**עקרונות שפות תכנות - משימה 2**

1. פרטים חשובים להרצת הפרוייקט:

הקוד מניח שקיים קובץ בשם BikeShare.csv באותה תיקייה בה נמצא קובץ הmybackend.py.

חבילות שהשתמשנו בהן: CSV, Pandas, numpy, sqlite3, kivy, flask.

1. אלגוריתם המלצה:

נתאר את האלגוריתם בשלבים, תוך פירוט על כל שלב.

* 1. נטען את כלל הטיולים לתוך dataframe עם השדות - נק' התחלה, קו רוחב של נק' ההתחלה, קו אורך של נק' ההתחלה, נק' סיום, ומשך הטיול בדקות.

בשלב זה, אם אין טיולים אשר מתחילים בנק' ההתחלה שהמשתמש בחר, לא נוכל להמליץ, ונחזיר רשימה ריקה. הודעה מתאימה תקפוץ למשתמש.

* 1. אנו מחלצים את קו הרוחב והאורך של נק' ההתחלה, ומשאירים טיולים שנק' ההתחלה שלהם מרוחקת עד כקילומטר אחד מנק' ההתחלה המבוקשת. לdataframe נוסף כעת שדה שמתאר את המרחק מנקודת ההתחלה המבוקשת.
  2. בשלב זה אנו מבצעים אגריגציה על הdataframe, כך שניוותר עם רשומות unique-יות של נקודות התחלה וסיום. עבור השדה שמתאר את המרחק מנקודת ההתחלה הרצויה, נבחר שרירותית ערך כלשהו (כי כולם זהים עבור אותה נקודת התחלה), ועבור שדה הזמן נבחר את הערך החציוני, אשר יתאר בצורה טובה ביותר את זמן הטיול הצפוי בין נקודות ההתחלה והסיום.
  3. כעת, עבור כל רשומה נחשב את הscore ע"י שימוש בשני הפרמטרים שחישבנו:
     1. ההפרש בין הזמן של הרשומה לבין הזמן הרצוי על ידי המשתמש.
     2. המרחק בין נקודת ההתחלה שהמשתמש ביקש לבין נקודת ההתחלה ברשומה.

הניקוד לרשומה מוגדר להיות סכום של מינוס המרחק מהנקודה הרצויה, ושל מינוס ההפרש בזמנים. כלומר, הציון המקסימלי, 0, יתקבל עבור רשומה שמתחילה מהנקודה שהמשתמש ביקש, ומשך הזמן שהיא לוקחת שווה בדיוק לזמן שביקש המשתמש.

בנקודת זמן זו של הפיתוח יש משקל זהה להפרש בזמנים ולמרחק, אך בקוד ממומשים מקדמים שניתן לשנות, כדי לתת חשיבות גדולה יותר לאחד מהם או לשני.

* 1. לאחר מכן, נמיין בסדר יורד את הdataframe על פי שדה ה-score, ונחזיר את k הרשומות הראשונות, הן אלו עם הציון הגבוה ביותר.

1. צילומי מסך מתוך האפליקציה:





