**שלב 4 - חבילת אלמנטים, קמרה, יצירת קרניים**

לפני הכל נקפיא את השלב הקודם באתר **github** ע"י יצירת **branch** חדש בשם **recitation3**. לאחר מכן נוודא שענף ברירת המחדל נשאר **master**.

בשלב הזה של העבודה נתחיל לבנות תשתית עבור סצנה. נתחיל מבניית קמרה ויצירת קרניים ממנה דרך ה-**View Plane**.

1. נוסיף חבילה חדשה בשם **elements**.
2. בחבילה הזו נוסיף מחלקה חדשה **Camera**. נוסיף במחלקה:

* שדות רלוונטיים פרטיים (מיקום מסוג **Po****int3D** ושלושה וקטורים של אורינטציה של הקמרה בהתאם – עבור *vto*, ו-*vup*, ו-*vright*) עם פעולות מאחזרות (גטרים).
* בנאי עם פרמטרים עבור ערכי המיקום ושני וקטורים של כיוון – קדימה ומעלה (*vto* ו-*vup*). על הבנאי בין היתר לוודא שהוקטורים *vto* ו-*vup* מאונכים, וליצור וקטור *vright*.

**נ.ב.1** יש לדאוג שכל הווקטורים השמורים עם הקמרה יהיו מנורמלים

* פונקציה **constructRayThroughPixel** כפי שנלמד בקורס, עם החתימה הבאה:

**public Ray constructRayThroughPixel (int nX, int nY,  
 int j, int i, double screenDistance,  
 double screenWidth, double screenHeight)**

***נ.ב.1 שימו לב שה-View Plane מיוצג בפועל על ידי הארגומנטים שתתנו בטסטים עבור הפרמטרים nX, nY, screenDistance, screenWidth ו-screenHeight. הפרמטרים i ו-j הם אינדקסים של שורה ועמודה של פיקסל מסוים במטריצת ה- שדרך מרכזו מיוצר הקרן מהקמרה.***

***נ.ב.2 אסור ליצור בנאיי ברירת מחדל והעתקה ופעולות מעדכנות (סטרים) עבור מחלקת Camera.***

***נ.ב.3 שימו לב לסדר הפרמטרים של גודל המטריצה ושל אינדקסים של פיקסלים!*** *הסיבה לכך שברישום הקואורדינטות אנחנו מקדימים את ציר ה-****X*** *(קואורדינטה אופקית) לציר ה-****Y*** *(קואורדינטה אנכית), אבל בסריקת מטריצות (של פיקסלים למשל) אנחנו מקדימים את אינדקס השורה (אינדקס אנכי) לאינדקס העמודה (אינדקס אופקי)*

בשלב הראשון הפונקציה תחזיר **null**, למחלקה החדשה ולכל הפעולות שלה (הבנאי, הגטרים והפונקציה הנ"ל) ייכתב תעוד בפורמט **javadoc**.

1. בחבילה **unittests** נוסיף מחלקת בדיקות **CameraTests** (המודול של המחלקה מצורף לתרגיל – יש לקחתו ולהציב בחבילת הבדיקות – אם שם החבילה אצלכם שונה – יש לעדכן את שם החבילה בפקודת **package** במודול). במחלקה הזו מופיעות בדיקות יחידה (**unit tests**) עבור הפונקציה הנ"ל בהתאם למה שנלמד בקורס לגבי **TDD** ומפורט במצגת המעבדה, כמו כן בדיקות הבנאי.
2. יש ליצור מחלקה נוספות בחבילה **unittests** לבדיקות אינטגרציה בין יצירת קרניים מקמרה לבין חישוב חיתוכים של קרן עם גופים גאומטריים מהשלב הקודם. במחלקה תהינה שלוש פונקציות בדיקה (עבור אינטגרציה מול ספירה, מישור ומשולש – הבדיקות כפי שמופיעות במצגת המעבדה). בבדיקות אינטגרציה אלה אין צורך לחלק את הבדיקות לפי **EP** ו-**BVA**.

* כל בדיקה תיצור קרניים עבור כל הפיקסלים ב-**View Plane** 3 על 3 פיקסלים, וגודל 3 על 3, תחשב חיתוך של כל קרן עם הגוף של הבדיקה ותסכם את כמות החיתוכים מכל הקרניים
* כפי שמובן מהנאמר בפסקה הקודמת - אין צורך לחשב את כל נקודות החיתוך במדויק ו-\או לצבור אותם ברשימה כוללת
* חובה להוסיף תעוד למחלקה ולכל פונקציית טסט ולכל בדיקה כמו בכל המודלים של בדיקות
* מיקום הקמרה, נתוני ה-**View Plane**, מיקום וגודל הגוף הגאומטרי של כל בדיקה יילקח מהשקפים של המעבדה, ואם חסרים שם פרטים – יושלמו ע"י הסטודנטים לפי התוצאה הנדרשת בטסט
* הבדיקות חייבות לכסות לפחות את המקרים שמופיעים במצגת המעבדה ובמקרה הצורך על פי שיקול דעת הסטודנטים ניתן להוסיף בדיקות נוספות לפי החשיבה שלכם וע"פ דרישות ה-**TDD**

1. **רק לאחר כתיבת כל הבדיקות** יש לממש בפועל את הפונקציה **constructRayThroughPixel** כפי שנלמד בקורס התאורטי ומוצג בשקפים של המעבדה, ויש לוודא שכל הבדיקות מגיעות להצלחה מלאה - הן במודול של בדיקות יחידה שקיבלתם עם התרגיל והן בבדיקות האינטגרציה.

* במימוש הפונקציה יש לתת את הדעת למקרה של פיקסלים מרכזיים (כאשר במטריצה כמות אי זוגית של שורות ו-\או עמודות של פיקסלים), כפי שמופיע בשקף המתאים במצגת המעבדה
* **נ.ב.** האינדקסים של הפיקסלים מתחילים מ-0 עד לכמות הפיקסלים במימד המתאים פחות אחת

**הערה כללית להמשך**: כדאי ליצור מיקומים של קמרה וגופים, כיוונים ומרחקים כך שיקלו עליכם את יצירת הטסטים ואת החישובים הדרושים לכך.

בהצלחה!