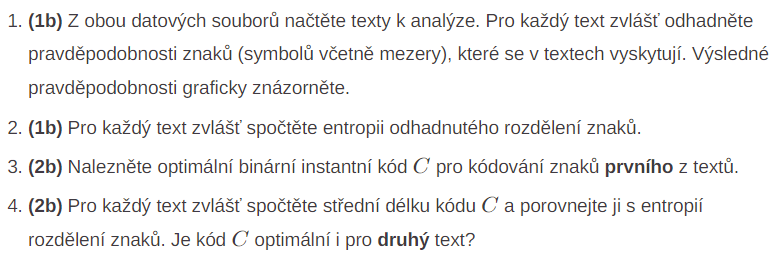
**Domácí úkol č. 1**

Karel Vrabec

**Zadání**



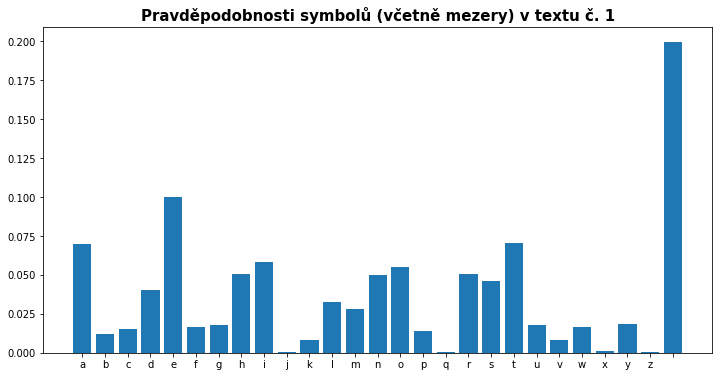
**Řešení**

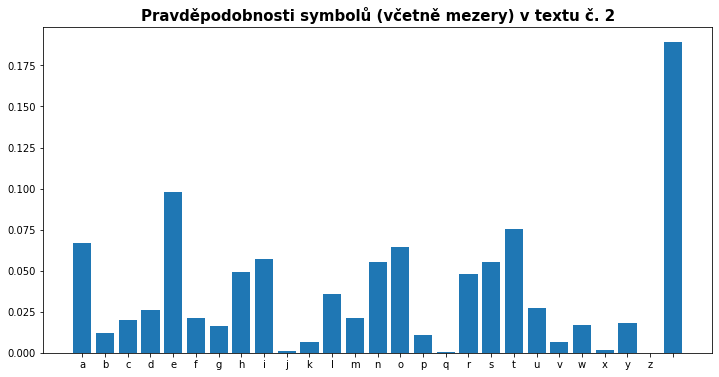
Testované soubory: **005.txt** (jako **text č. 1**), **017.txt** (jako **text č. 2**)

Programovací jazyk: **Python**

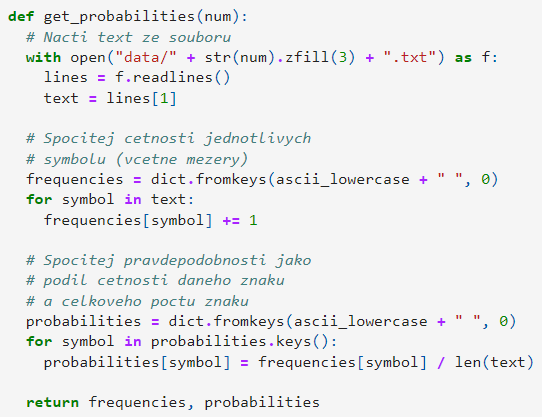
Kódy k dispozici v přiloženém archivu **source.zip** (soubor **assignment.ipynb**)

**1.** Načetl jsem texty z obou datových souborů. Pro každý z nich jsem vypočítal **četnosti** jednotlivých znaků (včetně mezery). Následně jsem tyto četnosti vydělil celkovým počtem znaků v daném textu, čímž jsem získal **pravděpodobnosti** těchto znaků. Tyto odhadnuté pravděpodobnosti jsem graficky znázornil v následujících grafech. Písmena **j**, **q**, **x** a **z** jsou v obou úryvcích nejméně používanými znaky a **mezera** naopak tím nejpoužívanějším.





Příslušná okomentovaná ukázka kódu (načítání textů, výpočet četností a pravděpodobností):

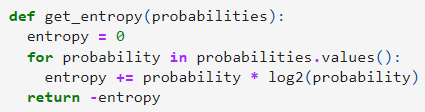


**2.** Získané pravděpodobnosti z předchozího bodu jsem použil pro výpočet **entropie** u obou textů. Vycházel jsem z **definice 5.1** (pro výpočet **entropie diskrétní náhodné veličiny**) uvedené ve studijním textu.

Entropie (pro text č. 1):

Entropie (pro text č. 2):

Příslušná ukázka kódu (metoda pro výpočet entropie):

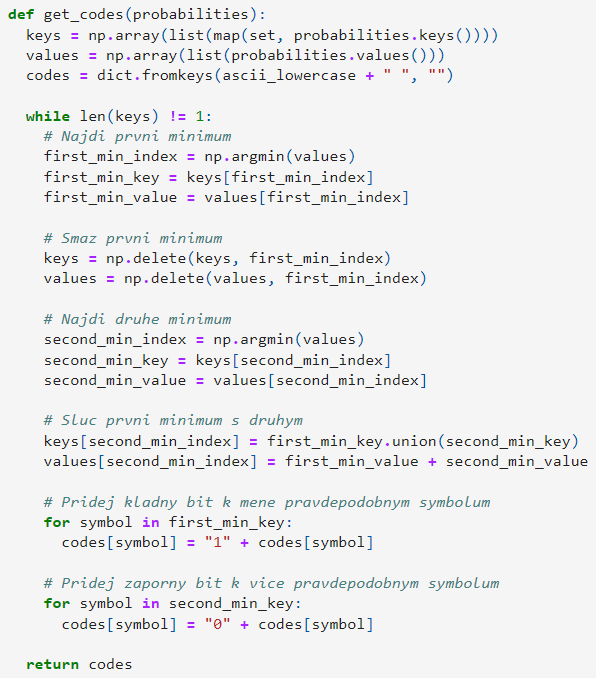


**3.** Nejdříve jsem implementoval vlastní algoritmus pro sestrojení optimálního binárního instantního **Huffmanova kódu** (okomentovaná ukázka níže). Tento algoritmus jsem použil pro vygenerování kódu *C\_1* pro **text č. 1**. V následující tabulce se nachází podrobné údaje (včetně vygenerovaného kódu) k jednotlivým znakům z **textu č. 1** (poslední prázdné políčko indikuje mezeru). Ze struktury algoritmu je patrné, že **nejvíce pravděpodobné** znaky obdrží **krátké** kódy, zatímco **nejméně pravděpodobné** znaky obdrží naopak **dlouhé** kódy.

Obsah obrázku stůl

Popis byl vytvořen automaticky

Příslušná okomentovaná ukázka kódu (algoritmus pro sestrojení Huffmanova kódu):



**4.** Získaný kód a pravděpodobnosti z předchozích bodů jsem použil pro výpočet **střední délky kódu** u obou textů. Vycházel jsem z **definice 6.3** (pro výpočet **střední délky kódu**) uvedené ve studijním textu. Všechny získané střední délky kódu jsou **zdola omezeny entropií**, od které se mohou vzdálit **maximálně o 1**.

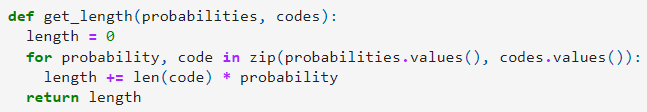
Střední délka kódu *C\_1* (pro text č. 1):

Střední délka kódu *C\_1* (pro text č. 2):

Porovnání s entropií (pro text č. 1):

Porovnání s entropií (pro text č. 2):

Příslušná ukázka kódu (metoda pro výpočet střední délky kódu):



Algoritmus z předchozího bodu vytvoří binární Huffmanův kód, který je **vždy optimální** pro daný text (resp. pravděpodobnosti jednotlivých symbolů na vstupu). Huffmanův kód *C\_1* je **optimální** pro **text č. 1**. Tento kód má tudíž **nejmenší možnou střední délku**. V případě textu č. 2 to tak platit nemusí. Z tohoto důvodu jsem vygeneroval binární Huffmanův kód *C\_2* pro **text č. 2**. Střední délka kódu vyšla **optimálnější** než u předchozího kódu. To znamená, že **kód *C\_1* není optimální pro text č. 2**.

Střední délka kódu *C\_2* (pro text č. 2):

Porovnání s entropií (pro text č. 2):

V následující tabulce se nachází podrobné údaje (včetně vygenerovaného kódu) k jednotlivým znakům z **textu č. 2** (poslední prázdné políčko indikuje mezeru).

