**BIESZCZADZKI TOUR**

Specyfikacja funkcjonalna

Warszawa, 06.11.2019

1. **Wstęp**
   1. **Cel dokumentu**

Celem niniejszego dokumentu, będącego specyfikacją funkcjonalną projektu „Bieszczadzki Tour”, realizowanego na potrzeby zadania projektowego z przedmiotu Algorytmy i Struktury Danych, jest wprowadzenie użytkownika końcowego w jego problematykę i możliwości. Ma on za zadanie również pełne poinstruowanie użytkownika w jaki sposób odpowiednio i bezproblemowo korzystać z omawianego programu.

* 1. **Cel projektu**

Celem projektu jest pomoc zakłopotanym studentom w odnalezieniu optymalnej trasy pomiędzy wybranymi przez nich punktami turystycznymi na mapie Bieszczad. Stworzony program ma umożliwiać wygenerowanie trasy przez wszystkie wybrane przez studentów miejsca w możliwie najkrótszym czasie, z uwzględnieniem kosztów podróży.

* 1. **Użytkownik końcowy**

Użytkownikiem końcowym programu „Bieszczadzki Tour” jest grupa studentów wybierających się na wyprawę po Bieszczadach, którzy z wykorzystaniem programu mają możliwość zoptymalizowania swojej wędrówki. Użytkownikiem końcowym jest również mgr. Inż. Paweł Zawadzki, prowadzący zajęcia z Algorytmów i Struktur Danych w semestrze 2019Z, który to na podstawie działania programu dokona jego oceny.

1. **Uruchomienie programu**

W celu uruchomienia programu, na lokalnym komputerze należy otworzyć wiersz poleceń, a następnie przejść do folderu z projektem, podając ścieżkę do niego w pokazany poniżej sposób:

*cd C:\Maciej\_general\Maciej studia\3 semestr\Algorytmy i Struktury Danych\BieszczadzkiTour*

Po przejściu do powyższego folderu należy uruchomić program, podając jednocześnie argumenty do niego, zgodnie z poniższym schematem:

java -jar dist\bieszczadzkitour.jar path\to\data.txt ID\_miejsca path\to\wishlist.txt,

gdzie path\to\data.txt to ścieżka do pliku, w którym użytkownik powinien umieścić dane wejściowe dla programu (nazywanego dalej plikiem konfiguracyjnym), ID\_miejsca , czyli identyfikator miejsca rozpoczęcia podróży, zgodny z informacjami zawartymi w pliku konfiguracyjnym oraz **opcjonalnie** path\to\wishlist.txt, czyli ścieżkę do pliku, w którym użytkownik określa miejsca, które chce odwiedzić (nazywanego dalej listą życzeń). Formatowanie danych wejściowych dla programu jest omówione szczegółowo w punkcie trzecim niniejszego dokumentu.

1. **Dane wejściowe**

Dane wejściowe dla programu, to wspomniane w poprzednim punkcie argumenty, które podajemy przy uruchamianiu. Pliki będące argumentami do programu powinny być plikami tekstowymi, które muszą posiadać określone formatowanie zawartości, aby program mógł działać w pełni poprawnie. Poniżej szczegółowo omówiono poszczególne argumenty.

**3.1 Plik konfiguracyjny**

W tym pliku tekstowym (należy wykorzystać rozszerzenie .txt) użytkownik powinien umieścić wszelkie informacje (unikatowy numer ID, nazwę oraz opis miejsca) na temat punktów, które rozpatrujemy podczas działania programu oraz informacje na temat czasów (w obu kierunkach) i cen przejść pomiędzy poszczególnymi punktami. Plik powinien być zgodny z poniższym schematem.

**3.2 ID\_miejsca**

Jest to ciąg znaków, zgodny z analogicznym polem w pliku konfiguracyjnym, który powinien jednoznacznie identyfikować i określać miejsce rozpoczęcia i jednocześnie zakończenia podróży.

**3.3 Lista życzeń**

W tym pliku tekstowym (ponownie należy wykorzystać rozszerzenie .txt) użytkownik określa jednoznacznie, za pomocą pola ID\_miejsca, miejsca, które chce odwiedzić. Formatowanie pliku powinno być zgodne z poniższym schematem.

1. **Dane wyjściowe**

W wyniku działania programu tworzony jest tekstowy plik wynikowy o nazwie result.txt, w którym przedstawiona jest pełna, najkrótsza i możliwie najtańsza trasa z wyliczonym czasem oraz kosztem wędrówki. Plik tworzony będzie w folderze projektu, w podfolderze output, a jego formatowanie będzie zgodne z poniższym schematem.

1. **Scenariusz uruchomienia**

Po uruchomieniu programu zgodnie z punktem 2 niniejszej specyfikacji, w  
przypadku poprawnego formatowania plików, działanie programu zakończy się, a w podfolderze output w folderze projektowym ukaże się nowy plik wynikowy result.txt.

1. **Opis sytuacji wyjątkowych**

W przypadku podania nieprawidłowych argumentów przy uruchamianiu programu, w powłoce tekstowej, w której uruchamiany jest program, wyświetlony zostanie stosowny komunikat o niepoprawnym sformatowaniu argumentów z informacją, w którym argumencie wystąpił błąd.

1. **Testowanie**

Program będzie testowany poprzez podawanie jako argumenty różnych zestawów danych -- zarówno poprawnych, jak i niepoprawnych. W przypadku podania danych o niepoprawnym formatowaniu sprawdzana będzie funkcjonalność w jaki sposób program informuje nas o błędach w argumentach -- czy komunikat o błędzie jest wyświetlony prawidłowo oraz czy jednoznacznie informuje nas jaki błąd wystąpił. W przypadku testowania poprawnego działania programu, przygotowane zostaną zestawy danych wejściowych o poprawnym formatowaniu oraz o wiadomym poprawnym wyniku działania programu. Po wykonaniu programu, dane z pliku wynikowego zostaną porównane z wynikiem oczekiwanym, co pozwoli nam stwierdzić czy program zadziałał prawidłowo.