דוח מטלה 1 – מבוא לתכנות מונחה עצמים

**שמות מגישים:**

דביר ברזילי 318751971

שמואל שמעוני 204903959

**קישור לפרויקט ב-GitHub:**

<https://github.com/Dvir570/Geographic-Application---OOP-course>

**תיאור המערכת**:

* **המחלקה WiFi**

המחלקה מייצגת רשת WIFI בודדת מתוך כלל הרשתות שיש בקובץ CSV שהאפליקציה מצאה (אנחנו השתמשנו באפליקציה Wiggle WiFi).

במחלקה זו יש שדות שמאחסנות את נתוני רשת ה WIFI שמצאנו:

type, mac, ssid, time, freq, lat, lon, alt, signal, model

מצורפות פעולות GET עבור השדות ובנאי.

* **המחלקה Row**

המחלקה מייצגת שורה בודדת מתוך הטבלה בקובץ CSV הסופי שאותו אנו עתידים לייצר.

במחלקה יש שדה אוסף עם כל האובייקטים של WiFi שאותם נרצה להציג בשורה בקובץ הסופי.

* **המחלקה ResultFile**

המחלקה האחראית על כל ענייני הקובץ CSV הסופי שאותו נרצה לייצר.

פונקציות עיקריות:

insertRows - מקבלת אוסף של אובייקטי WiFi ואחראית להוסיף לקובץ הסופי את הרשתות WIFI לפי הגדרת סעיף 2 במטלה.

rowsGroupByTimeModel - שמקבלת אוסף של אובייקטי WiFi ואחראית לקבץ אותם לקבוצות כך שכל קבוצה תאופיין ע"י הזמן והמודל (ID) של הדגימה.

top10fromAnyGroup – עבור כל קבוצת אובייקטי WiFi הפונקצייה תמיין אותם לפי הסיגנל, ולאחר מכן אם יש יותר מ10 רשתות בקבוצה אז הוא מסיר את כל השאר.

* **המחלקה KML**

מחלקה שתפקידה לייצר קובץ KML.

פונקציות עקריות:

makeKML תפקידה לייצר את הקובץ KML עם כל הנקודות WIFI שהפונקציה מקבלת באוסף.

OpenTag פתוחת תגית.

CloseTag סוגרת תגית.

* **המחלקה DisplayMap**

אחראית על סינון נקודות הWIFI שאותן ירצה המשתמש לראות על המפה לפי הגדרת סעיף 3 במטלה ושימוש באובייקט מהמחלקה KML שיצור קובץ מתאים.

פונקציות עקריות:

displayByTime מסננת את נקודות הWIFI שצריך להציג על המפה לפי זמן שהפונקצייה תקבל. לאחר מכן הפונקצייה תייצר קובץ KML ע"י שימוש במחלקה KML.

displayByPlace מסננת את נקודות הWIFI שצריך להציג על המפה לפי מיקום שהפונקצייה תקבל. המיקום יתקבל ע"י הפרמטרים lon lat alt rdiuse. לאחר מכן הפונקצייה תייצר קובץ KML ע"י שימוש במחלקה KML.

displayByModel מסננת את נקודות הWIFI שצריך להציג על המפה לפי מודל (ID) שהפונקצייה תקבל. לאחר מכן הפונקצייה תייצר קובץ KML ע"י שימוש במחלקה KML.

* **המחלקה matala0**

זו המחלקה הראשית.

במחלקה תהיה פונקציית ה- MAIN שתבנה אובייקטים ותדאג לתפעול כל הפרויקט ע"י שימוש מסודר בשאר המחלקות שפורטו לעיל כפי שנדרש במטלה.

פונקציות עיקריות נוספות:

isDouble תחזיר "אמת" אם המחרוזת שהיא מקבלת ברת-המרה לטיפוס מסוג Double. אחרת תחזיר "שקר".

getCSVs תחזיר אוסף עם כל קבצי הCSV שנמצאו בתוך התקייה שהPATH שלה נשלח כפרמטר לפונקצייה.

**כלי תוכנה שהיו בשימוש:**

1. האפליקציה Wiggle WiFi שבעזרתה ביצענו דגימות WIFI . האפליקציה יצרה לנו מהדגימות קובץ CSV שממנו התחלנו לעבוד.
2. נעשה שימוש בסיפריות של JAVA כגון

java.io.File

java.util.ArrayList

java.util.Scanner

וספריות נוספות

1. GOOGLE EARTH
2. Java API for KML (JAK) - כלי תוכנה שאפשר לנו לייצר קובץ KML בקלות, בלי לכתוב כל מיני מחרוזות, ובכך מנענו מעצמנו באגים וטעויות כתיבה...
3. GitHub/Git

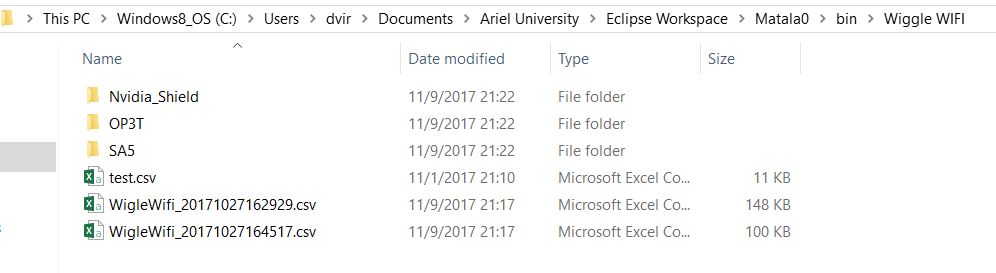
**הסבר על הניסוי שערכנו**

**שלב 1**

ערכנו כמה דגימות ברכבי האוניברסיטה

ע"מ שיהיה יותר תוכן להתעסק בו השתמשנו גם בדגימות מהקבצים לדוגמא שעלו למודל.

העברנו את כל הקבצי CSV לתקייה המיועדת. חלק מהקבצים היו בתוך תקיות נוספות ע"מ לבדוק אם התוכנית תצליח למצוא אותם.



**שלב 2**

הרצת התוכנית שכתבנו.

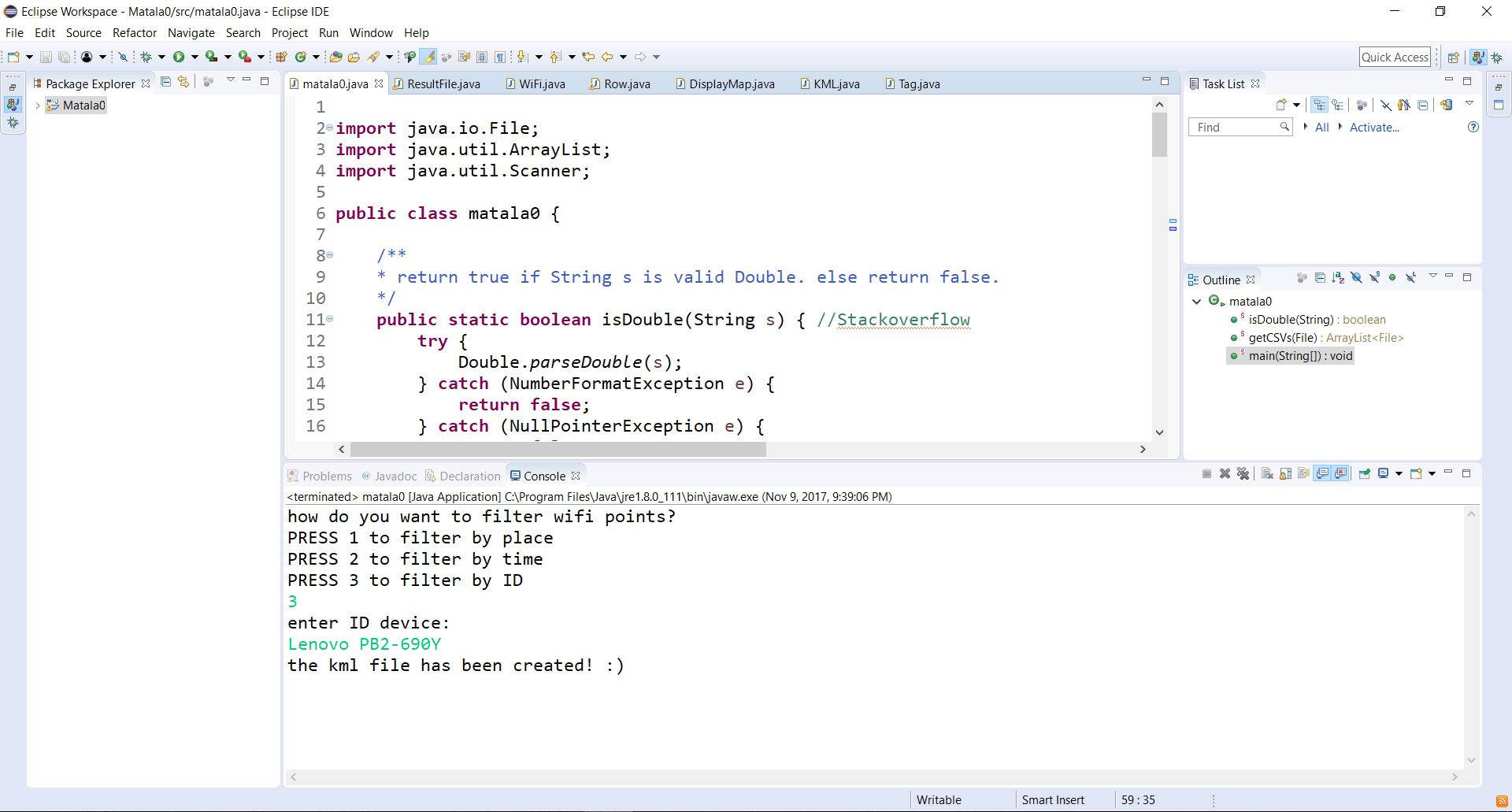
לאחר הבדיקה בדקנו האם הוכן קובץ CSV שעונה על דרישות הסעיף 2 למטלה והתוצאה היתה חיובית!



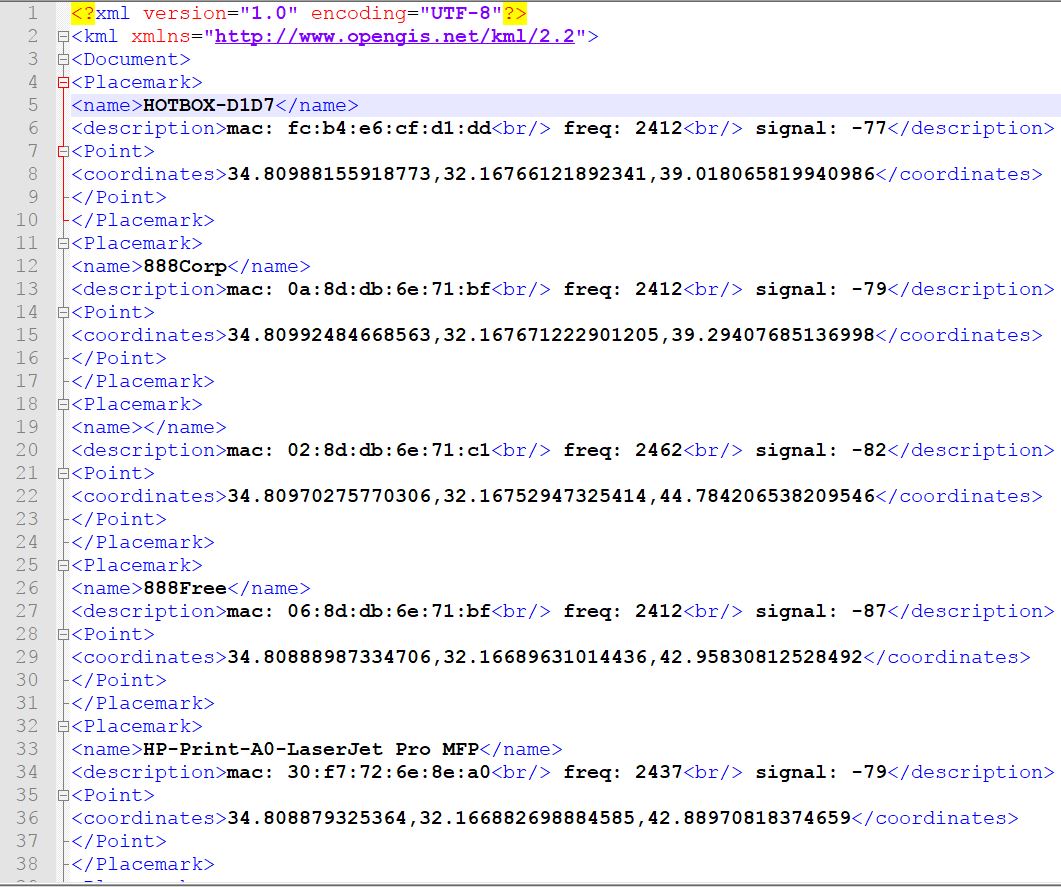
**שלב 3**

בדיקה האם התוכנית יוצרת קובץ KML כמתבקש בסעיף 3.

באחת מההרצות שעשינו בחרנו לעשות סינון לפי ID של המכשיר:



ואז וידאנו שאכן הוכן הקובץ KML:



כמו שרואים בתמונה, הוכן קובץKML כפי שנדרש היה ליצור במטלה.

כל שנותר לנו לבדוק זה שהקובץתקין וניתן לקרוא אותו בGoogle Earth אז העלאנו אותו וזו התוצאה שקיבלנו:



לפיכך, ניתן לראות שהניסוי עבר בהצלחה והתכנית שכתבנו עובדת ☺ .