Programowanie

Tymoteusz Małkowski

**Translator**

Profesor: dr. Zbigniew Marszałek

Gliwice, 12 stycznia, 2020

1. Wprowadzenie

Ten dokument zawiera raport odnośnie programu stworzonego na laboratoria z programowania.

Zostało wybrane następujące zadanie:  
Translator. Napisz program który “tłumaczy” plik tekstowy poprzez zamianę słów na słowa znajdujące się w osobnym pliku będącym biblioteką. Słownik powinien być odczytany z pliku raz a następnie trzymany w pamięci.

2. Rozwiązanie problemu

Program potrzebuje dwóch wejściowych plików tekstowych aby działać poprawnie. Są to: plik ze słownikiem oraz plik ze słowami które zostaną przetłumaczone.

Program na początku sprawdza czy plik z bazą danych istnieje. Jeśli wynik jest pozytywny, pliki zostaje otwarty i słowa są wczytywane w pętle z bazy danych, w innym przypadku odpowiednia wiadomość zostaje ukazana. Następnie program sprawdza czy plik zawierający słowa do przetłumaczenia istnieje. Jeśli wynik jest negatywny, zostanie wyświetlona odpowiednia wiadomość. Po otrzymaniu dwóch pozytywnych wyników program stworzy plik tymczasowy który będzie przetrzymywać wszystkie tłumaczenia, podczas gdy program nadal działa i zamienia słowa z pliku wejściowego przy użyciu słów ze słownika. Gdy pętla zostanie zakończona, pamięć przeznaczona dla słownika zostanie zwolniona. W tym czasie oryginalny plik ze słowami do przetłumaczenia zostanie usunięty a w zamian za to plik tymczasowy zostanie nazwany tak jak przed chwilą usunięty plik.

3. Specyfikacja wewnętrzna

Funkcja main() działa w dany sposób:  
- Używa funkcję load\_dictionary() aby załadować słownik w strukturę

- Jeśli poprzedni krok był pozytywny, otwiera plik który zostanie przetłumaczony

- Jeśli poprzedni krok był pozytywny, tworzy plik tymczasowy do tłumaczenia

- Jeśli poprzedni krok był pozytywny, main() zaczyna działać nad główną pętlą.

- Po zakończeniu pętli, oryginalny plik text.txt zostaje usunięty, zaś tymczasowy plik zostaje nzwany text.txt.

Główna pętla funkcji działa w określony sposób:

1. Zanim zacznie główną pracę programu, zostają utworzone dwa array’e o wielkości największego dozwolonego słowa.  
2. Array’e są czyszczone.

3. Do pierwszego array’u wpisywane są słowa z pliku. Jeśli EOF zostanie napotkany, pętla zostaje zakończona.

4. Załadowany danymi ze struktury z tłumaczeniami, słowo wejściowe i drugi array są podane dalej do get\_translation(), gdzie jeśli:

- zostanie zwrócony stan true, tłumaczenie dla danego słowa zostało znalezione I załadowane do drugiego array’u -> tłumaczenie jest zapisane w pliku tymczasowym

- zostanie zwrócony stan false, tłumaczenie dla danego słowa nie zostało znalezione

-> oryginalne słowo zostaje zapisane w pliku tymczasowym

5. Na końcu pętli zostaje użyty copy\_white\_chars(), który kopiuje wszystkie białe znaki do pliku tymczasowego.

6. Powrót do kroku 1.

4. Specyfikacja zewnętrzna

Program został napisany tak, aby działał w terminalu UNIX. Może zostać uruchomiony przy użyciu GUI lub polecenia w terminalu *./NameOfFlie*.

Przykład:  
Pierwsze zmieniamy ścieżkę na tą, w której znajduje się folder zawierający plik wykonywalny przy użyciu komendy cd:

*cd /Users/admin1/Home/project*

Następnie wykonujemy polecenie: *chmod +x project*

Ta komenda daje systemowi permisję do otworzenia pliku wykonywalnego.

Następnie możemy uruchomić program przy użyciu komendy *./project.*

Program wymaga do poprawnego działania, aby w pliku words.txt był pusty znak pomiędzy angielskim oraz polskim słowem. Przykład poniżej:  
*<słowo1ang> <słowo2pol>  
<słowo2ang> <słowo2pol>*

Każde kolejne słowo znajdujące się w pliku text.txt musi znajdować się w następnym wierszu, jak na przykładzie poniżej:  
*<słowo1ang>*

*<słowo2ang>*

Dla poprawnych argumentów program przetłumaczy każde słowo wejściowe oraz zapisze w pliku przetłumaczone słowa zamiast nich. Jeśli podczas tego procesu wystąpi jakiś problem, pojawi się powiadomienie o problemie mówiące nam co zaszło nie tak.

5. Testowanie programu

Działanie z poprawnymi plikami wejściowymi

Translator.sh words.txt text.txt

Zawartość pliku words.txt:  
cat kot

dog pies

tea herbata

paper papier

sugar cukier

water woda  
  
Zawartość pliku text.txt:  
cat

dog

tea

paper

sugar

water  
  
Program zwraca:

kot

pies

herbata

papier

cukier

woda

**Program działa poprawnie.**