Univerzita Karlova
Přírodovědecká fakulta

Doprovodný dokument

k programu

Nalezení nejkratšího a nejdelšího slova v textu

Úvod do programování
MZ370P19

Filip Zadražil

3. BGEKA

Chýnov, 5. 2. 2021

1) Zadání

Cílem tohoto programu je najít v textu z načteného souboru nejkratší a nejdelší slovo. Slova v textu musí být oddělena mezerami a mohou obsahovat běžná diakritická znaménka. Nalezené nejkratší a nejdelší slovo má být uloženo do textového souboru.

2) Použitý algoritmus

Proces hledání nejkratšího a nejdelšího slova v textu nemá nijak přesně pojmenované a popsané algoritmy. Pro sestavení tohoto programu byl v podstatě sestaven algoritmus vlastní, který je založen převážně na užití funkcí *try except*, *open*, *read*, *split*, *write* a *for* cyklu užitých ve dvou hlavních *funkcích def()*. První funkce s názvem *load_file* využívá hlavně *try except*, *open* a *read*. Algoritmus pokračuje druhou funkcí – *search_file*, která text z otevřeného souboru rozdělí na slova, která *for* cyklus projde a uloží výsledky do nového *txt* souboru.

3) Potenciální algoritmy

Pro hledání nejkratšího a nejdelšího slova v textu by šlo rovněž vzniklý algoritmus obohatit použitím funkcí *max()* a *min()*, které by však kvalitě kódu zřejmě nijak závratně nepomohly z hlediska kvality ani z hlediska přehlednosti. Nebylo jich však využito především z důvodu prostého opomenutí. Další, zcela zásadní úpravou algoritmu, by bylo využití knihovny *itertools*, jejíž funkce dokážou efektivně pracovat s textem a kód by se díky tomu jistě zkrátil. O její existenci jsem se však dozvěděl až po úplném dokončení programu.

4) Program

V první části programu dochází k otevření zdrojového souboru a jeho ošetření proti nevalidnímu vstupu. Jeho otevření funguje pomocí příkazu *open* ve funkci *try except*. Aby mohl sám uživatel zvolit, který soubor chce prozkoumat, je v blocích *try* a *except* užito proměnné *file_name*. Do té na konci programu, v řádku pro spuštění první funkce, vloží uživatel název požadovaného souboru. V bloku *try* zároveň dochází k dekódování textu pomocí utf-8. Lze v něm nalézt i první *if* podmínku, která zamezuje otevření prázdného souboru. Při ní došlo k využití knihovny *os* a funkci *path.getsize()*. V následných *except* blocích jsou vypsány podmínky zamezující vstupu např. nepřístupných či nenalezených souborů.
V hlavní funkci programu nejprve dojde k vytvoření proměnných, které zaznamenají znění nejkratšího/nejkratšího slova jejich délku. Poté je text rozdělen na slova pomocí funkce *split()*. Následuje samotný *for* cyklus, který prochází slovo po slově a zaznamenává vždy nejkratší a nejdelší slovo včetně jeho délky (funkce *len()*). Po konci cyklu dojde k uložení proměnných do nově vytvořeného textového souboru pomocí funkcí *open* a *write()*. Program končí voláním výše popsaných funkcí.

5) Reprezentace vstupních a výstupních dat

Vstupem do programu může být jakýkoli textový soubor o délce alespoň jednoho písmene. Nejvhodnějším vstupem je soubor, který je tvořen souvislým textem, není nijak složitě strukturován a neobsahuje specifická znaménka (např. matematické operátory). Při netextovém vstupním souboru se program ohlásí a skončí chybovou hláškou *"Soubor {file_name} je chybný."*. Do programu pochopitelně nelze vstoupit ani s neexistujícím,

případně nepřístupným souborem. V obou případech se program opět ohlásí příslušnou chybovou hláškou a skončí.
Výstupem programu je opět textový soubor. Tentokrát se však konkrétně jedná o soubor formátu *.txt*, jenž program sám vytvoří a požadovaný výstup do něj zapíše. V tomto případě se do výsledného souboru zapíší dvě proměnné – *longest_word* a *shortest_word*, do nichž je ukládán pomocí funkce write() výstup ve formátu *"Nejdelší slovo/slova v textu je/jsou: " + nejkratsi/nejdelsi + " (s délkou " + str(nejkratsi_delka/nejdelsi_delka) + ")."*.

6) Průběh práce

Práce na tomto kódu probíhala překvapivě hladce. Při psaní programu jsem bez přemýšlení začal využívat do té doby nabitých znalostí a nepřemýšlel o užití žádné knihovny pro samotné hledání slov. Důsledek toho je zřejmě o něco delší kód, jehož jednotlivé procesy jsou však pro uživatele daleko lépe představitelnější, než by tomu bylo za použití různých funkcí z knihoven. Zdrojový kód programu byl vytvářen postupně. Od samotného načtení souboru až po uložení výsledků do nového textového souboru.

7) Možná vylepšení programu

Teoretickým přínosem pro program by bylo například omezení na vstup pouze pro formát *.txt*. Nyní je program schopen prozkoumávat i formáty, které nejsou typicky textové (jako např. *.py*), obsahují často širokou škálu speciálních znaků a hledání nejdelšího a nejkratšího slova v nich reálně postrádá smysl.
Hlavní potenciální inovací programu by bylo rozsáhlejší ošetření právě proti výskytu různých specifických znaků. V současné podobě programu se tak může stát, že nejkratším nalezeným slovem bude znaménko + a naopak nejdelším nějaké slovo ohraničené v závorkách.

8) Závěr

Sestavování tohoto zdrojového kódu bylo pro měj jako pro autora velice přínosné a vyžadovalo bystrou mysl. O samotné praktické využitelnosti kódu by se dalo jistě polemizovat, ale dle mého názoru je tento program užitečný spíše jen tak „pro zajímavost“, neboť znát nejdelší a nejkratší slovo v textu mi nepřijde jako zvlášť důležitá informace. Kdybych býval tvořil program znovu, možná bych zauvažoval o užití některé z knihoven, specializovaných na práci s textem, ale i tak musím říct, že svého prvotního rozhodnutí nelituji.