Pracovný list – Úlohy na reťazcoch

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Úloha 1** | Úloha 1. Katka sa chystá na študijný pobyt v USA. Naštudovala si, že okrem inej meny a dĺžkových jednotiek používajú obyvatelia USA aj inú jednotku teploty – stupeň Fahrenheita (F). Zaskočilo ju to, lebo zistila, že prepočet medzi stupňami Fahrenheita a Celzia je dosť náročný.  Vytvorte program **prevod\_teploty.py**, ktorý na vstupe dostane teplotný údaj (celé číslo v stupňoch Fahrenheita alebo v stupňoch Celzia) – predpokladajme, že vstup je vždy správne zadaný. Výstupom programu bude teplotný údaj v druhej jednotke teploty, ktorý vypočíta vami definovaná funkcia **prevod()**. Príklad vstupu a výstupu:   |  |  | | --- | --- | | Vstup | Výstup | | 45 F | 45 F = 7. 222222222222222 C | | 37 C | 37 C = 98.60000000000001 F | | 2 C | 2 C = 35.6 F | | 157 F | 157 F = 69. 44444444444444 C |   Pomôcka   * vzorec pre prevod teploty zo stupňov Celzia na stupne Fahrenheita: F = C \* 1.8 + 32, z toho vzorca vieme určiť výpočet °C ak poznáme °F: C=...... * predpokladajme korektný vstup v tvare celé číslo – medzera – jednotka tepla (F alebo C), * vieme sa dostať k jednotlivým dôležitým údajom pomocou výrezov - jednotka teploty sa nachádza na poslednej pozícii reťazca (s indexom 0), číselný údaj od začiatku po tretiu pozíciu od konca (tá má index –3).   úloha na zopakovanie si   * indexovanie reťazca * výrez * vnorené podmienky * pretypovanie reťazca |

|  |  |
| --- | --- |
| **Úloha 2** | Peter poprosil svoju sestru Betku, aby prepísala jeho rukou písaný referát na počítači. Keď mu sestra odovzdala prepísaný a vytlačený referát, zistil, že pri písaní mala zamenené klávesy y a z.  Vytvorte program **vymen\_y\_z.py**, ktorý vstupný reťazec upraví tak, že opraví chyby spôsobené zamenenými klávesmi y a z.  úloha na zopakovanie si   * reťazcová metóda replace() |

|  |  |
| --- | --- |
| **Úloha 3** | Pani učiteľka slovenského jazyka posudzuje náročnosť diktátu podľa počtu písmen i, í, y, ý v diktovanom texte. Definujte funkciu pocet\_iy(), ktorá pre zadaný reťazec vypíše, koľko písmen i, í, y, ý reťazec obsahuje.  úloha na zopakovanie si   * cyklus na reťazci * operátor príslušnosti * logické operátory a zložené podmienky (ako príklad neefektívneho riešenia) * reťazcová metóda count() |

|  |  |
| --- | --- |
| **Úloha 4** | Vo vstupnom formulári pre evidenciu nových zamestnancov programátori vytvorili textové pole pre zadanie mena a priezviska zamestnanca. Neskoro si uvedomili, že pre potreby databázy zamestnancov potrebujú samostatne meno a samostatne priezvisko.  Vytvorte pre potreby programátorov program **zamestnanec.py**. Vstupom programu je reťazec v tvare 'menomedzerapriezvisko'. Definujte v ňom funkcie **meno()** a **priezvisko()**, ktorých vstupným parametrom je spomínaný reťazec načítaný v hlavnom programe. Návratovou hodnotou funkcie meno() je meno zamestnanca (prvé písmeno veľké, ostatné malé), návratovou hodnotou funkcie priezvisko() je priezvisko zamestnanca (všetky písmená veľké).  úloha na zopakovanie si   * reťazcová metóda find(),upper() , lower() |

Riešenia:

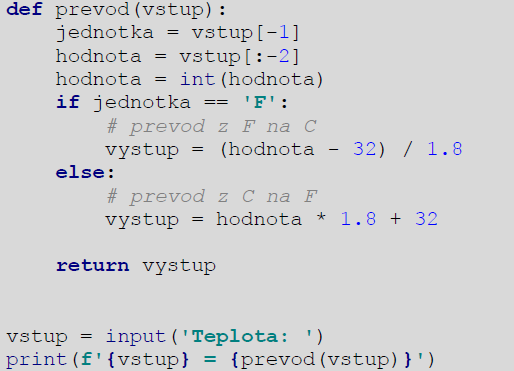
***Úloha 1:***

Riešenie úlohy je založené na analýze vstupného reťazca. Keďže môžeme predpokladať korektný vstup v tvare celé číslo – medzera – jednotka tepla (F alebo C), vieme sa dostať k jednotlivým dôležitým údajom pomocou výrezov. Jednotka teploty sa nachádza na poslednej pozícii reťazca (s indexom 0), číselný údaj od začiatku po tretiu pozíciu od konca (tá má index –3).

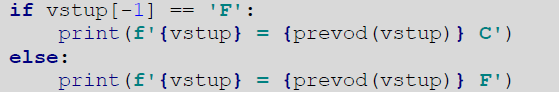
jednotka = vstup[-1]

hodnota = vstup[:-2]

Reťazec reprezentujúci číselný údaj pretypujeme na celé číslo. Pri výpočte zistíme, z akej jednotky teploty do ktorej prevádzame a vykonáme prevod:



Má to však ešte jeden malý nedostatok – vo výstupe nie je zobrazená výsledná jednotka teploty. Preto upravíme posledný riadok hlavného programu tak, aby sa nám podarilo vypísať túto jednotku:



***Úloha 2:***

Riešenie: Prvým nápadom môže byť riešenie založené na postupnom prechode reťazca. Pre každý jeho

znak by sme zisťovali či je alebo nie je jedným zo znakov y, Y, z, Z a ak:

* nie je, pripojili by sme ho do novovytváraného výstupného reťazca,
* ak je to y, no novovytváraného výstupného reťazca pripojíme znak z,
* ak je to Y, no novovytváraného výstupného reťazca pripojíme znak Z,
* ak je to z, no novovytváraného výstupného reťazca pripojíme znak y,
* ak je to Z, no novovytváraného výstupného reťazca pripojíme znak Y,

My však máme k dispozícii nástroje jazyka Python a využijeme ich. Ak sa pozrieme do pomocného materiálu k reťazcovým metódam, nájdeme tam metódu replace(čo, čím).

Teraz to už vyzerá jednoducho, kým si neuvedomíme, že ak nahradíme všetky výskyty znaku y znakom z, nebudeme vedieť nahradiť pôvodné znaky z znakom y.

Pomôžeme si zástupnými znakmi, napr. môžeme predpokladať, že v texte sa nenachádzajú podreťazce ‘qwS’ a ‘!\*\_’. Prvým nahradíme znak y, druhým znak Y. Potom už môžeme nahradiť znaky z a Z za y a Y. Následne už stačí nahradiť zástupné znaky znakmi z a Z.



***Úloha 3:*** Riešenie:

Ukážeme si dva možné prístupy k riešeniu tejto úlohy. To, ktorým začneme, vyplynie z prvých reakcií a návrhov žiakov. Po realizácii prvého riešenia odporúčame predstaviť žiakom aj druhé riešenie – môžeme ho premietnuť na tabuľu a diskutovať o ňom (porovnať náročnosť, funkčnosť).

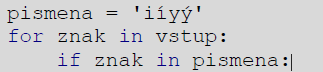
**I. spôsob riešenia**

Pri riešení úlohy budeme prechádzať jednotlivými znakmi skúmaného reťazca (premenná vstup). Na to použijeme cyklus for (v tvare for znak in retazec).

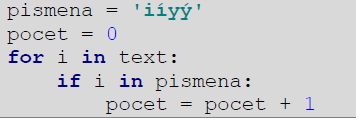
Každý znak reťazca otestujeme – či to je jedno zo skúmaných písmen. Žiaci možno navrhnú „drevorubačský“ postup – pomocou zložených podmienok a logického operátora or otestovať zhodu:

**if** znak == **'i' or** znak = **'í' or** znak = **'y' or** znak = **'ý'**:

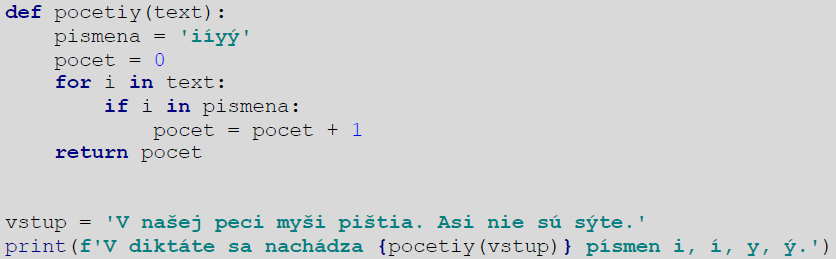
Namiesto zdĺhavého porovnávania a použitia zložených podmienok predsa vieme použiť operátor príslušnosti in – teda či sa znak nachádza v reťazci vytvoreného zo sledovaných písmen:



Čo chce urobiť, ak zistíme, že znak je jedným z hľadaných písmen? Potrebujeme si zaevidovať, že sme našli ďalší takýto znak – použijeme na to premennú pocet, ktorej počiatočná hodnota je 0 a pri každom nájdení sledovaného písmena sa jej hodnota zvýši o 1:



V tejto fáze môžeme prejsť k vytvoreniu funkcie a jej prepojeniu s hlavným programom:

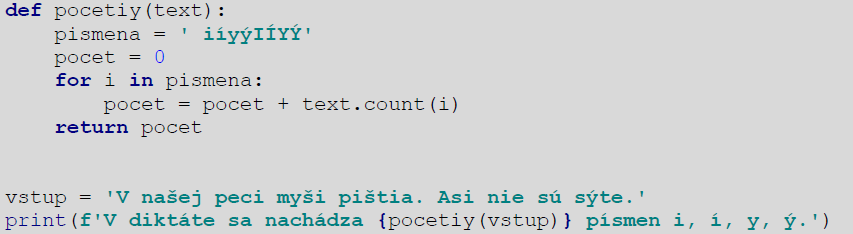


**II. spôsob riešenia**

Pri riešení úlohy použijeme reťazcovú metódu retazec.count(podretazec), ktorá vracia počet výskytov podreťazca v reťazci (podreťazcom môže byť aj jediný znak). Z tohto pohľadu by sa nám núkalo riešenie:

pocet = vstup.count('i') + vstup.count('í') + vstup.count('y') + vstup.count('ý')

My sa však na úlohu musíme dívať komplexne – čo ak by tých „sledovaných“ písmen bolo viac (napr. aj veľké písmená)? Definujeme si preto reťazec, ktorý bude obsahovať tieto písmená. Postupne budeme týmto reťazcom prechádzať (pomocou cyklu for znak in retazec) a budeme zisťovať, koľko krát sa jednotlivé písmená nachádzajú v skúmanom reťazci:



***Úloha 4:*** Riešenie:

Je zrejmé, že dôležitým momentom riešenia je zistenie pozície medzery a „vytiahnutie“ mena a priezviska do samostatných premenných. Potom ich môžeme upraviť podľa stanovených pravidiel.

Na zistenie pozície medzery použijeme metódu find() (predpokladáme, že reťazec je zadaný korektne a medzera sa v ňom nachádza). Pred touto medzerou končí podreťazec reprezentujúci meno (začína na začiatku reťazca), a za ňou začína podreťazec reprezentujúci priezvisko (až do konca reťazca).

Získané meno upravíme pomocou metódy upper(), ktorú aplikujeme na prvý znak (s indexom 0), a metódy lower(), ktorú aplikujeme na zvyšok mena. Výsledky oboch metód zreťazíme (+) a priradíme do novej premennej vystup.

**Vyrieš samostatne za domácu úlohu ☺**