupem je dlouhý seznam, kvadratické algoritmy řazení nemají šanci seznam seřadit ve vymezeném časovém limitu. Paměťový limit navíc nedává šanci na větší alokace - k dispozici je navíc jen několik málo stovek KiB.
- V přiloženém ukázkovém zdrojovém souboru je vidět seznam dostupných hlavičkových souborů. Jiné hlavičkové soubory nejsou k dispozici a ani je nejde pomocí `#include` dodatečně vložit.

Vzorová data: Download

☐ **Referenční řešení**

1	**15.12.2022 17:05:05**	**Download**

Stav odevzdání:	Ohodnoceno
Hodnocení:	5.5000

- **Hodnotitel: automat**
  - Program zkompilován
  - Test 'Zakladni test podle ukazky': Úspěch
    - Dosaženo: 100.00 %, požadováno: 100.00 %
    - Celková doba běhu: 0.000 s (limit: 2.000 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test meznich hodnot': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %
    - Celková doba běhu: 0.012 s (limit: 2.000 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test nahodnymi daty': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %
    - Celková doba běhu: 0.016 s (limit: 1.988 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test nahodnymi daty + mem debugger': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %
    - Celková doba běhu: 0.022 s (limit: 2.000 s)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Bonus - rychlost': Program překročil přidělenou maximální dobu běhu
    - Vyčerpání limitu na celý test, program násilně ukončen po: 5.009 s (limit: 5.000 s)
    - Neúspěch v bonusovém testu, hodnocení: Bonus nebude udělen
  - Test 'Bonus - rychlost + pamet': Nebylo testováno
    - Neúspěch v bonusovém testu, hodnocení: Bonus nebude udělen
  - Celkové hodnocení: 100.00 % (= 1.00 * 1.00 * 1.00 * 1.00)
- Celkové procentní hodnocení: 100.00 %
- Bonus za včasné odevzdání: 0.50
- Celkem bodů: 1.00 * (5.00 + 0.50) = 5.50

SW metriky: