Zpracování a parsování textových dat, regulární výrazy, kódování a stringy

Stringy
String neboli řetězec je v Pythonu jedním ze základních datových typů. Jedná se o sekvenci znaků uzavřenou v jednoduchých ('...') nebo dvojitých ("...") uvozovkách. Stringy jsou v Pythonu velice důležité, protože se používají pro reprezentaci textu a zpracování textových dat.

Z pohledu paměti je string v Pythonu neměnný objekt (immutable). To znamená, že jakmile je string vytvořen, nelze ho změnit - jakákoliv metoda, která vypadá, že string mění, ve skutečnosti vytváří nový objekt. Tato vlastnost má význam pro optimalizaci paměti, protože Python může sdílet stejné stringy mezi různými částmi programu.

Co se týče uložení stringů v paměti, Python používá Unicode pro reprezentaci znaků, konkrétně UTF-8 kódování. To znamená, že každý znak může zabírat jeden až čtyři bajty v závislosti na tom, jaký znak reprezentuje. Díky tomu může Python pracovat s jakýmkoliv znakem z různých jazyků včetně češtiny, čínštiny, arabštiny nebo emotikonů. Na rozdíl od některých jiných jazyků, v Pythonu nemusíme řešit rozdíl mezi stringem a znakem - jednotlivý znak je prostě string o délce 1.

Zpracovávání textových dat
Zpracování textových dat zahrnuje různé operace manipulace s textem jako celkem. Jde o transformace a úpravy textových řetězců, které nemění jejich význam, ale mění jejich formu.

<u>Metody</u>
.upper() - přepíše na velká písmena
.lower() - přepíše na malá písmena
.split() - rozdělí řetezce na podřetezce na základě oddělovače
.trim() - odstraní mezery ze začátku a konce stringu
.find() - najde něco ve stringu

a mnoho dalších…

Parsování
Parsování textu je proces, při kterém rozebíráme a analyzujeme text podle určitých pravidel, abychom z něj získali strukturovaná data. Na rozdíl od pouhého zpracování textu, kde většinou jen měníme formu textu (například změna velikosti písmen), parsování se zaměřuje na to, co text znamená a jak lze z něj extrahovat smysluplné informace. Když parsujeme text, snažíme se v něm identifikovat určité vzory nebo struktury a převést je na data, která můžeme dále programově zpracovávat

Regulární výrazy
Regulární výrazy jsou speciální textové řetězce, které slouží jako vzory pro vyhledávání a manipulaci s textem. Používají se k definování pravidel, podle kterých lze v textu hledat určité vzory, nebo ověřovat, zda text odpovídá určitému formátu. Regulární výrazy využívají speciální znaky a konstrukce, které mají specifický význam - například tečka zastupuje libovolný znak, hranaté závorky definují množinu znaků a kvantifikátory určují počet opakování atd… (u maturity budem mít tabulku takže chill)

V Pythonu pracujeme s regulárními výrazy pomocí modulu re, který nabízí funkce jako search(), match(), findall() nebo sub(). Tyto funkce umožňují vyhledávat vzory v textu, ověřovat shodu na začátku textu, najít všechny výskyty vzoru nebo nahradit části textu. Pomocí závorek můžeme také vytvářet skupiny a zachytávat konkrétní části textu, které odpovídají určitým částem vzoru.

Regulární výrazy nacházejí uplatnění v mnoha oblastech programování. Používají se pro validaci vstupních dat (např. kontrola formátu emailu, telefonního čísla nebo PSČ), pro extrakci informací z textu (např. získání všech odkazů z HTML stránky), pro nahrazování textu podle složitých pravidel nebo pro parsování strukturovaných dat z nestrukturovaného textu. I když mohou být zpočátku složité na pochopení, představují mocný nástroj pro každého, kdo pracuje s textovými daty.

Kódování

Kódování textu je způsob, jakým jsou znaky převáděny na binární data, která může počítač zpracovat a uložit. Každý znak, který vidíme na obrazovce (písmena, číslice, symboly), je ve skutečnosti v počítači reprezentován jako sada bitů (jedniček a nul). Kódování určuje, který znak odpovídá které sadě bitů.

Nejstarším a nejjednodušším standardem kódování je **ASCII** (American Standard Code for Information Interchange). ASCII je 7-bitové kódování, které definuje 128 znaků - především anglickou abecedu, číslice a základní symboly. Vzhledem k omezenému počtu znaků v ASCII nelze v něm reprezentovat znaky s diakritikou a znaky jiných písem, jako je čeština nebo čínština.

Pro řešení omezení ASCII vznikl **Unicode**, který přiřazuje každému znaku unikátní kód (tzv. kódový bod). Unicode podporuje znaky ze všech světových jazyků, matematické symboly, emotikony a mnoho dalšího. Nejčastěji používaným způsobem kódování Unicode je UTF-8, které používá proměnlivý počet bajtů (1-4) pro reprezentaci různých znaků. Znaky ASCII v UTF-8 zabírají pouze 1 bajt, což zajišťuje zpětnou kompatibilitu, zatímco znaky jiných písem mohou zabírat více bajtů.

Další známá kódování zahrnují **CP1250** (rozšíření ASCII, které podporuje i češtinu a další středoevropské jazyky, ale obsahuje pouze 256 znaků)

UTF-16 (kódování Unicode, které používá 2 nebo 4 bajty na znak). Správné nastavení kódování je zásadní při práci s textem, zvláště pokud obsahuje znaky mimo základní ASCII, jinak může dojít k problémům jako jsou "rozbité" znaky nebo tzv. mojibake, kdy se text zobrazuje nesprávně.