פירוט המחלקות נמצא בקובץ דוח בדף הראשי של הפרוייקט

# האלגוריתמים

בעיקרון השתמשנו באלגוריתם ובקבועים מהאקסל של בועז (וגם הרצנו בדיקה על מקרה זהה)

פירוט:

## אלגוריתם 1:

### קלט:

מחרוזת של path של קובץ csv מהסוג שאנחנו מייצרים שיהווה מאגר הנתונים

מחרוזת של path של קובץ שיהווה פלט

מספר שלם X שיהווה מספר הסריקות המקסימלי שלפיו נאכן את המיקום של כל ראוטר

### עיבוד:

האלגוריתם לוקח עבור כל ראוטר את X הסריקות שבהם קלטו אותו הכי גבוה, לכל סריקה קובע משקל לפי אחד חלקי עוצמת הקליטה של הראוטר בריבוע ולפי משקלים אלו עושה ממוצע משוקלל לקואורדינטות של הסריקות ומה שיוצא זה המיקום המשוערך של הראוטר

### פלט:

קובץ בפורמט של csv בpath השני שנקלט שהעמודה הימנית היא mac, הבאה אחריו ssid ולאחר מכן הקואורדינטות המשוערכות של הראוטר

## אלגוריתם 2:

### קלט:

מחרוזת של path של קובץ csv מהסוג שאנחנו מייצרים שיהווה מאגר הנתונים

מחרוזת של path של קובץ csv מהסוג שאנחנו מייצרים שצריך למצוא את הקואורדינטות של הסריקות שבו

מחרוזת של path של קובץ שיהווה פלט

מספר שלם X שיהווה מספר הסריקות המקסימלי שלפיו נאכן את המיקום של כל סריקה

### עיבוד

עבור כל סריקה שצריך למצוא את המיקום שממנו היא בוצעה, האלגוריתם קובע לכל סריקה במאגר הנתונים משקל, היא לוקחת את X הסריקות עם המשקלים הכי גבוהים ולפי משקלים אלו עושה ממוצע משוקלל לקואורדינטות של הסריקות ומה שיוצא זה המיקום המשוערך של הסריקה.

מציאת המשקל:

עבור כל ראוטר בסריקה שרוצים את מיקומה בודקים האם הוא נמצא בסריקה שאנו 'ממשקלים' עכשיו: אם הראוטר נמצא שם מגדירים שההפרש הוא הערך המוחלט של חיסור העוצמות אחד מהשני, אם זה יוצא פחות משלוש מגדירים את ההפרש שלוש, אם הראוטר לא נמצא אז מגדירים את ההפרש כמאה

המשקל של כל ראוטר שנמצא בסריקה שאנו מחפשים את מקומה מוגדר עבור הסריקה שאנו ממשקלים כעשרת אלפים חלקי: ההפרש בחזקת 0.4, כפול העוצמה של הראוטר ( בסריקה שאנו מחפשים את מקומה) בריבוע

המשקל של הסריקה נקבע למכפלת המשקלים של הראוטרים

### פלט:

אותו קובץ csv שהיה בpath השני יודפס ב path השלישי, כעת עם הקואורדינטות המשוערכות של כל סריקה

# טסטים

בכל תקייה של טסטים יש את קבצי האפליקציה שבהם השתמשנו

קבצי האפליקציה שהשתמשנו בהם (מתוך הקבצים שבועז נתן)

קובץ בשם wifi.csv שהוא איחוד קבצי האפליקציה (במקום קבצי הcomb)

קובץ router.csv שהוא התוצאות של האפליקציה שלי

קובץ boaz.csv שהוא התוצאות של האפליקציה של בועז (מהרצה שלנו כדי לוודא שעובדים עם אותו קלט)

קובץ אקסל השוואה שמימין נמצא הנתונים של בועז, באמצע הנתונים שלנו (הכל ממויין לפי MAC), משמאל ההפרשים בין הנתונים ומשמאל למטה ממוצע ההפרשים

## אלגוריתם 1

### BM2

רוב ההפרשים הם 0 ובודדים עם הפרשים זניחים לחלוטין, וכשמסתכלים בפנים במקומות שנטען שיש הפרשים זה נראה אותו דבר (אני לא מבין למה האקסל כותב שיש הפרש בכלל), ההפרש הממוצע שהוא טוען הוא **4.92002E-16**

הבדיקה נוסתה עם 3 הדומים ביותר

### BM3

הפעם הבדיקה נוסתה עם 20 נקודות הדומות (מתוך הנחה שזה כולם) התוצאות כמו קודם, הפרש ממוצע 6.66E-15 ועוד פעם כשמסתכלים בפנים נראה שאין בכלל הפרש.

## אלגוריתם 2

יצרנו junit בהסתמך על האקסל והוא רץ טוב.

## Excel

יצרנו קבצים לפי הנתונים שבאקסל שקיבלנו, חישבנו לפי שלושה נקודות, ניתן לראות שהנתונים שיצאו לנו ולבועז הם שונים, לנו יצא כמו האקסל ולבועז יצא אחרת (גם חבר אחר שהבאתי לו את הנתונים אמר שיצא לו בתוכנה שלו כמונו), לכן אנחנו מניחים שהוא יצר עם פונקציה מתמטית שונה ממה שהוגדר או עם קבועים אחרים וגם להבא אנחנו לא מצפים שייצא אותו דבר. (הנתונים בלי הקואורדינטות נקראים man), ההפרש בגובה הוא 0.653894 וההפרש הממוצע בקואורדינטות הוא 0.000895

## bm3TS2

עם נתונים של bm3 וקבצים בלי קואורדינטות של TS2 עם 3 הבדיקות הדומות ההפרש הממוצע בגובה הוא -4.27797 ובקואורדינטות 0.000671

כאן לא רק שכנראה האלגוריתמים קצת שונים (למרות שהפערים מאוד קטנים) אפשר לראות שספירת המקים שלנו ושל בועז שונה, הסיבה היא שבועז כשיש שני ראוטרים זהים באותה בדיקה מתייחס אליהם כשניים, אבל לפי הדרישות של מטלה 0 כשיש ראוטרים זהים לוקחים את החזק ביניהם אז זה מה שהפונקציה שלנו עושה(חוץ מזה שזה מוזר שהאפליקציה באותה סריקה זיהתה את אותו ראוטר פעמיים בשתי עוצמות שונות...)

## Bm2TS1

עם נתונים של bm2 וקבצים בלי קואורדינטות של TS1 עם 5 הבדיקות הדומות ההפרש הממוצע בגובה הוא -1.15672 ובקואורדינטות -0.00043

כאן עושה רושם שהספירה של ראוטרים שלנו זהה, ואכן הפער הממוצע נמוך יותר (אך עדיין כמו שציינו ב' Excel' יש הבדלים)