1)סיקור כללי-

המחלקה העיקרית שמומשה במטלה היא מחלקת הפולינום, שמורכבת ממבנה נתונים "פרטי" בצורת "ArrayList" שמאחסן את "מרכיבי הפולינום" שהם "מונומים" – מחלקה בפני עצמה, כלומר איברים בודדים בעלי מקדם ממשי וחזקות טבעיות אי שליליות – שבמערך מקושר בעצם מייצגים "רב איבר", כלומר פולינום.

מחלקת הפולינום מממשת את הממשק "polynom_able" שמחייב אותה למרבית הפונקציות שיש בה, כל הפונקציות במחלקת המונום נבנו ע"מ לממש את הפונקציות של הממשק הנ"ל בצורה אופטימלית במחלקת הפולינום, בעצם, מחלקת המונום נבנתה בעיקר ע"מ לתמוך במחלקת הפולינום והממשק שמממשת.

קיימת גם מחלקת "test" שמהווה מחלקת בדיקות הנעשות בMAIN, אבל בדיקת ה-"רכיבים" של שתי המחלקות נעשית בפועל בשני test cases של Junit.

חומרי עזר שנעזרתי בהם –

https://www.desmos.com/calculator - כלי רשת שניתן להציג בו גרפים שלפונקציות ולקבל את המאפסים שלהן, נעזרתי בכלי הנ"ל ע"מ לבדוק את פונקציית הroot

https://www.wolframalpha.com/ - אתר המבצע חישובי אינטגרלים מסוימים (בין היתר), נעזרתי בו ע"מ לבדוק את פונקציית הarea

2)מחלקת המונום-

מחלקה זו בנויה ע"מ לתמוך במחלקת הפולינום, ז"א כלל המתודות בה תוכננו ע"מ שנוכל לזמנן ולהשתמש בהן בצורה הנוחה ביותר במימושים הנדרשים במחלקת הפולינום ונקבעים ע"י הממשק "polynom_able".

בנוסף, כיוון שמחלקה זו מממשת את ממשק ה-"פונקציה", היא מחויבת לממש את המתודה "f".

לגבי הצגת המונום על המסך ע"י הפונקציה toString, פונקציה זו תציג מונומים בחזקת 0 בתור מקדם בלבד, ומונומים בעלי מקדם 1 בתור נעלם בחזקה ללא מקדם, וכמובן שבמידה והמקדם הוא אפס, הפונקציה תציג "0" בלבד.

פונקציות "פרטיות" שהוגדרו ע"מ לממש מחלקה זו:

1. initStringMonom-
   פונקציה שתומכת בבנאי המחרוזת של המונום, מקבלת מחרוזת, ופולטת אובייקט מסוג מונום שכולל את ערכי המונום שהתקבל כקלט.

יש לשים לב-

1. הבנאי שמקבל String כקלט **לא יעבוד** עבור ביטויים שאינם מתארים מונום בצורה נכונה לוגית, כלומר, הבנאי ידע להתמודד עם:

* מקדם ללא נעלם

*מקדם ו'x' ללא חזקה

* 'x' עם חזקה וללא מקדם

*נעלם עם מקדם וחזקה

*String ריק, יוגדר בתור מונום ה-0

כאשר המקדם הוא משתנה מסוג double והחזקה היא מספר טבעי אי שלילי מסוג int , במידה והבנאי יקבל מחרוזת שאינה מהצורה הנ"ל הוא יזרוק exception עם המידע הרלוונטי לגבי השגיאה.

2. פונ' הכפל שבמחלקת המונום שונה מבחינת השימוש שלה מאשר שאר הפונ' המתמטיות במחלקה, זו פונקציה סטטית שמקבלת שני מונומים ואת המכפלה שלהם מחזירה כמונום חדש, פונ' זו נוצרה ככה כיוון שהיה נוח יותר לממש איתה את פונ' הכפל במחלקת הפולינום.

3)מחלקת הפולינום-

מחלקת הפולינום היא המחלקה העיקרית שמימשנו, מחלקה זו מממשת את הממשק "polynom_able" ונתמכת ע"י מחלקת המונום.

לגבי הצגת הפולינום ע"י הפונקציה toString, פולינום ריק יוצג כ-"0", כיוון שהפולינום הריק מוגדר להיות פולינום האפס, מעבר לזה – פונקציה זו פשוט תדפיס את כלל המונומים שברשימה מסודרים ע"י החזקות שלהם.

ע"מ לממש מחלקה זו בצורה הטובה ביותר, הוגדרו בה מספר פונ עזר "פרטיות":

1. initStringPolynom-

פונ' סטטית שנעזרים בה בבנאי המחרוזת, היא מקבלת כקלט את מחרוזת הקלט מהבנאי ופולטת ArrayList שמהווה רשימה של כל המונומים שנמצאו במחרוזת (הסבר טכני על הפעולה נמצא בהערות בפונ')

2. subtract-

פונ' פרטית שמחסרת מונום קלט מהפולינום שהופעלה עליו, בדומה לפונ' המקבילה בadd.
פונ' זו נוצרה ע"מ להפוך את מימוש הפונ' subtract שנמצאת בממשק לפשוט יותר.

3. fixup -

פונ שמופעלת לאחר פעולות מתמטיות ובבנאי, עוברת על כלל המונומים, בודקת שאין כאלו עם חזקות שוות, במידה ויש מצמצמת אותם ע"י חיבור, בנוסף, במידה ומגלה "מונמי אפס" ברשימה, מוחקת אותם.

יש לשים לב-

1. הבנאי שמקבל String כקלט **לא יעבוד** עבור ביטויים שאינם מהצורה .....-/+ax^b כאשר כל מונום במחרוזת חייב להיות תקין לוגית כפי שמוגדר בקובץ זה עבור **"מונום"**, במידה והבנאי יקבל מחרוזת שאינה מהצורה הנ"ל הוא יזרוק exception רלוונטי עבור השגיאה.
2. עבור מחרוזת ריקה הבנאי יחזיר את פולינום ה-0.
3. פונקציית הroot פועלת עפ"י עקרון - "שיטת החצייה", שיטה נומרית לחישוב מאפסים של פונ' בטווח מסוים, להסבר נוסף https://en.wikipedia.org/wiki/Bisection_method
4. פונקציית הarea מבוססת על העיקרון של "אינטגרל רימן", שיטה נומרית לקירוב שטח של פונ' בתחום מסויים, הפונ' הנ"ל מחשבת **רק** שטח מעל ציר הX, להסבר נוסף https://en.wikipedia.org/wiki/Riemann_integral

4)מחלקת הטסטר-

מחלקה זו מציגה שימוש בכל הפונקציות של מחלקת הפולינום והמונום, כל הבדיקות מתועדות בהערות במהלך הMAIN, בחלק מהפונקציות נעשתה בדיקה ויזואלית, כלומר הצגה של הפלט שלאחר הפעולה של הפונ', ובחלק מהבדיקות נעשתה בדיקה לוגית – כלומר השוואה של הערך שאמור להתקבל לערך שאכן התקבל.

5)Junit testing case-

קיימות 2 מחלקות Junit הבודקות את כלל הפונקציות הפומביות של מחלקות ה-"מונום" וה-"פולינום", כל "רכיב" פומבי הקשור למחלקות אלה נבדק בtesting cases.