**Testování slov**

**Úkol 1-1:** Na vstupu máme zadaných $n$ slov délky $k$ a jedno vzorové slovo délky $k$. Rozhodněte, která ze zadaných slov jsou rotací vzorového slova, **v lineárním čase. Tzn. se složitostí úměrnou celkové velikosti vstupu, bez -dalších- multiplikativních konstant závislých na $n$ a $k$**
Slovo je rotací jiného, pokud vznikne cyklickým posunem znaků. Např. $abcdef$ je rotací $efabcd$.

**Řešení:**

Myšlenkou řešení je, že pokud slovo je rotací vzoru a zřetězíme ho dvakrát za sebou, někde uprostřed se bude nacházet původní vzor.
Postup řešení je takový, že nejprve zkonstruujeme vyhledávací automat (dle KMP) pro hledání vzoru. Následně pro každé slovo budeme vyhledávat projitím slova dvakrát dokola. Pokud vzor najdeme, slovo prohlásíme za rotaci vzoru.
Časová složitost je $\mathcal{O}(nk)$, tolik času strávíme na čtení zdvojených vstupních slov automatem. Pro konstrukci automatu potřebujeme pouze $\mathcal{O}(k)$.
Prostorová složitost bude pouze $\mathcal{O}(k)$ pomocného prostoru ($\mathcal{O}(nk)$ celkem). Pomocný prostor potřebujeme $\mathcal{O}(k)$ pouze pro automat (a nějakých $\mathcal{O}(1)$ ukazatelů). Zdvojování slov můžeme dělat několika způsoby, buď všechna zdvojit na začátku (to by vyžadovalo dokonce $\mathcal{O}(nk)$ pomocné paměti), nebo vždy máme pouze aktuální slovo zdvojené v paměti ($\mathcal{O}(k)$), nebo automat krmíme znak po znaku pouze přečtením každého slova dvakrát za sebou ze vstupu ($\mathcal{O}(1)$, stačí nám pouze nějaké ukazatele a počítadla).
Korektnost částečně plyne z korektnosti KMP, tedy pokud zdvojené slovo obsahuje vzor (a jen tehdy) ho KMP nalezne. Mějme tedy slovo $S$, které je rotazí vzoru $V$, tedy $V = AB$ a $S = BA$ pro nějaké podřetězce $A$ a $B$. Potom $SS = BABA = BVA$ obsahuje vzor. Naopak, pokud $SS = CVD$, všimneme si, že libovolná rotace $SS$ je také zdvojený řetězec. Zvolme libovoně dlouhé $Y$ tak aby $SS = XYXY$, rotací o $|Y|$ symbolů doprava dostaneme $YXYX$. Rotací $CVD$ o $|D|$ symbolů doprava tedy získáme $DCV = TT$ pro $T$ rotaci $S$, z čehož plyně, že $T = V$. Tím je dokázáno, že $S$ je rotace $V$.

**Řešení hešováním:**

Můžeme použít i myšlenku Rabin-Karpova algoritmu. Buďto úplně stejně jako v předchozím řešení (tzn. spočítáme heš vzoru, každé slovo zdvojíme a hledáme v něm vzor), nebo můžeme v čase $\mathcal{O}(k)$ sestrojit heše všech rotací vzoru úplně stejnou taktikou, zdvojením. Každé slovo pak stačí rovnouzahešovat a porovnat s možnými hodnotami heše (přímo nebo pomocí nějakého BVS, hešování slova stojí $\mathcal{O}(k)$, takže na tom příliš nesejde). Stačí nahlédnout, že když pravděpodobnost kolizí heše bude nejvýše $\mathcal{O}(1/k)$ (snadno zvládneme i lepší), dostaneme $\mathcal{O}_E(nk/k)$ očekávaných kolizí s celkovým časem ošetření $\mathcal{O}_E(k * nk/k) = \mathcal{O}(nk)$. Nezapomeňme také na kontrolu skutečných shod heše, ty zaberou také $\mathcal{O}(nk)$, ale musíme každé slovo opustit při první skutečné shodě.

**Úkol 1-2:** Pro text na vstupu délky $n$ najděte v linárním čase nejdelší prefix, který je palindrom.

**Řešení:**

Mějme vstup $S$, a označme si hledaný palindrom jako $P$.
Nejprve provedeme návodné pozorování. Pokud sestrojíme automat vyhledávající $S$, a spustíme ho nad senem (libovolným), které obsahuje $P$, potom v okamžiku po přečtení části textu s $P$ bude automat ve stavu odpovídajícímu délce prefixu $S$ minimálně tak dlouhého jako je $|P|$. To plyne z vlastností KMP, jeho stav musí vždy odpovídat nejdelšímu prefixu jehly, který je sufixem doposud přečteného textu. Pokud tedy $P$ je suffix doposud přečteného textu, stav KMP musí odpovídat minimálně stějně dlouhému prefixu $S$ (pozor, obecně může odpovídat i delšímu). Díky vlastnosti palindromu totéž platí i pro stejné seno pokud ho přečteme pozpátku.
Postupů jak tohoto principu využít je několik.
Nejčistší řešení je sestrojit automat pro hledání $S$ a spustit ho nad obráceným $S^{-1}$, délka $P$ pak odpovídá délce aktivního prefixu jehly automatu po přečtení $S^{-1}$. Korektnost tohoto postupu plyne z pozorování výše, po přečtení $S^{-1}$ jsme určitě právě dokončili čtení $P$ a tedy právě aktivní prefix jehly obsahuje minimálně celý $P$. Stačí nahlédnout, že aktuální stav neodpovídá nějakému delšímu prefixu $S$. To plyne z toho, že aktivní prefix jehly je roven suffixu sena, což je ale převrácený prefix $S$. Prefix jehly aktivní na konci čtení tedy musí být sám také palindromický prefix na začátku $S$ (pokud by byl delší než $P$, máme spor s volbou $P$).
Oblíbený přístup byl zřetězit $S$ s $S^{-1}$ a oddělujícím speciálním symbolem $x$ uprostřed. Pro takto získaný řetězec $T = SxS^{-1}$ se následující postup dost lišil. Některá řešení se snažila zkonstruovat pouze polovinu automatu na vyhledávání $T$, některá postavila celý a načítala pouze polovinu $T$, obě tyto konstrukce obykle selhaly na zbytečných komplikacích. Dá se nahlédnout, že pokud postavíme automat pro vyhledávání $T$ a spustíme ho nad $T$, dosáhneme stejného efektu jako v předchozím řešení, pouze oklikou.
Budováním automatu nad $T$ ale můžeme "rovnou" nahlédnout, že nám stačí se podívat na zpětnou hranu z posledního stavu. Ta totiž vede do stavu představující nejdelší prefix $S$, který je suffixem $SxS^{-1}$ (vložením symbolu $x$ dokonce pouze suffixem $S^{-1}$). Všimněme si, že toto je vlastně ekvivalentí prvnímu postupu.
Budováním automatu pro $T$ nejprve vybudujeme podautomat pro $S$, a konstrukce zpětných hran pro stavy 'za symbolem $x$' vlastně odpovídá načítání $S^{-1}$ tímto podautomatem. Podotkněme, že vložení $x$ je pro takový přístup nutné, např pro $S = aabba$ by $T = aabbaabbaa$ a poslední zpětná hrana by vedla na stav $aabbaa$.
Časová i prostorová složitost ve všech případech je $\mathcal{O}(n)$. Zkonstruování automatu, použití automatu, prostor na případné předzpracování slov, i samotný automat jsou vždy lineární v $n$.