**Virkņu (string) metodes**

**Virkņu apvienošana**

Lai savienotu virknes, vari izmantot metodi **concat()**. Tā nemaina sākotnējās virknes, bet atgriež savienošanas rezultātu kā virkni. Rezultāts ir jāsaglabā jaunā mainīgajā, pretējā gadījumā tas tiks zaudēts:

|  |  |
| --- | --- |
| Attēls, kurā ir teksts, fonts, ekrānuzņēmums  Mākslīgā intelekta ģenerēts saturs var būt nepareizs. | ***🡨Izmēģini pats!*** |
|  |  |

**Virknes konvertēšana par datu masīvu**

Virkni var pārvērst masīvā, izmantojot metodi **split()**. Atkal ir jāfiksē rezultāts, jo šī metode nemaina sākotnējās virknes. Ņem iepriekšējā piemēra rezultātu: **Sveiks Edvard**! Norādi metodei **split()**, pie kuras vērtības tai vajadzētu sadalīt datus. Katru reizi, kad tā sastopas ar šo vērtību, tā izveidos jaunu datu masīva elementu:

|  |  |
| --- | --- |
| Attēls, kurā ir teksts, ekrānuzņēmums, fonts  Mākslīgā intelekta ģenerēts saturs var būt nepareizs. | ***🡨Izmēģini pats!*** |
| Attēls, kurā ir teksts, fonts, ekrānuzņēmums, cipars  Mākslīgā intelekta ģenerēts saturs var būt nepareizs. |  |

Kā redzi, metode izveido visu elementu masīvu, kas atdalīti ar atstarpi. Elementus var atdalīt ar jebkuru rakstzīmi, piemēram, komatu:

|  |  |  |
| --- | --- | --- |
|  | | ***🡨Izmēģini pats!*** |
| Attēls, kurā ir teksts, ekrānuzņēmums, fonts, cipars  Mākslīgā intelekta ģenerēts saturs var būt nepareizs. | 🡨Skripts ir izveidojis datu masīvu ar sešiem elementiem. | |

**Masīva konvertēšana virknē**

Masīvu var pārvērst virknē, izmantojot metodi **join()**. Lūk, te tas ir parādīts:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
| Attēls, kurā ir teksts, ekrānuzņēmums, fonts, cipars  Mākslīgā intelekta ģenerēts saturs var būt nepareizs. |  |

Ja vēlaties komata vietā izmantot kaut ko citu, norādiet to šādi:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

**Darbs ar īpašībām index un position**

Iespēja uzzināt konkrētas apakšvirknes indeksa vērtību virknē ir ļoti noderīga. Piemēram, ja meklējat konkrētu vārdu failā ar ierakstītu lietotāja ievadi, lai izveidotu apakšvirkni, sākot ar šo indeksu. Lūk, kā atrast virknes indeksu. **indexOf()** metode atgriezīs vienu vērtību — apakšvirknes pirmās rakstzīmes indeksu:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

Rezultāts ir 7. Tas ir tāpēc, ka pirmā “**re**” vērtība parādās vārdā “**are**”, un pirmās rakstzīmes indekss šajā virknē ir ar numuru 7.

Ja indekss netiek atrasts, tiks atgriezta vērtība **-1**. Testē!

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

Alternatīvs veids, kā virknē atrast konkrētu apakšvirkni, ir izmantot metodi **search()**:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

Rezultāts būs 17, kas pirmās rakstzīmes indekss vārdā "fellow". Ja vērtība netiek atrasta, **search()** metode atgriezīs -1.

Funkcija **search()** kā ievadi izmantos regulāras izteiksmes formātu, savukārt **indexOf()** vienkārši izmanto virkni. Funkcija **indexOf()** ir ātrāka nekā **search()**, bet, ja vēlaties meklēt tikai virkni, izmantojiet **indexOf()**. Ja vēlaties virknes modeli, izmantojiet metodi **search()**.

|  |
| --- |
| **Regulārā izteiksme** ir īpaša sintakse virkņu modeļu definēšanai, ko var izmantot, lai aizstātu visas meklēšanas laikā radušās vērtības. |

Kā atceries, **indexOf()** metode atgriež pirmās atrastās vērtības indeksu. Pastāv arī līdzīga metode **lastIndexOf()**, kas atgriež tās pozīcijas indeksu, kurā virknes arguments tiek sastapts pēdējo reizi. Ja **lastIndex()** neatrod atbilstības, tad izvada -1. Piemēram:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

Šī darbība atgriež vērtību **24**. Šī ir pēdējā reize, kad vārds "**re**" parādās dzejolī — otrajā vārdā "**are**".

Dažreiz būs jādara pretēji: nevis jāmeklē indekss, zem kura atrodas virkne, bet gan jānoskaidro, kura rakstzīme atrodas zem noteikta indeksa. Šeit noder **charAt(index)** metode, kas kā argumentu ņem norādīto indeksa pozīciju:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |

Konsole izdrukā vērtību **r**, jo rakstzīme indeksā 10 ir vārda "**red**" rakstzīme "**r**". Ja vaicāsi indeksa pozīciju, kas atrodas ārpus virknes diapazona, tā atgriezīs tukšu virkni. Ja vaicāsi mainīgā **pozicija** tipu, saņemsi vērtību **string**.

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
| Tukša rinda |  |

**Apakšvirkņu izveide**

Apakšvirknes var izveidot, izmantojot metodi **slice(start, end)**. Tā nemaina sākotnējo virkni, bet atgriež jaunu virkni, kas sastāv no atlasītās apakšvirknes vērtībām. Metode pieņem divus parametrus: pirmo indeksu, no kura metode sāksies, un otro - beigu indeksu. Ja izlaidīsiet otro indeksu, metode tiks izpildīta no sākuma indeksa līdz virknes beigām. Beigu indekss netiek iekļauts apakšvirknē. Šeit ir piemērs:

|  |  |
| --- | --- |
|  | ***🡨Izmēģini pats!*** |
|  |  |