11. Závěrečná všehochuť

Řetězce:

Speciální znaky v řetězcích: `\"` `\n` `\t` `\\`

Délka řetězce: `StringLength[řetězec]`

Spojení řetězců: `StringJoin[řetězec`$_1$`,...,řetězec`$_n$`]`

Vložení řetězce na zadanou pozici: `StringInsert[původní řetězec,vkládaný řetězec,pozice]`

Výběr části řetězce: `StringTake[řetězec,{od,do}]`

Odstranění části řetězce: `StringDrop[řetězec,{od,do}]`

Hledání v řetězci: `StringPosition[řetězec,hledaný řetězec]`

Počet výskytů: `StringCount[řetězec,hledaný řetězec]`

Nahrazení: `StringReplace[řetězec,pravidlo]` `StringReplace[řetězec,seznam pravidel]`

Pomocí volby `IgnoreCase->True` lze zabránit rozlišování velkých a malých písmen.

Převod řetězce na seznam znaků: `Characters[řetězec]`

Převod řetězce na výraz: `ToExpression[řetězec]`

Převod výrazu na řetězec: `ToString[výraz]`

Práce se soubory:

`Import["název souboru nebo URL"]` `Export["název souboru",exportovaný objekt]`

Podporované formáty lze zjistit pomocí `$ImportFormats` a `$ExportFormats`, další podrobnosti jsou v dokumentaci (*Listing of All Formats*).

Interaktivní vyhodnocování výrazů závislých na parametrech:

`Manipulate[výraz,{t,...},{u,...},...]`

Definiční obory parametrů lze zadat např. takto:

`{t,min,max}` znamená $\texttt{t} \in [\texttt{min}, \texttt{max}]$

`{t,min,max,krok}` znamená $\texttt{t} \in \{\texttt{min}, \texttt{min} + \texttt{krok}, \ldots, \texttt{max}\}$

`{t,{t`$_1$`,t`$_2$`,...}}` znamená $\texttt{t} \in \{\texttt{t}_1, \texttt{t}_2, \ldots\}$

`{t,Locator}` odpovídá dvourozměrnému ovládacímu prvku, který se nachází uvnitř grafiky

Místo `t` lze psát `{t,počáteční hodnota}` nebo `{t,počáteční hodnota,"popis ovládacího prvku"}`.

Integrované databáze:

`CountryData["název státu nebo útvaru", "vlastnost"]`

Seznam států a útvarů lze získat pomocí `CountryData["Countries"]`, `CountryData["Continents"]`, `CountryData["Groups"]`, `CountryData["Oceans"]`.

Seznam všech dostupných vlastností lze získat pomocí `CountryData["Properties"]`. Často používané vlastnosti jsou např. `"Area"`, `"CapitalCity"`, `"Flag"`, `"GDP"`, `"Population"`, `"Shape"`.

`CityData["název města nebo obce", "vlastnost"]`

Identifikace města může mít také tvar `{"název města", "země"}` nebo `{"název města", "oblast", "země"}`. Pomocí `CityData[{All,"země"}]` nebo `CityData[{Large,"země"}]` lze získat seznam všech nebo všech velkých měst v dané zemi.

Seznam všech dostupných vlastností lze získat pomocí `CityData["Properties"]`. Často používané vlastnosti jsou např. `"Coordinates"`, `"Elevation"`, `"Population"`.

K dispozici jsou další databáze: `WordData`, `FinancialData`, `WeatherData`, `AstronomicalData`, `ChemicalData`, … (viz dokumentaci).

Cvičení

1. Rovinná křivka s parametrizací

$$\varphi(t) = ((a+1)\cos t - h\cos((a+1)t), (a+1)\sin t - h\sin((a+1)t)),$$

kde $t \in [0, 2\pi]$, se nazývá epitrochoida (a, h jsou parametry). Zobrazte epitrochoidu příkazem `ParametricPlot` a pomocí `Manipulate` umožněte uživateli, aby mohl volit hodnoty $a \in \{1, 2, \ldots, 10\}$ a $h \in [0, 4]$.

2. Naprogramujte funkci, která zadaný řetězec zašifruje pomocí Caesarovy šifry, tj. nahradí znaky podle schématu `a->b`, `b->c`, ... , `z->a` (předpokládejte, že text neobsahuje háčky a čárky; můžete se omezit pouze na malá písmena abecedy). Naprogramujte také inverzní funkci pro dešifrování textu.

Návod: Seznam písmen anglické abecedy lze získat pomocí `CharacterRange["a","z"]`. Pomocí něj zkonstruujte vhodný seznam přepisovacích pravidel a použijte `StringReplace`.

3. Cykloida je dráha pevně zvoleného bodu na obvodu kružnice, která se valí po přímce. Uvažujme speciálně jednotkovou kružnici, jejíž střed se v čase $t \geq 0$ nachází v bodě $[t, 1]$ (tj. kružnice se valí po přímce $y = 0$). Bod kružnice, který měl na počátku souřadnice $[0, 0]$, je v čase t na pozici $[t - \sin t, 1 - \cos t]$. Vyrobte animaci ve formátu GIF, která bude zobrazovat valící se kružnici s bodem, který za sebou zanechává stopu ve tvaru cykloidy. Omezte se např. na časový interval $t \in [0, 8\pi]$.

4. Steinerův bod trojúhelníka je definován jako bod, pro který je součet vzdáleností od vrcholů trojúhelníka minimální.

a) Naprogramujte funkci, která dostane souřadnice vrcholů trojúhelníka a vrátí jeho Steinerův bod; k jeho nalezení použijte funkci `NMinimize` (má stejnou syntaxi jako `Minimize`, hledá minimum numericky).

b) Pomocí `Manipulate` vytvořte program, ve kterém bude možné pomocí lokátorů zadávat vrcholy trojúhelníka, a program bude zobrazovat příslušný Steinerův bod spojený úsečkami s vrcholy trojúhelníka.

c) Pokuste se odpovědět na následující otázky: Spojíme-li Steinerův bod úsečkami s vrcholy trojúhelníka, jaké úhly tyto úsečky svírají? Pro které trojúhelníky platí, že Steinerův bod splývá s některým vrcholem trojúhelníka?

5. Seřaďte všechny země v databázi `CountryData` podle očekávané délky života (`"LifeExpectancy"`), pomocí `TableForm` výsledek přehledně zobrazte. (K setřídění použijte funkci `Sort`; pokud ji aplikujete na seznam dvojic, dojde k seřazení podle prvních prvků.)

6. Pomocí `CountryData["název země", "BorderingCountries"]` lze získat seznam zemí sousedících se zadanou zemí. Zjistěte, které státy mají největší počet sousedů.

7. Použijte databázi `CityData` k vyřešení následujících úloh:

a) Kolik obcí z ČR je zaneseno v databázi `CityData`?

b) Naprogramujte funkci, která vrátí názvy všech obcí v ČR s aspoň n obyvateli.

Návod: `CityData["název města", "Population"]` vrací počet obyvatel včetně jednotek („people“); číselný údaj bez jednotek lze získat použitím
`QuantityMagnitude[CityData["název města", "Population"]]`.

c) Pomocí `Manipulate` vytvořte program, které umožní uživateli měnit hodnotu n a zobrazí mapu ČR, na které budou vyznačena města s aspoň n obyvateli.

Návod: `CountryData["CzechRepublic", "Polygon"]` vrací mnohoúhelník představující mapu ČR.
`CountryData["název města", "Coordinates"]` vrací dvojici (zeměpisná šířka, zeměpisná délka); před zobrazením příslušného bodu je potřeba zaměnit pořadí souřadnic.