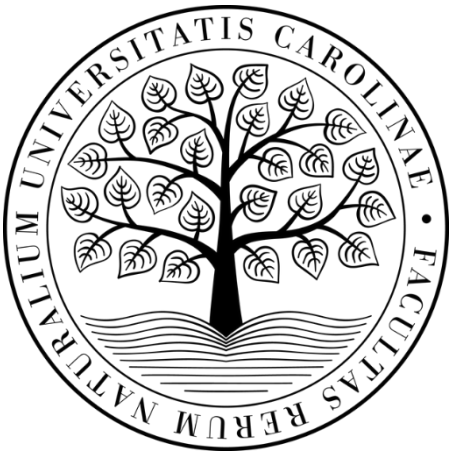
Univerzita Karlova

Přírodovědecká fakulta



Převod textu do Morseovy abecedy a zpět

Úvod do programování

Miroslav Hrubeš

3. BGEKA

Světice, 2022

Zadání

Vytvořte program, který bude překládat text do Morseovy abecedy a zpět. Součástí úlohy bude kromě aplikace také dokumentace, o rozsahu 4-5 stran ve formátu PDF, která bude obsahovat následující:

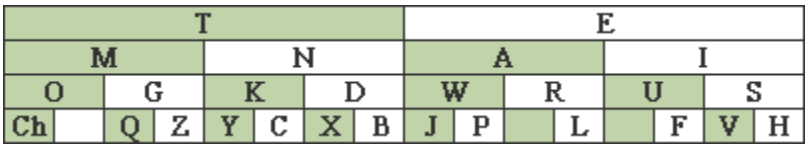
- rozbor problému
- existující algoritmy
- popis zvoleného algoritmu
- strukturu programu (datové struktury, metody,…)
- popis vstupních/výstupních dat
- problematická místa
- možná vylepšení

Program bude považována za nefunkční pokud:

- při zpracování dat dojde k pádu (runtime chyby,…)
- vrací špatné výsledky
- neřeší možné singulární případy

Rozbor problému

Morseova abeceda (někdy lidově nazývána též „morseovka“) je způsob kódování znaků latinky a číslic, popř. dalších speciálních znaků. Znaky jsou kódovány v sekvenci různě vydávaných signálů, které jsou podle délky trvání obecně definovány jako „tečka“ (·) a „čárka (−), přičemž obecně platí, že délka signálu jedné čárky by měla odpovídat dvojnásobné či trojnásobné délce signálu jedné tečky. Různé seskupení těchto signálů definuje různé znaky. Důležitou roli hraje mezera mezi signály, resp. chvíle, kdy je vysílání signálu přerušeno. Typy mezer lze definovat dle délky. Nejkratší mezera od sebe vzájemně odděluje signály teček a čárek. Delší mezera od sebe odděluje dvě konkrétní písmena, resp. jeden soubor signálů odpovídající jednomu písmenu a druhý (začínající) soubor signálů odpovídající dalšímu písmenu. V případě psané formy, jsou mezery mezi písmeny nahrazeny lomítkem. Pokud půjde o konec slova, používají se na konci posledního písmena lomítka dvě a v případě konce věty tři. Abeceda je pro svoji jednoduchost a praktičnost velice populární, zejména u skautských a klubových aktivit venku. Existuje několik pomůcek pro rozluštění kódu. Jedna z nich je zobrazena na Obr. 1



Obr. 1: Pomůcka pro odkódování Morseovi abecedy

Tato abeceda je pojmenovaná po americkém vynálezci Samuelu F. B. Morseovi (Obr. 2). Její vznik souvisí se vznikem a rozvojem telegrafie. Bylo zapotřebí přijít s univerzálním jazykem, kterým se budou moci vzájemně dorozumívat osoby skrz telegraf, a který bude fungovat jen na principu vysílání elektrických impulsů (signálů) a ticha mezi nimi. Právě Samuel F. B. Morse vytvořil okolo roku 1837 předchůdce dnešní abecedy. Tento předchůdce měl pouze vysílat číslice a používat číselník k vyhledání každého slova podle odeslaného čísla. Morseův spolupracovník Alfred Vail však kód v roce 1840 rozšířil o písmena a speciální znaky. Při přiřazování signálům různým znakům se Vail opřel o písmena v angličtině a o jejich četnosti používání tak, aby nejčetnější písmena odpovídala co nejmenšímu počtu signálů. Vznikla tak tzv. Americká (Železniční) Morseova abeceda. Ta byla během dalších let ještě upravena například některá písmena, ale především „prodloužením“ signálu čárky z délky dvou teček na délku tří teček. Tyto úpravy daly vzniku tzv. Mezinárodní Morseově abecedě.

Existující algoritmy

Existující v zásadě dva přístupy pro algoritmizaci morseovky. První přístup funguje na principu „stromu“. Tím je myšleno, že se postupuje znak po znaku. Nejprve dojde u prvního znaku o rozlišení, zda se jedná o tečku či čárku a podle toho definuje písmeno. V případě, že je první znak následován druhým znakem, je opět vyhodnoceno, zda se jedná o tečku či čárku a písmeno se tak posune na další pozici nahoru, resp. dolů. Tento proces se opakuje, dokud není soubor znaků ukončen delší mezerou či lomítkem. Princip je využíván v případě praktického a intuitivního učení abecedy za pomocí „stromového“ klíče, který zobrazuje Obr. 1. V opačném případě se dá takto také postupovat a to ve smyslu zapisování cesty znaků ve směru daného písmene.

Jako druhý přístup by se dalo označit vytvoření slovníku, kde bude každý soubor znaků odpovídat jednomu písmenu. Algoritmus nebude při přímém překládání postupovat znak po znaku, ale po souboru znaků oddělených delší mezerou, resp. lomítkem, a uložených do proměnné, bude přímo porovnávat tuto proměnou se slovníkem a hledat odpovídající

písmeno. Proces bude probíhat stejně, pokud bude algoritmus porovnávat písmena se slovníkem souborů znaků.

Popis zvoleného algoritmu

Z výše zmíněných dvou algoritmů byl vybrán algoritmus používající slovník a to díky jeho jednoduchosti a přehlednosti, neboť program se bude zaměřovat pouze na překlad a není jeho cílem abecedu uživatele učit.

Pseudokód algoritmu, resp. funkce překládající text do Morseovy abecedy je v programu následující:

Definuj funkci, jejíž vstupy budou slovník morseovka-latinka a proměnná vstupního textu a:

vytvoř prázdnou proměnnou typu *string* jménem ‚outp‘

pro proměnnou ‚i‘, která začíná na 0 a postupně se o 1 zvyšuje, než dosáhne počtu znaků vstupního textu,

se zeptej, jestli znak vstupu na pozici i neodpovídá lomítku

a pokud ano zeptej se, jestli znak vstupu na pozici i-1 neodpovídá mezeře

a pokud ano, nech program běžet dál

jinak do proměné ‚outp‘ přičti lomítko

nebo se zeptej, jestli znak vstupu na pozici i neodpovídá tečce

a pokud ano, do ‚outp‘ přičti dvě lomítka

jinak se pro všechny klíče ve slovníku morseovka-latinka zeptej,

jestli znak vstupu na pozici i odpovídá klíči

a pokud ano přičti do ‚outp‘ hodnotu odpovídající tomuto klíči a také lomítko

nebo jestli se znak vstupu vůbec nachází ve slovníku morseovka-latinka

a pokud nenachází, vypiš to do terminálu a ukonči program

Jako výstup funkce vrať ‚outp‘

Pseudokód algoritmu, resp. funkce překládající Morseovy abecedu do latinky je obdobný:

Definuj funkci, jejíž vstupy budou slovník latinka-morseovka a proměnná vstupního textu a:

vytvoř prázdnou proměnnou typu *string* jménem ‚char‘

vytvoř prázdnou proměnnou typu *string* jménem ‚outp‘

pro proměnnou ‚i‘, která začíná na 0 a postupně se o 1 zvyšuje, než dosáhne počtu znaků vstupního textu,

se zeptej, jestli znak vstupu na pozici i neodpovídá lomítku nebo mezeře

Pokud ano přičti ho do ‚char‘.

Jinak

se pro všechny klíče ve slovníku latinka-morseovka zeptej

jestli se ‚char‘ rovná klíči,

a pokud ano přičti do ‚outp‘ hodnotu odpovídající tomuto klíči.

Nebo jestli se vůbec v tomto slovníku nachází

a pokud nenachází, vypiš to do konzole a ukonči aplikaci

definuj ‚char‘ jako prázdnou proměnnou

se zeptej, jestli znak vstupu na pozici i-1 odpovídá lomítku

a pokud ano zeptej se, jestli i znak na pozici i-2 odpovídá lomítku

a pokud ano odstraň z ‚outp‘ poslední znak a přičti k němu tečku s mezerou.

a pokud ne, přičti k ‚outp‘ mezeru.

Zeptej se, jestli je délka ‚char‘ větší než nula

a pokud ano přičti do ‚outp‘ hodnotu klíče ‚char‘.

vytvoř proměnou ‚c’, jejíž hodnota bude znak ‚outp‘ na pozici 0 převedená na velké písmeno

v ‚outp‘hodnotu na pozici 0 nahraď právě jednou proměnnou ‚c‘

pro proměnnou ‚j‘, která začíná na 0 a postupně se o 1 zvyšuje, než dosáhne počtu znaků ‚outp‘

se zeptej, jestli hodnota ‚outp‘ na pozici j odpovídá tečce

a pokud ano, pokus se dosadit do ‚c‘ hodnotu ‚outp‘ na pozici j+2, která bude velkým písmenem a v ‚outp‘ nahraď hodnotu na pozici 0 proměnnou ‚c‘ právě jednou.

očekávej IndexError a v případě, že nastane, nech program běžet dál.

jako výstup funkce vrať ‚outp‘

Struktura programu

Vstupní data jsou na počátku programu zadána uživatelem do externího souboru ‚input.txt‘, který se musí nacházet ve složce spolu s programem. Pokud uživatel zadá vstup ve formě morseovky, měl by dodržovat pravidla ohledně oddělování souboru znaků určitým počtem lomítek. V případě napsání vstupního textu latinkou se může jevit jako problémová diakritika. Program ovšem po nahrání dat a jejich následné úpravě (oříznutí a převedení velkých písmen na malá písmena) dekóduje data tak, aby obsahovala znaky bez diakritiky. Dále program vytvoří reverzní slovník abecedy (z před už předem napsaného slovníku) tak, aby se na pozici klíčů nacházeli soubory teček a čárek a na pozici hodnot se nacházeli písmena a čísla.

Program dále zjišťuje, jestli je první znak u vstupních dat tečka nebo čárka a pokud ano, spustí se funkce překladače z Morseovy abecedy do latinky, která na vstupu obsahuje daný slovník a vstupní data z textového souboru. Tato funkce již byla popsána v předchozí kapitole. Pokud by tomu tak ovšem nebylo, spustí se obdobná funkce, která překládá text z latinky do Morseovy abecedy. Jako vstupní data funkce jsou slovník tohoto typu překladu a vstupní data z textového souboru. Výstup dané funkce je poté spolu se vstupními daty vytisknut do terminálu a samotný výstup je pak následně zapsán do výstupního textového souboru ‚output.txt‘

Případné vylepšení programu by mohlo být v lepší kontrole toho, kterou překladatelskou funkci použít, než je stávající kontrola, zda je první znak vstupu tečka či čárka. Na tento výběr překladače by se také mohl program uživatele zeptat před samotným překladem. Stejně tak alternativou externích vstupních dat může být přímé zadávání dat uživatelem do programu anebo naopak slovník by se místo v samotném programu mohl nacházet v externím souboru. Další možnosti vylepšení nebo úpravy jsou spíše formátové záležitosti k diskuzi. Neboť některé problémy, jako je velikost písmen na začátku věty či ukončení věty tečkou a mezerou tento program sice řeší, ale z pohledu algoritmu se jedná spíše o nadstavbu, která překladače zatěžuje a jejich kód činí mnohem méně přehledným, a tento způsob nadstavby či nadstavba samotná nemusí danému uživateli vůbec vyhovovat.

Zdroje

http://baronjh.sweb.cz/tabulky/morse.abeceda.htm

https://en.wikipedia.org/wiki/Morse_code

https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.1677-1-200910-I!!PDF-E.pdf