Specyfikacja funkcjonalna programu "Wyprodukuj tekst".

Projekt na zajęcia: *"Języki i Metodyka Programowania"*
wydz. Elektryczny Politechniki Warszawskiej, rok akademicki 2014/2015
wykonał: Paweł Drapiewski

1 Opis ogólny programu

Program służył do tworzenia zdań na podstawie zestawu plików tekstowych zawierających przykładowe zdania, lub na podstawie istniejącej bazy danych.

Użytkownikiem docelowym programu jest osoba która ma pewną wiedzę komputerową, gdyż do obsługi programu jest wymagane użycie powłoki tekstowej. Z założenia główną grupą odbiorców są programiści chcący zastosować ten rodzaj "sztucznej inteligencji" w swoich programach.

2 Opis funkcjonalności

Program uruchamiany jest przy pomocy komendy **prodtekst** *parametry* [słowo_rozpoczynajace], gdzie w przypadku nie podania żadnego słowa rozpoczynającego lub braku tego słowa w programowej bazie, zostanie ono wylosowane z istniejących.

Dostępne parametry i opisy funkcji z nimi związane:

- **-p** plik1 [,plik2, ...] - powoduje dołączenie pliku tekstowego będące wzorcem dla generatora tekstu. Może być ich kilka, jeden po drugi rozdzielone znakiem przecinka.
- **-b** baza_danych1 [, baza_danych2, ...] - daje możliwość podłączenia pliku z gotową bazą danych, zawierającą zbiór n-gramów.
- **-n** nGram - jest to liczba, określająca na jakiej bazie n-gramów będzie tworzony tekst. Parametr ten ma wysoki wpływ na merytoryczną poprawność produkowanego tekstu. Wartością domyślą jest liczba 2.
- **-d** IleSłów - określa ile maksymalnie słów ma zawierać produkowany tekst. Domyślnie jest to 1024 słów.
- **-a** IleAkapit - określa ile maksymalnie akapitów ma zawierać tekst wynikowy. Domyślnie jest to wartość nie ograniczona.
- **-z** plikwynikowy - pozwala na określenie pliku do którego nastąpi zapisanie wynikowego tekstu lub statystyk. Domyślnie program utworzy/nadpisze plik prodtekst_wynik.txt
- **-x** plik_do_zapisu_bazydanych - pozwala zapisać bazę n-gramów do pliku, w celu późniejszego użycia w tym programie. Domyślnie program nie podejmuje tej akcji.
- **-t** {p, s} - określa tryb w jakim program działa. Gdzie *p* jest trybem domyślnym i oznacza produkcję tekstu. A *s* jest trybem statystyki, czyli zajmuję się budowaniem statystki.

3 Format danych i struktura plików

3.1 Struktura katalogów

Program jest umieszczony w jednym katalogu z bazami danych w postaci plików binarnych, oraz dodatkowego pliku tekstowego niezależnego od użytkownika przechowującego logi programu.

3.2 Struktura plików

- **Pliku wejściowego**
  Jest to plik tekstowy kodowany w UTF-8, reprezentujący zestaw form zdaniowych, które posłużą za wzorzec dla generatora tekstu.

- **Pliku wyjściowego**
  Plikiem wyjściowym jest plik tekstowy kodowany w UTF-8, zawierający utworzony przez program tekst.

- **Pliku przechowującego logi programu**
  Jest to plik tekstowy zapisany w standardzie CSV. Przechowuje on informacje o błędach jakie wystąpiły podczas działania programu. Każda z takich informacji jest rozdzielona znakiem nowej linii. Pojedyncza informacja jest reprezentująca przez następujące pola: data i godzina wystąpienia błędu, kod błędu, treść błędu, plik(i) wejściowe programu, plik(i) wyjściowe programu. Same pola są rozdzielone średnikami, a dodatkowo pola tekstowe są opakowane w znaki cudzysłowu.

- **Pliku przechowującego bazę danych**
  Baza danych programu będzie przechowywana w pliku binarnym. Początek pliku stanowią informację o przechowywanych danych takie jak: typ przechowywanych n-gramów (czyli czy są to 2-gramy, 3-gramy itp) oraz ilość przechowywanych n-gramów. A za tymi informacjami będą umieszczone kolejne struktury danych reprezentujące pojedynczy n-gram.

4 Scenariusz działania programu

4.1 Scenariusz ogólny

1. Uruchom i przeanalizuj parametry wejściowe pod względem poprawności, błędy zasygnalizuj informacją wypisaną na ekran oraz do logów.

2. Przeanalizuj zadane pliki i dodaj przeanalizowane dane do swojej bazy danych. Pliki niespełniające wytycznych zignoruj, a informacje o tym fakcie wypisz na ekran oraz do logów programu.

3. Zależnie od wybranego trybu:

   (a) Produkuj tekst

   (b) Utwórz statystyki

4. Zapisz dane do pliku, ewentualnie wypisz je na ekran

5. Zakończ program

5 Testowanie

Testowanie programu odbędzie się przy pomocy standardowych narzędzi dla programów pisanych w C dla programów konsolowych, czyli m.in. przy pomocy programu o nazwie "valgrind", który służy do badania wycieków pamięci. A przede wszystkim testowanie będzie oparte o wielokrotne uruchamianie programu na różnych zestawach danych (w szczególności gotowych dziełach literackich), w celu wykrycia błędów programu, a w przypadkach błędów "ukrytych" (czyli ujawniających się podczas działania programu) zostanie użyty "GNU Debbuger".