Pracovný list č. 12 – Reťazce

|  |  |
| --- | --- |
| **Úloha 1** | Otvorte program **retazec\_01.py**.  slovo1 = 'zu' slovo2 = 'lienka' spolu = 2 \* slovo1 + slovo2  print(spolu) print(len(spolu))  if 'ie' in spolu:  print('ano')  else:  print('nie')  print(spolu[4])  Program spusťte viackrát. Skúšajte meniť reťazce/znaky, čísla, operácie s reťazcami a sledujte výpisy programu do konzoly. Na základe svojich pokusov odpovedzte na nasledujúce otázky:   1. Doposiaľ sme pri výpočtoch s číslami používali aritmetické operácie +, –, \* a /. Ktoré z týchto operácií môžeme používať aj pri práci s reťazcami/znakmi? Čo je výsledkom použitia týchto operácií?   + zreťazenie alebo spojenie reťazcov  \* zopakovanie reťazca  program vypíše zuzulienka   1. Čo je výsledkom operácie retazec1 in retazec2? nájdenie reťazca v reťazci(hľadá reťazec ‚ie‘ v reťazci ‚zuzulienka‘) 2. Čo je návratovou hodnotou funkcie len(retazec)? počet znakov v reťazci (10) 3. Čo vypíše program do konzoly príkazom v poslednom riadku print(spolu[4])znak s poradovým číslom 5, ale indexom 4. (vypíš písmeno l) |

|  |  |
| --- | --- |
| **Úloha 2** | Porovnajte výstupy nasledujúcich cyklov for – najprv odhadnite ich výstupy, svoje predpoklady overte spustením programu **retaze\_02c.py**:  retazec = 'Informatika'  for i in range(len(retazec)):  print(i)  for i in retazec:  print(i)  Prvý cyklus for vypíše:   1. môj tip 2. skutočnosť indexy jednotlivých znakov v reťazci na samostatný riadok   Druhý cyklus for vypíše:   1. môj tip 2. skutočnosť jednotlivé znaky reťazca na samostatný riadok |

**Poznámky**

* textové reťazce spájame operáciou **+**
* opakovanie textového reťazca operáciou **\***
* náhodný výber reťazca zo zoznamu **random.choice(('text1‘,‘text2‘,‘text3‘))**
* načítať zo vstupu pomocou **input()**
* vypísať pomocou **print()**
* zmeniť textový reťazec na číslo **int(), float()**
* zmeniť číslo na textový reťazec **str()**
* dĺžka reťazca **len()**
* výskyt konkrétneho znaku v reťazci **in**
* prístup k jednotlivým znakom reťazca **indexovanie []**
* porovnávanie jednoznakových reťazcov pomocou relačných operátorov **==, !=, <, <=, >, >=**
* kódovanie znakov do a z **Unicode** - znak na číslo **ord(),** číslo na znak **chr()**
* prevod celého čísla do reťazca v dvojkovej sústave **bin()**
* prevod celého čísla do reťazca v šestnástkovej sústave **hex()**
* prevod celého čísla do reťazca v osmičkovej sústave **oct()**

**Podreťazce**

Príklad: reťazec = ‘Informatika‘

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |

znaky číslujeme od nuly

výrez reťazca – podreťazec (časť reťazca)

retazec[*zaciatok:koniec:krok*]

kde retazec – meno premennej

zaciatok – prvý znak výrezu reťazca

koniec – posledný znak výrezu reťazca

krok – hodnota, o ktorú sa budeme pri výbere znakov posúvať v reťazci

|  |  |
| --- | --- |
| **Úloha 3** | V okne Shell do premennej slovo vložte ‘Python‘. Zobrazte obsah premennej slovo. Potom zistite, čo sa vypíše do okna Shell, ak budete postupne zadávať:  slovo[2]  slovo[1:4]  slovo[:5]  slovo[3:]  slovo[::2]  slovo[::-1]  slovo[-5]  slovo[-6:]  slovo[1:7:3]  slovo[-6:-1:2]  slovo[1:7:-2]  for i in slovo:  print(i)  Odfoťte obrazovku a obrázok vložte do pracovného listu |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Úloha 4** | 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.  Správne pracujú obidve funkcie. Líšia sa len postupom. Zuzkina funkcia prechádza reťazcom a hodnotu, na ktorú odkazuje premenná **vysledok**, postupne mení tak, že k nej pripája nultý, druhý, štvrtý, ... znak pôvodného. Katkina funkcia využíva operáciu indexovania, presnejšie prácu s výrezmi vstupného reťazca. Do premennej **vysledok** sa priradí každý druhý znak počnúc nultým až po koniec reťazca.   |  |  | | --- | --- | | 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 5** | Pani učiteľka v rámci záverečného opakovania pripravuje na každú hodinu slovenského jazyka krátky diktát. Jeho náročnosť posudzuje najmä podľa počtu písmen i, í, I, Í, y, ý, Y, Ý v diktáte.  Vytvorte pre pani učiteľku program, ktorému zadá text diktátu a program spočíta a vypíše počet sledovaných samohlások i, í, I, Í, y, ý, Y, Ý v zadanom texte.  Napr. pri vstupnom texte „V našej peci myši pištia. Asi nie sú sýte.“ bude odpoveďou číslo 8.“  Riešenie:  vstup = input('Zadaj vetu')  pismena = 'iíyýIÍÝY'  pocet = 0  for i in vstup:  if i in pismena:  pocet += 1  print('V diktáte sa na chádza', pocet,'iíyýIÍÝY') |

|  |  |
| --- | --- |
| **Úloha 6** | 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.  Riešenie:  vstup = input('Zadaj slovo')  if vstup[0] in 'aeiouy':  vystup = vstup[0:]+'way'  else:  vystup = vstup[1:]+vstup[0]+'ay'  print('Zašifrované slovo je', vystup) |

|  |  |
| --- | --- |
| **Úloha 7** | Napíšte program **vety.py**, ktorý bude tvoriť náhodné vety pozostávajúce zo slov kto, corobil, ake, co. Pre každé slovo vytvorte zoznam slov. Pomocou cyklu for vypíšte 15 náhodných viet.  Riešenie:  import random  for i in range(1,21):  kto = random.choice(('Kamarát','Spolužiak','Andrej','Roman'))  corobil = random.choice(('videl','prezradil','povedal','napísal','zistil','nakreslil'))  ake = random.choice(('veľké','malé','obrovské','drobné','smutné','veselé'))  co = random.choice(('tajomstvo','prekvapenie','predsavzatie'))  spojene = kto +' '+corobil+' '+ake+' '+co+'.'  print(spojene) |

|  |  |
| --- | --- |
| **Úloha 8** | Napíšte program **dlzka.py**, ktorý zistí dĺžku reťazca zadaného z klávesnice.  Riešenie:  text = input('Zadaj text ')  d = len(text)  print(d) |

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 „3Pxaob4šSbl4sils 4Is45pM0r7GáAAvmKuL8.4“).  Doplňte chýbajúcu časť kódu, aby bol program funkčný.  zasifrovana\_sprava = input(’Zašifrovaná správa: ’)  odsifrovana\_sprava = zasifrovana\_sprava[1: :3]  print(f’{zasifrovana\_sprava} => {odsifrovana\_sprava}’) |
| 2. | Vyberte z ponúknutých dvoch programov **A** a **B** ten, ktorý pre zadaný reťazec vypíše ordinálne hodnoty jeho znakov v tvare:  0 => ordinálna hodnota prvého znaku reťazca  1 => ordinálna hodnota druhého znaku reťazca  2 => ordinálna hodnota tretieho znaku reťazca  ...   |  |  | | --- | --- | | ***Program A*** | ***Program B*** | | retazec = input(’Vstupný reťazec: ’)  for i in range(len(retazec)):  print(f’{i} => {ord(retazec[i])}’) | retazec = input(’Vstupný reťazec: ’)  for i in retazec:  print(f’{i} => {ord(i)}’) |   Daný problém rieši program: A |