Projekt 1 - Ověřování síly hesel (práce s textem)

Motivace projektu

Snad všichni známe tu situaci, kdy si chceme vytvořit nové heslo do nějakého systému, ale ten nás pořád otravuje nesmyslnými požadavky na formát hesla. I když jednoduchá kombinatorika říka, že většina těchto pravidel je zbytečná a nejbezpečnější je prostě vytvořit dostatečně dlouhé heslo, pravidla pořád existují a uživatelé se s nimi musí naučit žít. V tomto projektu se na daný problém podíváme z opačné strany, konkrétně si vyzkoušíme, jak takový jednoduchý systém ověřující sílu hesel naprogramovat.

Popis projektu

Cílem projektu je vytvořit program, který na vstupu dostane sadu hesel a pro každé z nich ověří, jestli heslo splňuje všechna (pevně zadaná) požadovaná pravidla. Ta hesla, která projdou kontrolou, budou vypisována na výstup, ostatní budou zahozena.

Detailní specifikace

Program implementujte ve zdrojovém souboru "pwcheck.c". Vstupní data (seznam hesel) budou čtena ze standardního vstupu (stdin), výstup (filtrovaný seznam hesel) bude tisknut na standardní výstup (stdout).

Překlad a odevzdání zdrojového souboru

Odevzdání: Odevzdejte zdrojový soubor *pwcheck.c* prostřednictvím informačního systému.

Překlad: Program překládejte s následujícími argumenty:

```
$ gcc -std=c99 -Wall -Wextra -Werror pwcheck.c -o pwcheck
```

Syntax spuštění

Program se spouští v následující podobě: (./pwcheck značí umístění a název programu):

```
./pwcheck LEVEL PARAM [--stats]
```

Program je spouštěn se dvěma pevně zadanými argumenty `LEVEL` a `PARAM` a s jedním volitelným argumentem `--stats`, případně zadaným na třetí pozici:

`LEVEL`

celé číslo v intervalu [1, 4], které určuje požadovanou *úroveň bezpečnosti* (viz níže)

`PARAM`

kladné celé číslo, které určuje dodatečný *parametr pravidel* (viz níže)

`--stats`

pokud je zadané, určuje, zda se na konci programu mají vypsat souhrnné statistiky analyzovaných hesel

Úrovně bezpečnosti (kontrolovaná pravidla)

Jsou definovány celkem 4 úrovně bezpečnosti vyjádřeny pomocí 4 pravidel. Úroveň bezpečnosti určuje, že hesla musí splňovat všechna pravidla na dané a nižší úrovni. Tzn. např. úroveň bezpečnosti 3 specifikuje, že hesla musí splňovat pravidla 1, 2 a 3.

Některá pravidla jsou parametrizovatelná celým číslem zadaným pomocí argumentu programu `PARAM`. V následujícím seznamu je tento parametr označen jako X.

Seznam pravidel:

1. Heslo obsahuje alespoň 1 velké a 1 malé písmeno.
2. Heslo obsahuje znaky z alespoň X skupin (v případě, že je číslo X větší než 4, myslí se tím všechny skupiny). Uvažované skupiny jsou:
    - malá písmena (a-z)
    - velká písmena (A-Z)
    - čísla (0-9)
    - speciální znaky (podporované musí být alespoň nealfanumerické znaky z ASCII tabulky na pozicích ~~33-126~~ 32-126, tedy včetně mezery)
3. Heslo neobsahuje sekvenci stejných znaků délky alespoň X.
4. Heslo neobsahuje dva stejné podřetězce délky alespoň X.

Statistiky

Pokud je zadaný argument programu `--stats`, program musí na konec výstupu vypsat celkové statistiky ve formátu:

```
Statistika:
Ruznych znaku: NCHARS
Minimalni delka: MIN
Prumerna delka: AVG
```

kde `NCHARS` je *počet různých znaků* vyskytujících se napříč všemi hesly, `MIN` je *délka nejkratšího hesla* (resp. hesel) a `AVG` je *průměrná délka hesla* (aritmetický průměr) zaokrouhlená na 1 desetiné místo. Statistiky zahrňují i hesla, která byla zahozena.

Implementační detaily

Vstupní data (seznam hesel)

Seznam hesel je programu předán na standardním vstupu (stdin). Každé heslo je zadáno na samostatném řádku a obsahuje pouze ASCII textová data, kromě znaku nového řádku. Maximální délka hesla je 100 znaků, jinak se jedná o nevalidní data. Program musí podporovat neomezený počet hesel na vstupu.

Výstup programu

Program na standardní výstup (stdout) vypisuje hesla ze vstupního seznamu, každé na samostatný řádek, která splňují požadovanou úroveň bezpečnosti zadanou jako argument programu `LEVEL`. Hesla musí být vypsána beze změny a ve stejném pořadí, v jakém se objevila na vstupu.

Za výstupním seznamem hesel pak program volitelně vypisuje statistiku (viz. Statistiky).

Omezení v projektu

Je zakázané použít následující funkce:

- volání funkcí z knihoven `string.h` a `ctype.h` - cílem projektu je naučit se implementovat dané funkce ručně,
- volání z rodiny `malloc` a `free` - práce s dynamickou pamětí není v tomto projektu zapotřebí,
- volání z rodiny `fopen`, `fclose`, `fscanf`, ... - práce se soubory (dočasnými) není v tomto projektu žádoucí,
- volání funkce `exit` - cílem projektu je naučit se vytvořit programové konstrukce, které dokáží zpracovat neočekávaný stav programu a případně program řádně ukončit návratem z funkce main.

Program musí vyhovovat standardu ISO C99. Jsou povoleny pouze hlavičkové soubory: `stdio.h`, `stdlib.h`, `stdbool.h` a případně (není nutné pro funkcionalitu) `assert.h`, `errno.h` a `limits.h`.

Neočekávané chování

Na chyby za běhu programu reagujte obvyklým způsobem: na neočekávaná vstupní data, formát vstupních dat nebo chyby při volání funkcí reagujte přerušením programu se stručným a výstižným chybovým hlášením na příslušný výstup a odpovídajícím návratovým kódem. Hlášení budou v kódování ASCII česky nebo anglicky.

Příklady vstupů a výstupů

Pomocný soubor se seznamem hesel:

```
$ cat hesla.txt
1234567890
Password
Heslo123
Mojevelmidlouhehesloscislem0
IZP2021:fit@vut.cz
```

Příklady spuštění:

```
./pwcheck 1 1 <hesla.txt
Password
Heslo123
Mojevelmidlouhehesloscislem0
IZP2021:fit@vut.cz
./pwcheck 2 3 <hesla.txt
Heslo123
Mojevelmidlouhehesloscislem0
IZP2021:fit@vut.cz
./pwcheck 3 2 <hesla.txt
Heslo123
Mojevelmidlouhehesloscislem0
IZP2021:fit@vut.cz
./pwcheck 4 2 <hesla.txt
Heslo123
IZP2021:fit@vut.cz
./pwcheck 2 4 --stats <hesla.txt
IZP2021:fit@vut.cz
Statistika:
Ruznych znaku: 36
Minimalni delka: 8
Prumerna delka: 14.4
```

Hodnocení

Funkcionalita vašeho programu bude hodnocena na strojích s operačním systémem GNU Linux.

Na výsledné hodnocení mají hlavní vliv následující faktory:

- přeložitelnost zdrojového souboru,
- formát zdrojového souboru (členění, zarovnání, komentáře, vhodně zvolené identifikátory),
- dekompozice problému na podproblémy (vhodné funkce, vhodná délka funkcí a parametry funkcí),
- správná volba datových typů, případně tvorba nových typů,
- správná funkcionalita filtrování hesel,
- ošetření chybových stavů.

Priority funkcionality

1. Detekce jednotlivých pravidel: jelikož každá úroveň zabezpečení vyžaduje ověření všech pravidel nižší úrovně, dává smysl jednotlivá pravidla implementovat v pořadí, jak jsou zadána.
2. Vypisování statistik.
3. Další (prémiové) rozšíření funkcionality.

Prémiové řešení

Prémiové řešení je dobrovolné a lze za něj získat bonusové body. Podmínkou pro udělení prémiových bodů je výborné vypracování povinných částí projektu zadání. Výsledné hodnocení je plně v kompetenci vyučujího, který bude projekt hodnotit. Výše prémiových bodů závisí také na sofistikovanosti řešení.

Navrhované rozšíření:

- Pokročilé načítání argumentů příkazové řádky (až 1 bod): program nevyžaduje argumenty pouze na pevných pozicích, ale je možné zadat úroveň zabezpečení a parametr pravidel také pomocí volitelných přepínačů. Syntax spuštění:

```
./pwcheck [-l LEVEL] [-p PARAM] [--stats]
```

přičemž jednotlivé přepínače mohou být zadány v libovolném pořadí. Při nezadání přepínače je výchozí hodnota příslušného argumentu 1. Upozornění: program v rozšířené formě musí stále akceptovat argumenty příkazové řádky původní, nerozšířené varianty:

```
./pwcheck LEVEL PARAM [--stats]
```