Pracovný list – reťazec

Skúmanie

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Úloha 1** | Otvorte program **11\_01\_retazec.py**.   |  |  |  | | --- | --- | --- | |  | meno = 'Juraj' |  | |  | priezvisko = 'Bok' |  | |  | spolu = meno+' '+priezvisko |  | |  | print(spolu) |  | |  | print(len(spolu)) |  | |  | print(spolu[4]) |  | |  | print(spolu[2:5]) |  | |  | print(spolu[6:]) |  | |  | if 'raj' in spolu: |  | |  | print('ano') | |  | else: | |  | print('nie') | |  | for znak in spolu: |  | |  | print(znak) |   Program spusťte viackrát. Skúšajte meniť reťazce/znaky, čísla, operácie s reťazcami a sledujte výpisy programu do konzoly. Na základe svojich pokusov vyplňte prázdny stĺpec tabuľky, v ktorom vysvetlíte, čo je výsledkom jednotlivých príkazov. |

Vysvetlenie

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Úloha 2** | Zuzka sa odsťahovala s celou rodinou do Kanady. So svojou najlepšou kamarátkou Katkou komunikujú písomne a na utajenie svojich správ si dohodli šifru: do textu správy vložia za každý znak ľubovoľný znak. Takto upravená správa vyzerá ako motanica nezmyselných slov, napr. text Ahoj Zuzka! po zašifrovaní vyzerá takto: A\*huoXjj QZ8uyzKk+a,!(“.  Aby sa im správy ľahšie dešifrovali, obidve vytvorili vlastnú funkciu desifruj(), ktorej návratovou hodnotou je dešifrovaná správa. Každá z funkcií však vyzerá odlišne, dievčence sa nevedia dohodnúť, ktorá je správna. Pomôžte im pri rozhodovaní – určte, ktorá z funkcií plní danú úlohu.   |  |  | | --- | --- | | Zuzkina funkcia: | Katkina funkcia: | | **def** desifruj(s):  vysledok = ''  **for** i **in** range(0, len(s), 2):  vysledok = vysledok + s[i]   **return** vysledok | **def** desifruj(s):  vysledok = s[::2]   **return** vysledok | |
| **Úloha 3** | Vytvorte program **palindrom.py**, ktorý zistí, či je slovo zadané na vstupe palindróm. Slovo zadávame malými písmenami, nepoužívame medzery a diakritiku.  Poznámka: Palindróm je slovo, veta, číslo (všeobecne akákoľvek postupnosť symbolov), ktorá má tú vlastnosť, že ju možno čítať v ľubovoľnom smere (sprava doľava alebo zľava doprava) a má vždy rovnaký význam. |

Rozpracovanie

|  |  |
| --- | --- |
| **Úloha 4** | Tajomstvo komunikácie Zuzky a Katky odhalil Katkin brat Miško. Preto sa dievčatá rozhodli, že budú komunikovať po anglicky a zároveň budú používať Pig Latin – jazykovú hru, ktorá slúži na pobavenie, aj na utajenie komunikácie pred nepovolanými osobami. Princíp hry spočíva v úprave slov podľa týchto pravidiel   * Ak slovo začína spoluhláskou, táto sa presunie na koniec slova a za ňu sa pridá prípona –ay, napr. door => oorday, pen => enpay. * Ak slovo začína samohláskou, pridá sa len prípona –way, napr. apple =>appleway, old => oldway.   Vytvorte program **pig\_latin.py**, ktorý na vstupe dostane slovo (zapísané malými písmenami anglickej abecedy) a do konzoly vypíše toto slovo upravené podľa pravidiel jazykovej hry Pig Latin. |

Sebahodnotiaci test

|  |  |
| --- | --- |
| 1. | Nasledujúci program dešifruje vstupnú správu, ktorá vznikla podľa tohto pravidla šifrovania – pred a za každý znak správy bol vložený jeden náhodný znak (napr. správa „Pošli správu.“ je zašifrovaná v tvare „3Pxaob4šSbl4sil4s45pM0r7GáAAvmKuL8.4“). Doplňte chýbajúcu časť kódu, aby bol program funkčný pre ľubovoľnú správne zašifrovanú správu.  zasifrovana\_sprava = input(’Zašifrovaná správa: ’)  odsifrovana\_sprava = zasifrovana\_sprava[ : : ]  print(f’{zasifrovana\_sprava} => {odsifrovana\_sprava}’) |
| 2. | Aký výstup bude mať nasledujúci program, ak mu na vstupe zadáme slovo ‘Zima’?  retazec = input('Vstupný reťazec: ') podretazec = retazec[-1] + retazec[::-1] + retazec[0] print(podretazec)  Výstup programu pre vstupné slovo ‘Zima’: |

|  |  |
| --- | --- |
| Vedomosti v kocke |  |
| Pri práci s reťazcami vieme používať:   * Operáciu +, ktorá umožňuje spojiť viaceré reťazce do nového v poradí, v akom ich v súčte uvedieme (tento súčet nie je komutatívny). * Relačné operátory <, >, >=, <=, ==, !=, ktoré umožňujú porovnávať reťazce; napr. ‘auto’ > ‘astma’, heslo == ‘informatika’ apod. * Operátor príslušnosti in, ktorý umožňuje pohybovať sa po znakoch reťazca; napr. nasledujúci program vypíše z reťazca priezvisko len písmená a, e, i, o, u, y (v programe sa in nachádza dva krát – len tučným písmom zvýraznené **in** je operátorom príslušnosti):   for znak in priezvisko:  if znak **in** 'aeiouy':  print(znak)   * Funkciu len(), ktorej návratovou hodnotou je dĺžka reťazca. * Indexovanie ­– k jednotlivým znakom daného reťazca môžeme pristupovať pomocou indexov (nekladných alebo záporných), pričom prvý znak má index 0.   Ak zvolíme napr. retazec = 'Informatika', potom platí:   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **I** | **n** | **f** | **o** | **r** | **m** | **A** | **t** | **i** | **k** | **a** | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | | -11 | -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |  * retazec[4] => r * retazec[-1] => a * retazec[20] => chybové hlásenie * Výrezy – k podreťazcom daného reťazca môžeme pristupovať pomocou výrezov, resp. výrezov s krokom, pričom formálny zápis tejto operácie má tvar retazec[zaciatok:koniec], kde hodnota zaciatok určuje prvý znak výrezu, hodnota koniec určuje posledný znak výrezu – znak s indexom koniec – 1.   Ak neuvedieme prvý index (zaciatok), výrez začne od začiatku reťazca. Ak neuvedieme druhý index (koniec), výrez skončí posledným znakom reťazca. Ak neuvedieme ani jeden index, budeme pracovať s celým reťazcom; napr. použijeme reťazec z predchádzajúceho bodu:   |  |  | | --- | --- | | * *retazec[:] => Informatika* * *retazec[5:] => matika* * *retazec[:4] => Info* * *retazec[5:10] => matik* | * *retazec[2::3] => fmi* * *retazec[-4:] => tika* * *slovo[::2] => Ifraia* * *slovo[::-1] => akitamrofnI* | | |