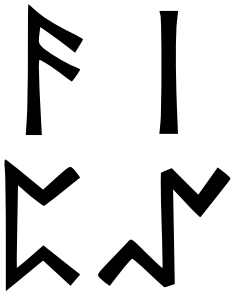
Úloha č. 2
Obsidiánové runy



Odpověz Sfinze! 10 b

Tato úloha je vyhodnocována automaticky. Je potřeba, aby výstup programu ***přesně*** *korespondoval se specifikací výstupu níže. Jak odevzdávat tento typ úloh se můžeš dočíst na webových stránkách FIKSu pod záložkou „Jak řešit FIKS“.*

Pro zadaný řetězec chceme najít jeho nejdelší podřetězec, ve kterém se žádný znak neopakuje.

Vstupní řetězec budeme číst znak po znaku. Budeme si pamatovat poslední nalezenou pozici (index) každého znaku ASCII (na začátku nastavíme např. na -1). Dále si budeme pamatovat délku zatím nejdelšího podřetězce a začátek a konec aktuálně zkoumaného podřetězce (vše na začátku nastavíme na 0).

Pro načtený znak budeme chtít zjistit, jestli je možné ho přidat na konec aktuálně zkoumaného podřetězce. Porovnáme tedy poslední pozici načteného znaku a pozici začátku podřetězce. Pokud se načtený znak už vyskytuje někde za začátkem podřetězce, nemůžeme tento podřetězec dále prodloužit. Zkusíme porovnat délku aktuálního podřetězce s nejdelším nalezeným a případně jej změníme. Dále nastavíme začátek aktuálního podřetězce těsně za poslední výskyt načteného znaku – získáme tak nejdelší možný podřetězec, za který můžeme načtený znak připojit. Nakonec vždy nastavíme poslední výskyt načteného znaku a konec aktuálního podřetězce zvýšíme o 1. Načteme další znak a celý postup opakujeme. Protože je řetězec „cyklický“, nemůžeme se zastavit po zpracování posledního znaku – nejdelší podřetězec by mohl dále pokračovat prvním znakem. Budeme tedy pokračovat znovu od začátku a zastavíme se až ve chvíli, kdy i začátek aktuálního podřetězce bude za posledním znakem – tedy vlastně podruhé někde na začátku, takový podřetězec jsme již určitě uvážili.

Složitost algoritmu

Zpracování jednoho znaku zabere konstantní čas a každý znak budeme zpracovávat nejvýše dvakrát, časová složitost je tedy $O(N)$, kde N je délka řetězce. Paměťová složitost je konstantní, resp. $O(P)$, kde P je počet různých znaků, které se mohou v řetězci vyskytnout, protože pro každý z nich si musíme pamatovat poslední pozici.

Přiložený kód řešení je v jazyce Go.