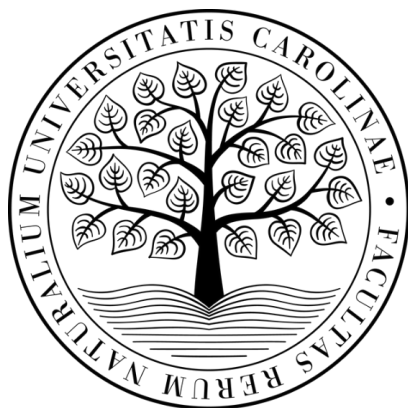


Univerzita Karlova
Přírodovědecká fakulta



ÚVOD DO PROGRAMOVÁNÍ

Odstranění duplicitních prvků z posloupnosti
a sdělení jejich počtu

Petra Krsková
3. ročník Geografie a kartografie
Solnice 2022

Zadání

Na vstupu je neseříděná posloupnost celých čísel. Nalezněte v posloupnosti duplicitní prvky, odstraňte je a sdělte jejich počet. Pokuste se tuto operaci učinit dostatečně rychle, aby byla metoda použitelná i pro dlouhé posloupnosti (v řádech stovek tisíc prvků). Vstupní data načtěte z textového souboru, výstup uložte také do textového souboru.

Součástí odevzdané úlohy bude zdrojový kód aplikace, vstupní/výstupní data a dokumentace se zadáním v rozsahu 4–5 stran ve formátu PDF obsahující následující:

- rozbor problému
- existující algoritmy
- popis zvoleného algoritmu
- struktura programu (datové struktury, metody,...)
- popis vstupních/výstupních dat
- problematická místa
- možná vylepšení

Aplikace bude považována za nefunkční, pokud:

- při zpracování dat dojde k pádu (runtime chyby, ...)
- vrací špatné výsledky
- neřeší možné singulární případy

Rozbor problému

Posloupnost v matematice představuje sadu objektů, v níž se mohou jednotlivé objekty opakovat a záleží na jejich pořadí. Počet objektů, který může být konečný i nekonečný, je vyjádřen délkou posloupnosti. Posloupnost lze definovat jako funkční vztah mezi množinou přirozených čísel (pozice člena posloupnosti) a jejich obrazem (hodnota členu posloupnosti na dané pozici). Posloupnost bývá označována písmeny ve formě a_n , kde dolní index n označuje n -tý člen posloupnosti (Wikipedia 2022).

Existující algoritmy

Pro odstranění duplicitních prvků z posloupnosti existuje několik řešení. První možností je použití FOR cyklu, ve kterém jsou členy původního seznamu představující danou posloupnost čísel porovnávány s novým seznamem. Pokud se v něm nenachází, jsou do něj přiřazeny a na konci cyklu tak uživatel získá nový seznam, ve kterém se prvky z původního seznamu vyskytují právě jednou. Tento způsob je možné zkrátit i do podoby jednoho řádku za použití generátorové notace seznamu (list comprehension). Nejrozšířenější metodou pro odstranění duplicitních prvků je použití funkce `set()`. Nejrychlejší variantou je pak využití funkce `collections.OrderedDict.fromkeys()`, která na rozdíl od předchozí zachovává pořadí členů posloupnosti (GeeksforGeeks 2020; JournalDev 2022).

Použitý algoritmus je však založen na prvotním setřídění posloupnosti, čehož lze dosáhnout několika způsoby. Algoritmus *Bubble Sort* je založen na porovnávání sousedních prvků, u kterých dochází k prohození, pokud nejsou ve správném pořadí. Metoda *Merge Sort* nejprve rozdělí posloupnost na dvě poloviny, které pak vzájemně porovnává a ve správném pořadí opět slučuje dohromady. *Insertion Sort* vyhledává správnou pozici prvku ve tříděné posloupnosti. Na začátku porovná první dva prvky a seřadí je. Následně vezme třetí prvek posloupnosti, porovná s dvěma předchozími a zařadí na správnou pozici. Toto je zopakováno pro všechny zbylé prvky posloupnosti. Jednou z možností je také *Shell Sort* založený na rozdělení posloupnosti na menší části a jejich setřídění. Další variantou je *Selection Sort* použitý v tomto případě, který je podrobněji popsán v následující kapitole (Tutorialspoint 2022).

Popis zvoleného algoritmu

Pro eliminaci použití vestavěných funkcí a možnost ukládání konkrétních vymazaných duplicitních hodnot je vstupní posloupnost nejdříve vzestupně setříděna a následně dochází k vypořádání duplicitních hodnot. Pro setřídění posloupnosti byl zvolen algoritmus *Selection*

Sort, který je založen na procházení seznamu a nalezení nejmenší hodnoty. Tato hodnota je zařazena na začátek setříděného seznamu a tento proces je opakován pro všechny zbývající prvky v dosud nesetříděném seznamu. Každý nově přiřazený prvek do již setříděného seznamu je nejprve porovnán s ostatními prvky a poté zařazen na odpovídající pozici.

V takto setříděné posloupnosti jsou pak postupně procházeny jednotlivé prvky a porovnávány se sousedními. Pokud se nerovnájí a daný prvek se již nevyskytuje v seznamu odpovídající výsledné posloupnosti, je do ní přiřazen. V opačném případě se jedná o duplicitní prvky a aktuální prvek je tak uložen do příslušného slovníku jako klíč, případně je zde jeho hodnota zvýšena o jedna. Tím je docíleno zaznamenání všech duplicitních prvků i toho, kolikrát navíc se v posloupnosti vyskytovaly.

Pseudokód zvoleného algoritmu

Struktura programu

Program sestává ze 88 řádek včetně komentářů a odsazení a obsahuje čtyři metody.

První metoda *SequenceFromFile* slouží pro otevření souboru a načtení vstupních dat. Pomocí *try* a *except* bloků je odchyťován špatně zadaný název souboru či cesta k jeho umístění, nedostatečné oprávnění pro přístup k souboru a další případné chyby spojené s otevíráním a čtením souboru. V případě chyby je do konzole vypsána chybová hláška definující problém a program skončí. Po otevření souboru je také zkontrolováno, zda obsahuje nějaká data a v případě prázdného souboru je opět vypsána chybová hláška a program skončí. Funkce vrací seznam hodnot typu *string* načtených ze souboru.

Druhá metoda *WriteToFile* slouží pro zápis výsledné posloupnosti čísel a duplicitních prvků do výstupního souboru. Podobně jako v předchozí metodě jsou zde pomocí *try* a *except* bloků ošetřeny možné chyby při otevírání souboru. Výstupní soubor je otevřen v módu *a*, který umožňuje připisování dat na aktuální konec souboru a nedochází tak k přepisu již existujících dat.

Třetí metoda *StrToInt* převádí seznam s hodnotami typu *string* na seznam s hodnotami typu *integer*. Toho je docíleno pomocí cyklu, který postupně prochází prvky seznamu a převádí je. Pokud narazí na prvek, který není číslem a nejde tak převést na datový typ *integer*, vypíše chybovou hlášku a daný prvek přeskočí.

Čtvrtá metoda *SortList* slouží k vzestupnému setřídění posloupnosti. Algoritmus této metody je podrobněji popsán v předchozí kapitole, ale je založen na *for* cyklu s vnořeným druhým cyklem, který prochází dosud neseříděné prvky a vybírá z nich minimum.

Hlavní část

V poslední části programu je pak do výstupního souboru zapsán celkový počet odstraněných duplicitních prvků posloupnosti. Toho je docíleno součtem všech hodnot ve slovníku *duplicates*. Následně jsou pomocí *for* cyklu procházeny klíče a hodnoty v tomto slovníku, které odpovídají vymazaným duplicitním prvkům a jejich počtu, a funkcí *WriteToFile* jsou zapsány do výstupního souboru.

Vstupní a výstupní data

Program načítá vstupní data z textového souboru s názvem *input.txt*, který obsahuje na první řádce všechny celočíselné prvky posloupnosti oddělené mezerou. Při jiném názvu souboru je potřeba změnit kód na řádce 62, v závorce změnit název souboru či upravit cestu k jeho umístění. Pokud by byly od sebe prvky odděleny jiným znakem, je potřeba upravit kód programu na řádce 10 a to tak, že se daný znak zapíše do uvozovek do závorky funkce *split()*.

Výslednou posloupnost a odstraněné duplicitní znaky včetně informace o tom, kolikrát byly vymazány, jsou zapisovány do textového souboru s názvem *output.txt*. Obdobně jako u vstupního souboru je v případě jiného názvu nutné upravit kód na řádkách 74, 83 a 88. Pro oddělení prvků výsledné posloupnosti jinak než mezerou je potřeba upravit kód na řádce 74 a to tak, že se v části *f"{element} "* mezera za složenou závorkou nahradí požadovaným znakem.

Problematická místa a možná vylepšení

Zdroje

GEEKSFORGEEKS (2020): Python – Ways to remove duplicates from list,
<https://www.geeksforgeeks.org/python-ways-to-remove-duplicates-from-list/> (5.2.2022).

JOURNALDEV (2022): Python Remove Duplicates from a List,
<https://www.journaldev.com/32742/python-remove-duplicates-from-list> (5.2.2022).

TUTORIALSPPOINT (2022): Python - Sorting Algorithms,
https://www.tutorialspoint.com/python_data_structure/python_sorting_algorithms
(7.2.2022).

WIKIPEDIA (2022): Sequence, <https://en.wikipedia.org/wiki/Sequence> (5.2.2022).