**בס"ד**

**מיני פרויקט – מבוא להנדסת תוכנה**

**שפת תכנות: JAVA**



**הוגש על ידי:**

יוסף כלפה : 309633321

אביעד פייג : 311304448

**דו״ח- מיני פרויקט מבוא להנדסת תוכנה**

בפרוייקט זה נבנה מודל שייצר לנו צורות תלת ממדיות. להלן כמה דברים בסיסיים שצריך להבין על מנת להצליח להבין את הפרוייקט.

וקטור: בטבע ישנם גדלים מסוימים שניתן לתאר אותם באופן כמותי בלבד (כמו טמפרטורה, מסה, שטח ונפח). גדלים אלו נקראים **סקלרים** ואלו גדלים חסרי כיוון.

אך בטבע קיימים גם גדלים נוספים, אשר גם כיוון הפעולה שלהם חשוב לנו (כמו מהירות, תאוצה וכוח). גדלים אלו נקראים **וקטורים**.

את הווקטור ניתן לתאר על ידי זוג נקודות מסודרות: מוצא הווקטור וסוף הווקטור.



אם מוצא הווקטור בנקודה A וסופו בנקודה B אז מקובל להציג

את הווקטור בהצגה נוחה יותר בעזרת חץ שהחוד שלו מצביע על

סוף הווקטור.

שיוויון בין וקטורים: נאמר ששני וקטורים שווים אם הם שווים באורכם

וגם הכיוון שלהם. יכולים להיות שני וקטורים שלא מתחילים ונגמרים

באותה נקודה אבל הם יהיו שווים כי הם שווים באורכם ושווים בכיוון.

מכפלה סקלרית: מכפלה סקלרית היא פעולה על שני וקטורים מהמרחב האוקלידי R^n שמחזירה סקלר. לפי ההגדרה הגאומטרית מכפלתם הסקלרית של שני וקטורים שווה למכפלת אורכיהם.



לפי ההגדרה האלגברית :



מכפלה וקטורית: מכפלה וקטורית היא פעולה בינארית על שני וקטורים במרחב תלת ממדי, שמחזירה וקטור (בניגוד למכפלה הסקלרית שמחזירה סקלר). הווקטור המוחזר תמיד ניצב לשני הווקטורים המוכפלים.



תחילה נגדיר את האיברים הבסיסיים של הפרויקט שהם קואורדינאטות, נקודות, וקטורים, וקרניים.

Package primitives:

**Coordinate**

**Point2D**

**Point3D**

**Ray**

**Vector**

**Primitive**

**מחלקת- Coordinate:**

**שדות:** משתנה מסוג double שמציין נקודה (חד מימד)

**מתודות:**

בנאים:

1. בנאי ריק- מאתחל את משתנה המחלקה ל 0.
2. בנאי המקבל מספר (double) ומאתחל את משתנה המחלקה.
3. בנאי העתקה- מקבל אובייקט מסוג Coordinate ומעתיק את נתוניו למשתנה המחלקה

Add- מקבלת אובייקט מסוג Coordinate ,מוסיף את ערכו לאיבר המחלקה.

Substract- מקבלת איבר מסוג Coordinate ומורידה את ערכו מאיבר המחלקה.

CompareTo- משווה בין 2 איברים מסוג Coordinate מחזירה 0 אם שווים ו -1 אחרת.

Get/Set- פונקציה המחזירה/מעדכנת את איבר המחלקה.

**מחלקת- Point2D:**

**שדות:** 2 משתנים מסוג Coordinate שמציינים X ו- Y.

**מתודות:**

בנאים:

1. בנאי ריק- מאתחל את משתני המחלקה ע"י בנאי ברירת המחדל של Coordinate.
2. בנאי המקבל 2 איברים מסוג Coordinate ומאתחל את X ו- Y בהתאמה.
3. בנאי העתקה- מקבל איבר מסוג Point2D ומעתיק את נתוניו למשתנה המחלקה.

Get/Set- מחזירות/מאתחלות את איברי המחלקה.

CompareTo: משווה בין 2 איברים מסוג Point2D מחזיר 0 אם שווים, ו- (-1) אחרת.

Equals- משווה בין שני איברים ומחזירה true אם הם שווים אחרת מחזירה false.

**מחלקת- Point3D:**

**מאפיינים:** המחלקה יורשת מ- Point2D ולכן בנוסף למה שיש בה, במחלקה זו יש משתנה נוסף שמציין את קואורדינטת ה- Z במערכת תלת-מימד.

**מתודות:**

בנאים-

1. בנאי ריק- מאתחל את משתני המחלקה ל 0.
2. בנאי המקבל 3 איברים מסוג Coordinate ומאתחל את משתני המחלקה בהתאם.
3. בנאי העתקה- מקבל איבר מסוג Point3D ומעתיק את נתוניו למשתנה המחלקה.
4. בנאי המקבל 3 איברים מסוג double ומאתחל את משתני המחלקה בהתאם.

Get/Set- מחזירות/מאתחלות את איברי המחלקה.

CompareTo- משווה בין 2 איברים מסוג Point3D מחזיר 0 אם שווים, ו-(-1) אחרת.

Add- מקבלת איבר מסוג ווקטור ומוסיפה אותו לנקודת המחלקה, כך שנקודת המחלקה עוברת לראש הווקטור אילו היה מתחיל מנקודת המחלקה.

Substract- מקבלת איבר מסוג ווקטור ומוסיפה אותו לנקודת המחלקה, כך שנקודת המחלקה עוברת תחילת הווקטור אילו היה ראש הווקטור נגמר בנקודת המחלקה.

Distance- מקבלת איבר מסוג Point3D ומחזירה double המציין את המרחק בין 2 הנקודות.

Gets/Sets- מחזירות/מאתחלות את איברי המחלקה.

CompareTo- משווה בין 2 איברים מסוג Point3D מחזיר 0 אם שווים, ן- -1 אחרת.

toString- מחזירה ייצוג של המחלקה ע"י מחרוזת בתבנית- (X, Y, Z) עם 2 נקודות עשרונית מקסימום לכל קואורדינטה.

**מחלקת- Vector:**

**מאפיינים:** המחלקה מכילה איבר מסוג Point3D שמציין את ראש הווקטור במערכת תלת-מימד.

**מתודות:**

בנאים-

1. בנאי ריק- מאתחל את משתני המחלקה ע"י בנאי ברירת המחדל של Point3D לווקטור (0,0,0).
2. בנאי המקבל איבר מסוג Point3D ומאתחל את איבר המחלקה.
3. בנאי העתקה- מקבל איבר מסוג Vector ומעתיק את נתוניו למשתנה המחלקה.
4. בנאי המקבל 3 איברים מסוג double ומאתחל את משתנה המחלקה בהתאם.
5. בנאי המקבל 2 איברים מסוג Point3D ויוצר מהם ווקטור, ע"י כך שהיא מחסרת את המשתנה הראשון מהשני ובכך יוצרת ווקטור בכיוון של הנקודה הראשונה לשנייה.

Get/Set-מחזירות/מאתחלות את ערך משתנה המחלקה שהוא ראש הווקטור.

crossProduct (מכפלה וקטורית)- מקבלת איבר מסוג Vector ומבצעת מכפלה וקטורית של איבר המחלקה באיבר שהתקבל.

dotProduct (מכפלה סקלרית)- מקבלת איבר מסוג Vector ומבצעת מכפלה סקלרית של איבר המחלקה באיבר שהתקבל.

CompareTo- משווה בין 2 איברים מסוג Point3D מחזיר 0 אם שווים, ו- (-1) אחרת.

GetHead/SetHead- מאתחלת/מחזירה את איבר המחלקה המציין את ראש הווקטור.

Length- מחזירה איבר מסוג double שמציין את גודל ווקטור המחלקה. ע"י שימוש במשפט פיתגורס.

Normalize- פונקציה המנרמלת את ווקטור המחלקה כך שגודלו יהיה 1. ע"י הכפלה של כל איבר בווקטור הראש ב אחד חלקי גודל הווקטור.

Scale- מקבלת איבר מסוג double שהוא מספר שבו נרחיב את ווקטור המחלקה, ומכפילה כל איבר של ווקטור הראש באיבר שהתקבל.

Add- מקבלת איבר מסוג Vector ומוסיפה אותו לווקטור המחלקה, ע"י הוספה לכל קואורדינטה של ווקטור הראש את הקואורדינטה המתאימה בווקטור שהתקבל.

Substract- מקבלת איבר מסוג Vector ומפחיתה אותו מווקטור המחלקה, ע"י הורדה לכל קואורדינטה של ווקטור הראש את הקואורדינטה המתאימה בווקטור שהתקבל.

toString- מחזירה ייצוג של המחלקה ע"י מחרוזת בתבנית- (X, Y, Z) עם 2 נקודות עשרונית מקסימום לכל קואורדינטה.

**מחלקת- Ray:**

**מאפיינים:** המחלקה מכילה 2 איברים. אחד מסוג Point3D שמציין את מקום תחילת הקרן. ואחד מסוג Vector שמציין את הכיוון והגודל של הקרן.

**מתודות:**

בנאים-

1. בנאי ריק- מאתחל את משתני המחלקה ע"י בנאי ברירת המחדל של Point3D ושל Vector.
2. בנאי המקבל איבר מסוג Point3Dואיבר מסוג Vector ומאתחל את איברי המחלקה בהתאם.
3. בנאי העתקה- מקבל איבר מסוג Vector ומעתיק את נתוניו למשתנה המחלקה.

Gets/Sets- מחזירות/מאתחלות את איברי המחלקה

1. **חבילת geometry-**

בחבילה זו נמצאים כל הצורות הגאומטריות שנצייר בעזרת התוכנה

מבנה החבילה:

**Geometry**

**Flatgeometry**

**Radialgeometry**

**Plane**

**חבילת geometries**

**Triangle**

**Sphere**

**Cylinder**

**מחלקות החבילה**

**מחלקת Geometry-**

**מאפיינים:** זוהי מחלקה המהווה ממשק (interface) לכל הצורות שניצור בפרויקט.

הממשק מחייב בירושתו לממשק 2 פונקציות עיקריות:

1. getNormal להחזרת הנורמל בנקודה.
2. getIntersections להחזרת נקודות החיתוך של קרן (Ray ממחלקת primitive) עם הצורה.

וכן למחלקה יש 3 משתנים:

1. משתנה מסוג Matirial שמציין את סוג החומר.
2. משתנה מסוג double שמציין את הבהירות של החומר.
3. משתנה מסוג Color שמציין את הצבע שהצורה פולטת.

**מתודות:**

1. getIntersections- פונ׳ המקבלת "קרן" והמחזירה רשימה של כל הנקודות חיתוך שלה עם הצורה של המחלקה. זוהי פונ׳ אבסטרקטית וממומשת בכל יורש.
2. getNormal- פונ' המקבלת נקודה ומחזירה את האנך של הצורה באותה נקודה. גם כן פונ' אבסטרקטית וממומשת בכל יורש.

Gets/Sets- פונקציות שמאתחלות/מחזירות את איברי המחלקה או מאתחלות את האיבר מסוג material שבתוך המחלקה.

**מחלקת FlatGeometry-**

מחלקה זו היא interface ואינה מכילה שום מאפיינים מטרתה היחידה היא לאחד את כל הצורות השטוחות תחת אותה קטגוריה כך שאם נרצה להוסיף מאפיין כללי לצורות שטוחות נוכל לעשות זאת בקולות.

**מחלקת RadialGeometry-**

זוהי מחלקה אבסטרקטית שתפקידה להיות מחלקת האב לכל הצורות העגולות (בעלות רדיוס)

**מאפיינים-** מחלקה מכילה משנה אחד מסוג double שמייצג את אורך הרדיוס שיש לצורה.

**מתודות:**

Gets/Sets- מחזירות/מאתחלות את משנה המחלקה (שהוא גודל הרדיוס)

**מחלקת Sphere-**

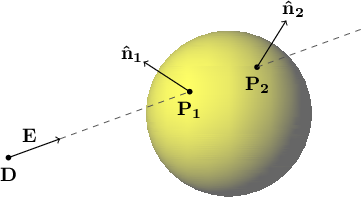
**מאפיינים:** מחלקה המצייגת כדור במרחב שמיוצג ע"י רדיוס ונקודה. המחלקה יורשת מRadialGeometry ובנוסף למשתנה הרדיוס ממחלקת האב הוא מוסיף משתנה מסוג Point3D שיציין את מיקום מרכז הכדור במרחב.

**מתודות:**

בנאים:

1. בנאי ריק- מאתחל את משתני המחלקה כך שהרדיוס בגודל 1, ומרכז הכדור בנקודה (0,0,0)
2. בנאי העתקה- מקבל משתנה מסוג Sphere ומעתיק ממנו את הרדיוס ונקודת המרכז ומשתני המחלקה.
3. בנאי המקבל 2 משתנים: 1. מסוג double שמציין את גודל הרדיוס 2. מסוג Point3D המציין את מרכז המעגל.

FindIntersections- פונ' המקבלת קרן ומחזירה רשימה של כל נקודות החיתוך של הקרן עם הכדור. (מקסימום 2 נקודות חיתוך).



מציאת הנקודות ע"פ השלבים:

1. ליצור ווקטור ממקור הקרן אל מרכז המעגל
2. לעשות מכפלה סקלרית בין כיוון הקרן לווקטור ב-(א)
3. מוצאים את המרחק בין מרכז המעגל להטלה שלו על הווקטור מ-(א) ע"י פיתגורס: שורש של (ב) בריבוע ועוד שורש של האורך של (א) בריבוע.
4. אם (ג) גדול מאורך הרדיוס אז אין חיתוך בתוך המעגל ומחזירים רשימה ריקה במידה ויש עוברים לסעיף הבא.
5. מוצאים את המרחק בין הטלת מרכז המעגל לנקודת החיתוך ע"י פיתגורס: שורש של רדיוס בריבוע ועוד (ג) בריבוע.
6. יוצרים ווקטור חדש שהוא הכיוון של הקרן ופעם אחת נוסיף לו (ג) פחות (ה) זאת נקודה אחת. ופעם שנייה נוסיף לו (ג) ועוד (ה).
7. מסעיף (ו) קיבלנו את 2 נקודות החיתוך נוסיף אותם לרשימה ונחזיר אותה.

getNormal- פונ' המקבלת איבר מסוג Point3D ומחזירה את האנך של הכדור בנקודה.

Gets/Sets- מחזירות/מאתחלות את משתנה המחלקה מסוג Point3D המציין את מרכז הכדור.

**מחלקת Plane-**

**מאפיינים:** מחלקה זו מייצגת מישור במרחב, המיוצג ע"י נקודה ונורמל למישור.

המחלקה מכילה 2 איברים אחד מסוג Point3D שמייצג נקודה על המישור. ואחד מסוג Vector שמייצג את הנורמל למישור

**מתודות:**

בנאים:

1. בנאי ריק- שמאתחל את הנורמל ל (0,0,1) ואת הנקודה לראשית הצירים (0,0,0).
2. בנאי העתקה- מקבל איבר מסוג Plane ומעתיקה את נתוניו לאיברי המחלקה.
3. בנאי המקבל 2 איברים אחד מסוג Pint3D שמייצג את הנקודה על המישור, ואחד מסוג Vector שמייצג את הנורמל למישור, ומאתחל את הנתונים בהתאם.

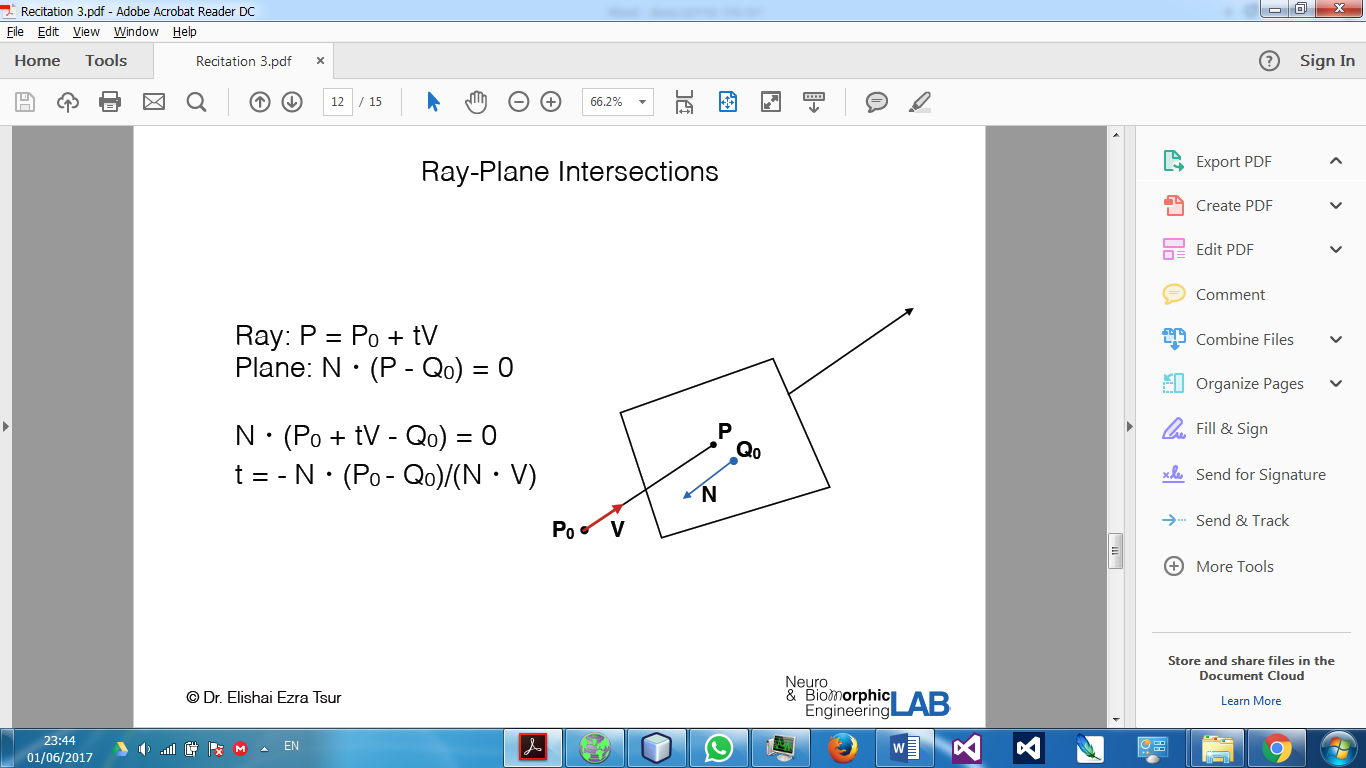
FindIntersections- פונ' המקבלת איבר מסוג Ray המייצג קרן ומחזירה רשימה של כל נקודות החיתוך של הקרן עם המישור. (בפועל יכולה לחזור רק נקודה אחת).

תהליך מציאת החיתוך:

בשביל למצוא את נקודת החיתוך צירך למצוא קבוע (t) שנכפיל בכיוון הקרן כך שכשנוסיף אותו לנקודת מקור הקרן נגיע לנקודת החיתוך.

בהנחה שיש חיתוך עם הקרן אז מכפלה סקלרית של הווקטור מהנקודה שיש על המישור(שהיא נקודת מקור הקרן ועוד t כלשהוא כפול כיוון הקרן) לנקודת החיתוך כפול הנורמל של המישור צריכה להיות 0.

אז אם נסדר את המשוואה נקבל שה- t שאנחנו צריכים הוא:



כאשר N זהו נורמל המישורת, P0 זו נקודת מקור הקרן, Q0 הנקודה הנתונה על המישור, V זהו ווקטור הכיוון של הקרן.

getNormal- פונ' המקבלת איבר מסוג Point3D ומחזירה את הנורמל של המישור בנקודה. פועל לכולם אותו הנורמל שהוא בעצמו איבר של המחלקה.

Gets/Sets- פונ' המחזירות/מאתחלות את איברי המחלקה.

**מחלקת Triangle-**

**מאפיינים:** מחלקה זו מייצגת משולש שטוח בתוך המרחב.

המחלקה מכילה 3 איברים מסוג Point3D שמייצגים את קודקודי המשולש.

**מתודות:**

בנאים:

1. בנאי ריק- זהו בנאי ברירת מחדל שיוצר משולש עם הקודקודיים: (1,0,0) (0,1,0) (0,0,1).
2. בנאי העתקה- מקבל איבר מסוג Triangle ומעתיק את נתוניו לאיברי המחלקה.
3. בנאי המקבל 3 איברים מסוג Point3D שמציינים את קודקודי המשולש.

FindIntersections- פונ' המקבלת קרן ומחזירה רשימה של כל נקודות החיתוך של המשולש עם הקרן.

תהליך מציאת נקודת החיתוך:

קודם מוצאים את נקודת החיתוך עם המישור שעליו נמצא המשולש, ונשתמש בפונ שהוגדרה במחלקה Plane למציאת חיתוך.

את איבר המישור שניצור נגדיר כך: הנורמל הוא מכפלה וקוטרית של הווקטורים p1->p2 והווקטור p1->p3. וניקח נקודה כלשהיא מנקודות המשולש p1.

עכשיו נשאר לבדוק אם נקודת החיתוך היא בתוך התחום של המשולש.

בשביל זה ניצור 3 ווקטורים כשאר P0 זוהי נקודת המקור של הקרן:

1. p0->p1 מכפלה וקוטרית עם p0->p2
2. p0->p2 מכפלה וקוטרית עם p0->p3
3. p0->p3 מכפלה וקוטרית עם p0->p1

ננרמל את שלושת הווקטורים ונעשה לכל אחד בנפרד מכפלה סקלרית עם הווקטור מ- נקודת החיתוך שנמצאה ל-p0.

אם הסימן של שלושת המכפלות זהה, ז"א כולם פלוס או כולם מינוס. אז נקודת החיתוך היא בתוך התחום.

נוסיף אותה לרשימה של נקודות חיתוך ונחזיר את הרשימה.

getNormal- פונ' המקבלת איבר מסוג נקודה ומחזירה את האנך של המשולש בנקודה, שבמקרה של משולש זה אותו אנך לכל הנקודות.

Gets/Sets- מחזירות/מאתחלות את משתני המחלקה, שהם קודקודי המשולש.

1. **חבילת elements-**

בחבילה זו נמצאים כל מיני אלמנטים מיוחדים שיהיה להם שימוש בפרויקט שלנו למשל מצלמה וסוגי אורות.

מבנה החבילה:

**Light**

**LightSource**

**DirectionalLight**

**AmbientLight**

**חבילת elements**

**PointLight**

**SpotLight**

**Camera**

**מחלקות החבילה**

1. **מחלקת Camera-**

**מאפיינים:** מחלקה זו מייצגת מצלמה בתוך תלת מימד,

המצלמה מיוצגת ע"י:

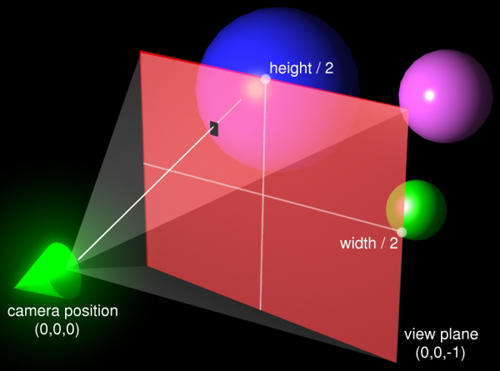
1. מיקום במרחב שמיוצג ע"י משתנה מסוג Point3D
2. כיוון שהמצלמה מסתכלת אליו שמיוצג ע"י 3 ווקטורי כיוון מסוג Vector, vTo vUp ו- vRight. בפועל vRight הוא מכפלה ווקטורית של 2 הווקטורים האחרים.

**מתודות:**

בנאים:

1. בנאי ריק- מאתחל את המיקום של המצלמה לנקודה (0,0,0), ואת ווקטורי הכיוון ל- vTo(0,0,-1), vUp(0,1,0), vRight(1,0,0). בפעול vRight מחושב בתור מכפלה ווקטורית.
2. בנאי העתקה- מקבל איבר מסוג Camera ומעתיק את נתוניו לאיבר משנה המחלקה בהתאם.
3. בנאי המקבל 3 איברים: point3D שמציין את מיקום המצלמה, 2איברים מסוג Vector שמציינים את כיוון המצלמה. ומאתחל את משתני המחלקה בהתאם.

constructRayThroughPixel- זוהי פונ' שמטרתה להחזיר קרן שעוברת מהמצלמה דרך פיקסל בתוך המסך.



הפונ' מקבלת 7 פרמטרים ונסמן אותם כדי שנשתמש בהם הפונ׳ בהמשך:

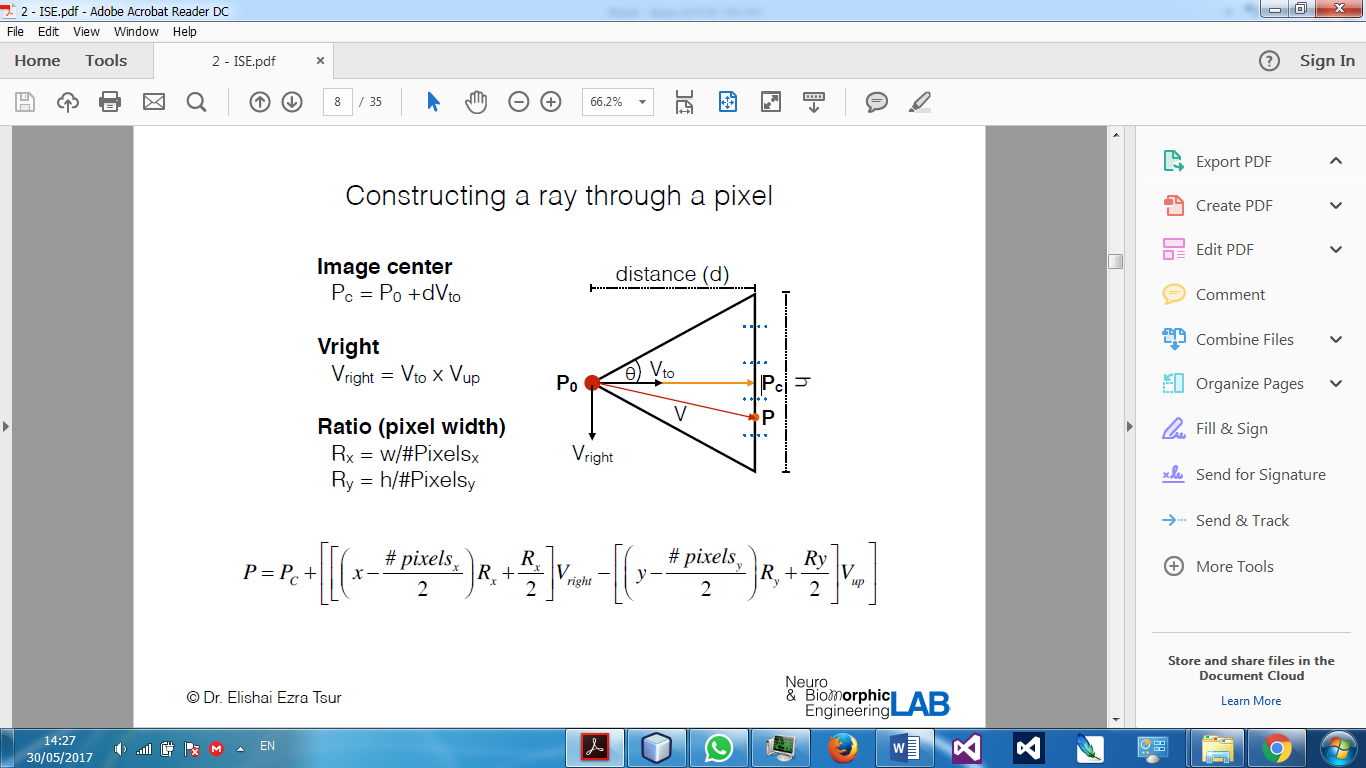
2 משתנים מסוג int שמציינים את היחס של המסך (Nx,Ny) , ו5 משתנים ש 2 מהם אלו האינדקסים של הפיקסל במסך(x,y) , 2 מציינים את הגובה והרוחב של המסך(width,height), ואחד מציין את המרחק בין המצלמה למסך (screendist).

בשביל למצוא את הקרן נצטרך למצוא את הנקודה במסך ע"י הפונקציה הזו:

Pc-זוהי נקודת האמצע במסך מקבלים אותה ע"י הוספה למקור המצלמה את הווקטור של vTo כפול screendist.

#pixels זה ה Nx ו- Ny.

Rx/Ry- זהו היחס בין הרוחב/גובה לNx/Ny.



קיבלנו ע"י הפונ׳ את הנקודה על המסך כל שנותר הוא ליצור ווקטור ממקור המצלמה לנקודה וכך קיבלנו את הכיוון של הקרן.

בסוף מחזירים את הקרן עם הנקודה שמצאנו והכיוון שמצאנו.

Gets/Sets- מחזירות/מאתחלות את משתני המחלקה. במקרה שמשנים את vTo או vUp אז הפונקציה תשנה גם את vRight בהתאם.

1. **מחלקת Light-**

**מאפיינים:** זוהי מחלקה אבסטרקטית שמשמשת מחלקת אב לכל האורות.

מוגדרת ע"י משתנה אחד מסוג Color.

**מתודות:**

בנאים:

1. בנאי ריק- מאתחל את הצבע ללבן.
2. בנאי העתקה- מקבל איבר מסוג Light ומעתיק את המאפיין color שלו למשתנה המחלקה.

getIntensity- זוהי פונ' אבסטרקטית לכל מי שיורש ממחלקה זו.

1. **מחלקת AmbientLight-**

**מאפיינים:** מחלקה זו מייצגת תאורה סביבתית.

המחלקה מורכבת ממשתנה מסוג double שהוא מספר בין 1 ל- 0 שמציין את הכמות של האור.

**מתודות:**

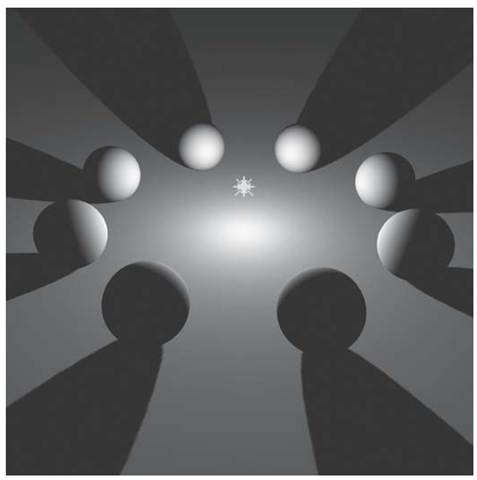
בנאים:

1. בנאי ריק- משתמש בבנאי האב של Light
2. בנאי העתקה- מעתיק את משתנה הצבע למשתנה המחלקה.
3. בנאי המקבל 3 איברים מסוג int שמציינים את ערכי ה rgb של הצבע שרוצים לאתחל בו את איבר המחלקה.

Gets/Sets- מחזירות/מאתחלות את איברי המחלקה. במקרה של getIntensity מוחזר צבע שערכי ה rgb שלו מוכפלים במשתנה המחלקה.

1. **מחלקת PointLight-**

**מאפיינים:** מחלקה זו מייצגת סוג תאורה כמו מנורה כל מה שקרוב אליה מואר יותר חזק ואין לה כיוון.



**מתודות:**

בנאי המקבל 5 משתנים :

צבע, נקודת מיקום האור, ו 3 מקדמי תאורה.

getIntensity- פונ המקבלת נקודה מחזירה את הצבע שנוצר ממפגש האור עם הנקודה

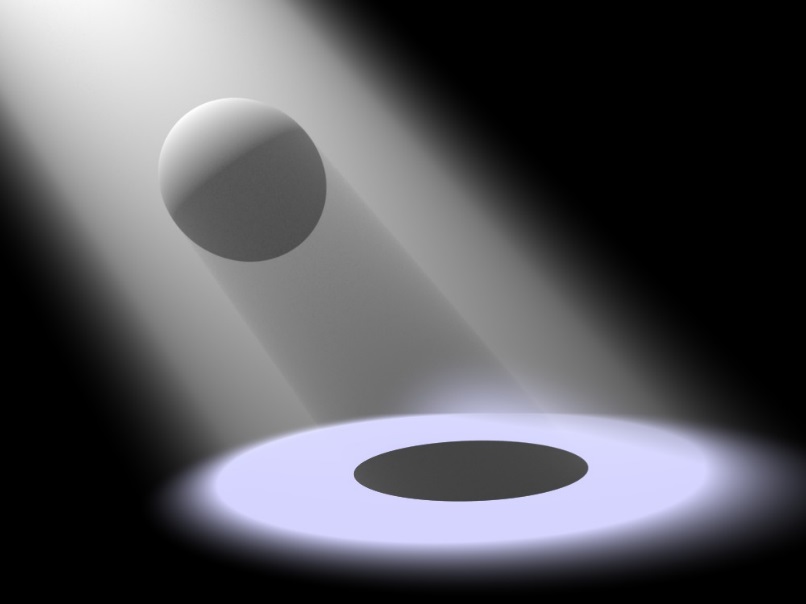
תהליך:

1. מצא את המרחק בין האור לנקודה
2. הכפל את 1 במקדם האור kl וחבר ל כפל של 1 פעמיים בkq והוסף את המקדם האחרון kc
3. חלק את ערכי הRGB של צבע האור בתוצאה שיצאה ב 2 והחזר את הצבע.

getL- פונ המחזירה ווקטור מנורמל בכיוון ממקור האור לנקודת החיתוך.

**5. מחלקת SpotLight-**

**מאפיינים:** מחלקה זו מייצגת תאורה כמו פרוזקטור, יש משמעות לקרבה אל מקור האור וגם לכיוון שבו נמצאים מול מקור האור, ככל שיותר קרובים ומול מקור האור, האור יותר חזק.



**מתודות:**

בנאי- בנאי המקבל 6 משתנים: מיקום, צבע, כיוון ו3 מקדמי תאורה

getIntensity- פונ המחזירה את האור המתאים לנקודה שהתקבלה כפרמטר

תהליך-

1. הכן ווקטור ממקור האור לנקודת וחשב את אורכו.
2. הכן מחלק שיהיה kc ועוד kl כפול המרחק ועוד kq כפול המרחק בריבוע.
3. הכפל את ערכי ה RGB של צבע האור במכפלה הסקלרית של הווקטור ב1 לכיוון האור
4. וחלק את ערכי ה RGB מ3 במחלק שיצרנו ב2
5. החזר את הצבע.

**6. מחלקת DirectionalLight-**

**מאפיינים:** המחלקה מייצגת מקור תאורה שיש לו כיוון ונמצא באינסוף עם עוצמה אינסופית, לכן אין משמעות למרחק ממנה אלא רק לכיוון.

**מתודות:**

בנאי- מקבל 2 משתנים אחד מסוג Color לקביעת צבע האור, ואחד מסוג Vector לקביעת כיוון האור

getIntensity- פונ שמחזירה את הצבע שנוצר ממהפגש עם הנקודה שניתנת לבין האור. פשוט מחזיר את צבע האור.

getL- מחזיר ווקטור ממקור האור את הנקודה שהתקבלה כפרמטר.

1. **חבילת Scene-**

חבילה זו מייצגת את הסצנה שעליה נעבוד שמכילה את כל האלמנטים שמרכיבים את הסצנה כולל הגאומטריות האורות

**Scene**

**חבילת Scene**

**SceneBuilder**

**מתודות:**

1. **מחלקת Scene-**

**מאפיינים:** במחלקה מייצגת את הסצנה שעליה נעבוד ומורכבת מ:

1. צבע רקע 2. תאורה סביבתית 3. רשימה של גאומטריות 4. מצלמה 5. המרחק מהמסך 6. רשימה של תאורות 7. שם הסצנה

**מתודות:**

1. בנאי ריק שמאתחל את איברי המחלקה לערכי ברירת מחדל
2. בנאי העתקה שמקבל איבר מסוג Scene ומעתיק את נתוניו לאיברי המחלקה בהתאם.
3. addGeometry- הפונ מקבלת גאומטריה ומוסיפה אותה לרשימת הגאומטריות של המחלקה.
4. addLight- הפונ מקבלת תאורה כלשהיא ומוספיה אותה לרשימת התאורות של המחלקה
5. Gets/Sets- פונ שמחזירות/מאתחלות את איברי המחלקה.
6. **חבילת renderer-**

בחבילה זו נמצא הכלי שבעזרתו נעבד את כל תוכן הסצנה ונדפיס אותו לקובץ תמונה (jpg)

**חבילת renderer**

**Render**

**ImageWriter**

**מחלקות החבילה:**

1. **מחלקת Render-**

**מאפיינים:** מחלקה זו היא הכלי שבעזרתו נחבר את כל הגאומטריות שיצרנו ואת כל האפקטים בסצנה לתמונה

המחלקה מכילה 3 איברים: אחד מסוג Scene שמייצג את הסצנה, אחד מסוג ImageWriter שזו מחלקה שבעזרתה נדפיס את התמונה, ואחד מסוג Int שמציין כמה קריאות רקורסיביות יתבצעו.

**מתודות:**

1. בנאי המקבל איבר מסוג Scene ואיבר מסוג ImageWriter ומאתחל את איברי המחלקה.
2. renderImage- הפונ משתמשת בנתוני המחלקה כדי לייצר את התמונה הרצויה

תהליך: יוצרים לכל פיקסל קרן/קרניים (לפי רמת הsuper sumpling) ובודקים נקודת חיתוך עם כל הצורות בסצנה, אח"כ מוצאים את הנקודה הקרובה ביותר למצלמה וצובעים את הפיקסל בצבע המתאים לפי נקודת החיתוך ע"י הפונ המיועדת לכך.

1. getSceneRayIntersections- מוצא את נקודות החיתוך של הקרן הנתונה עם הצורות שבסצנה ומחזיר איבר מסוג Map שבו נמצאים נקודות החיתוך והגאומטריות שלהם.

תהליך: עוברים על רשימת הגאומטריות ומוצאים נקודת חיתוך ע"י פונ החיתוך שיש בכל גאומטריה, ע"י פולימורפיזם, ומחזירים רשימה של הנקודה חיתוך והצורות שלהם.

1. getClosestPoint- מוצא את הנקודה הקרובה ביותר למצלמה מבין כל נקודות החיתוך שהתקבלנו אליו בפרמטר מסוג Map ומחזיר גם כן רשימה של Map של נקודות החיתוך עם הגאומטריות שלהם.

מציאת הנקודה הקרובה נעשית ע"י הפונ של מציאת מרחק בין 2 נקודות שהוגדרה במחלקה Point3D.

1. calcColor- מקבל 4 איברים גאומטריה נקודה קרן ומספר שלם ומחשב את הצבע הראוי לפיקסל הנוכחי.

תהליך:

1. קודם נסכום את הצבע של האור הסביבתי והצבע של הגאומטריה.
2. נחשב את האור המפוזר (diffusive) והאור הספקולרי (Specular) ע"י הפונ המיועדות לכל אחד(נפרט בהמשך) אם הקרניים נחסמות ע"י צורה אחרת אז לא נחשב חלק זה ויהיה צל.
3. נחשב את ההשתקפות והשקיפות של הנקודה ע"י חישוב סיכום הצבעים מנקודות ההשתקפות/שקיפות.
4. נסכום את הכל ונחזיר את הצבע.
5. calcDiffusiveComp- פונ שבאמצעותה נחשב את האור המפוזר שנקודה.

תהליך:

1. חשב מכפלה סקלרית בין הנורמל של הצורה בנקודה לבין הווקטור ממקור האור לנקודה.
2. הכפל אותו במקדם הדיפיוזן של החומר של הצורה (KD)
3. הכפל את ערכי הRGB של הצבע של האור בתוצאה של 2 והחזר את הצבע.
4. calcSpecularComp- פונ המחשבת את האור הספקולרי בנקודה.

תהליך:

1. חשב מכפלה סקלרית בין הנורמל של הצורה בנקודה לבין הווקטור ממקור והכפל ב-2
2. הכפל את הווקטור ממקור האור ב-(שורה 1)
3. הכפל את מקדם הספקולר של הצורה ב (מכפלה הסקלרית בין הווקטור ממקור האור לווקטור מהצורה למצלמה בחזקת הshininess של הצורה)
4. הכפל את ערכי הRGB של הצבע של האור ב( שורה 3) והחזר את הצבע
5. constructReflectedRay- מחשב את הקרן המשתקפת מהחיתוך עם הצורה

תהליך:

1. הכפל את נורמל הצורה בכיוון של הקרן הנכנסת ואז הכפל ב 2.
2. הכפל את נורמל הצורה ב תוצאה של שורה 1
3. הפחת הכיוון הקרן את הווקטור של שורה 2.
4. החזר קרן עם נקודת החיתוך עם קידום של אפסילון, ווקטור כיוון משורה 3.
5. constructRefractedRay- מחשב את קרן השקיפות, שחודרת דרך הצורה, הגדרנו שאין מקדם חיכוך אז מחזירים קרן חדשה עם אותו כיוון של הקרן המקורית רק מקור הקרן משתנה לנקודת החיתוך.
6. findClosesntIntersection- מקבל קרן ומחזיר את נקודת החיתוך הקרובה ביותר לקרן.

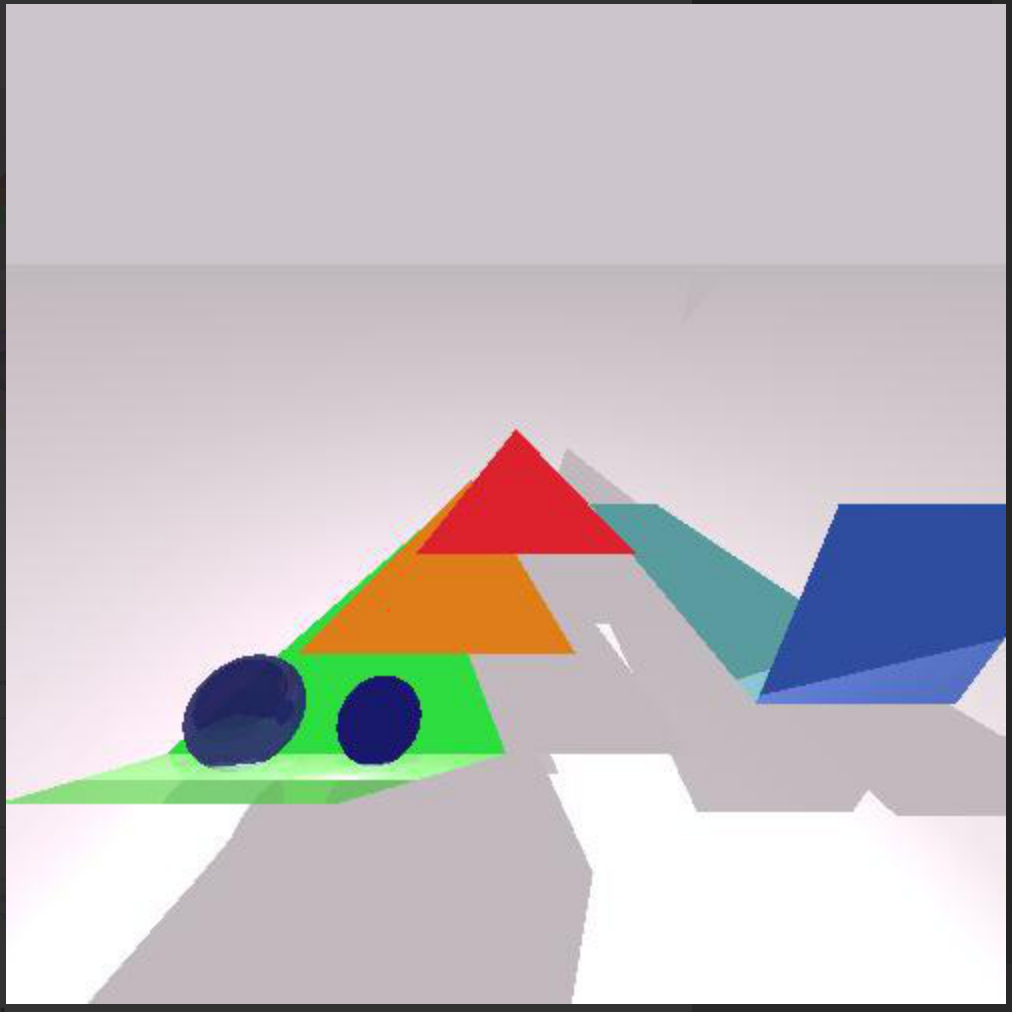
תהליך:

1. עוברים על כל הגאומטריות ומשתמשים בפונ שלהם למציאת חיתוך.
2. בודקים כל פעם אם יש צורה שהמרחק שלה יותר קצר ע"י פונ מובנת ב Point3D.
3. מחזירים את הנקודה ואת הצורה של אותה נקודה.
4. getClosestPoint- מקבל רשימה של נקודות חיתוך ומוצא את הנקודה הקרובה ביותר למצלמה. ע"י הפונ למציאת מרחק ב Point3D.
5. getSceneRayIntersections- מחזיר רשימה של כל נקודות החיתוך של קרן עם כל הצורות בסצנה. פשוט עוברים על כל הצורות משתמשים בפונ' של מציאת החיתוך שלהם ומוסיפים את נקודות החיתוך לרשימה ומחזירים אותה.
6. Occluded- פונ בוליאנית הבודקת אם צורה מסויימת חסומה ע"י צורה אחרת או היא עצמה ולכן לא מגיע אליה אור ממקור האור הספציפי.

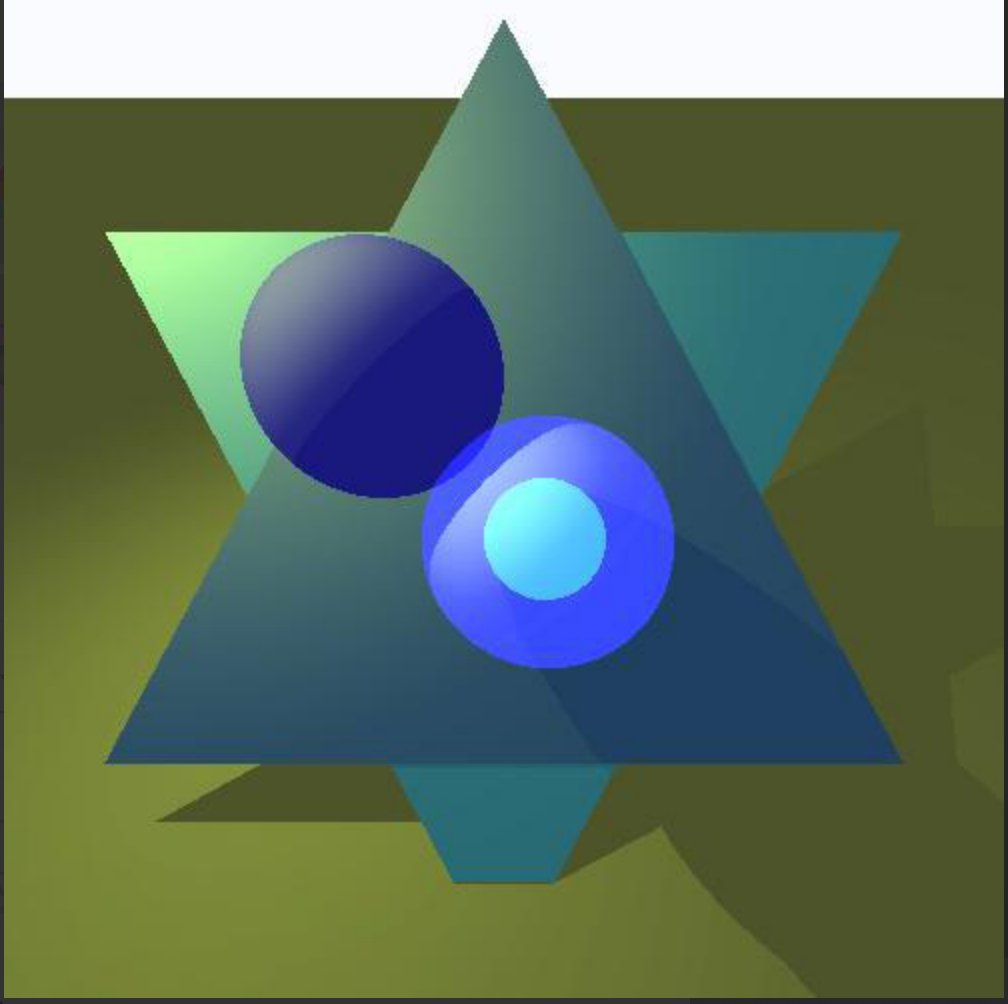
תהליך:

1. צור קרן מהאור אל הצורה והפוך את כיוונו.
2. בדוק נקודות חיתוך של הקרן עם כל הצורות בסצנה
3. אם הרשימה ריקה או שכל הצורות שיש איתם חיתוך שקופות החזר שקר (האור כן מגיע)
4. אם הרשימה לא ריקה ואחת הצורות לא שקופה כלל החזר אמת (לא מגיע אור)

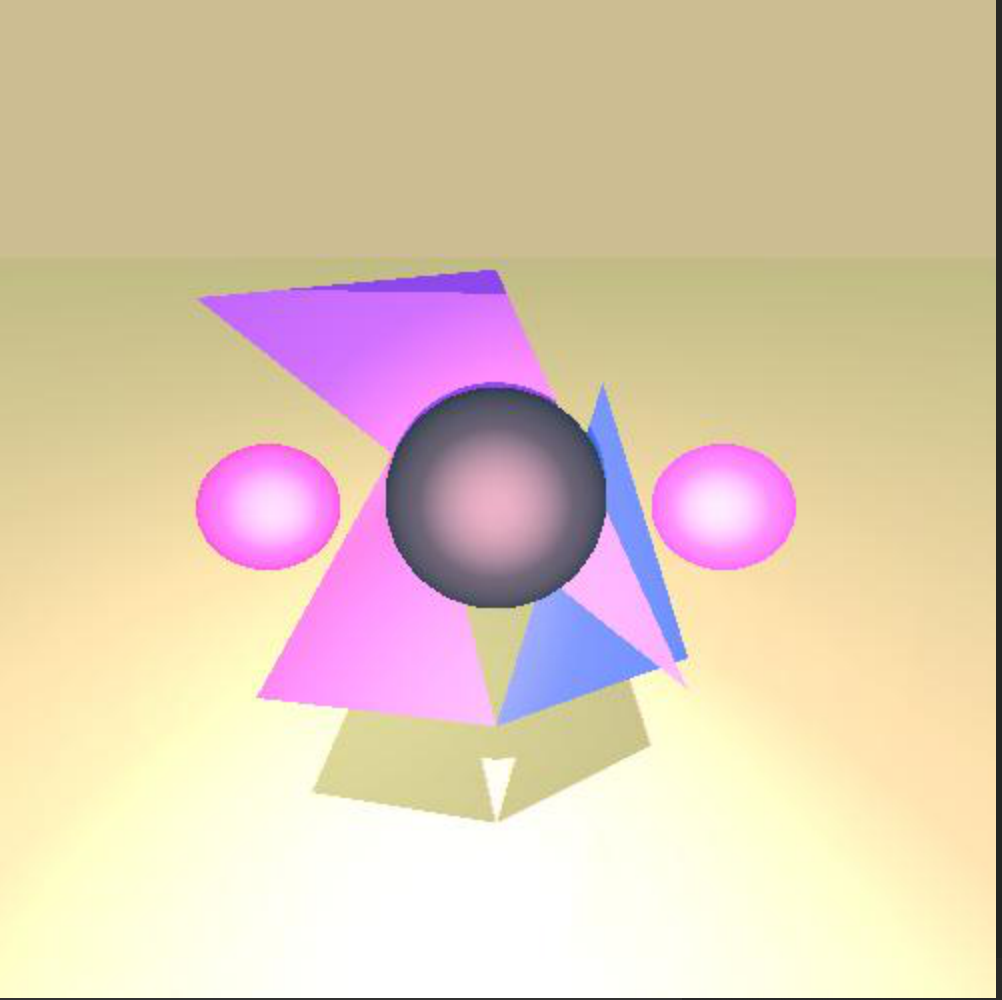
טסטים להגנה 3:



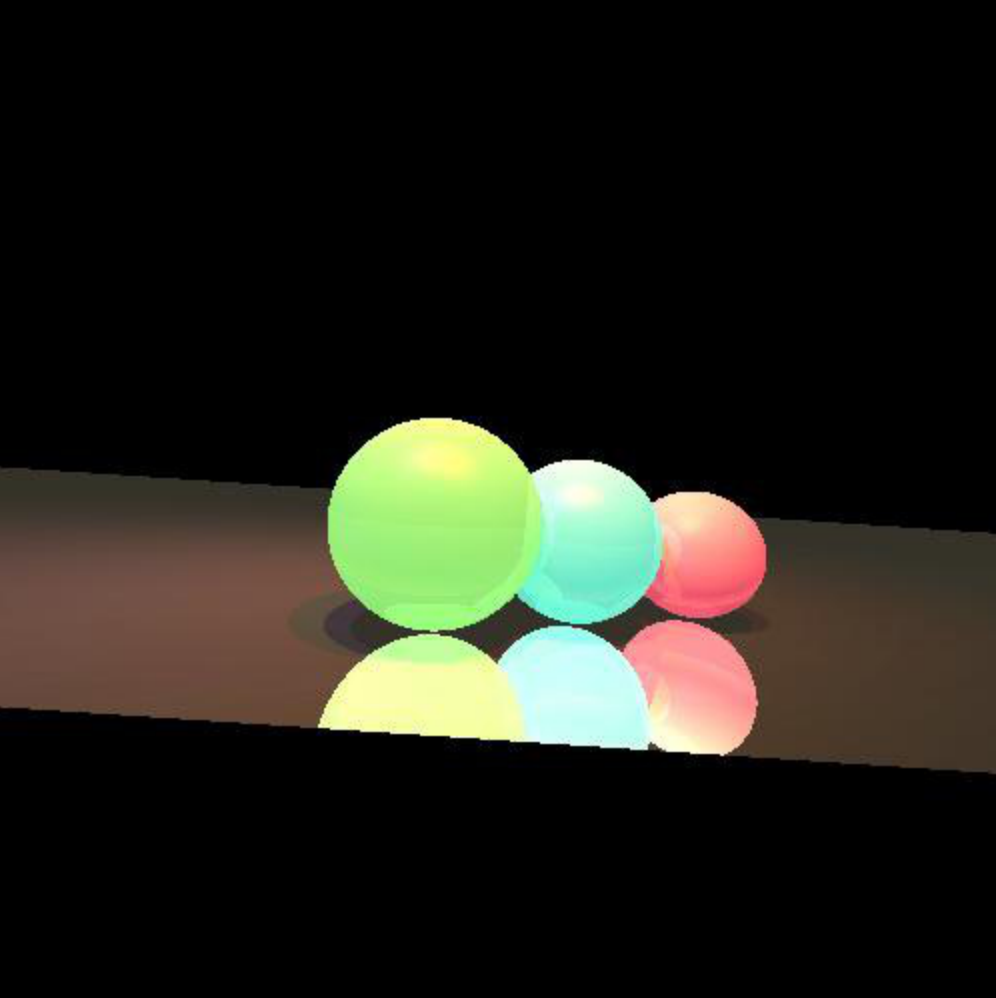
**Test of direction light**



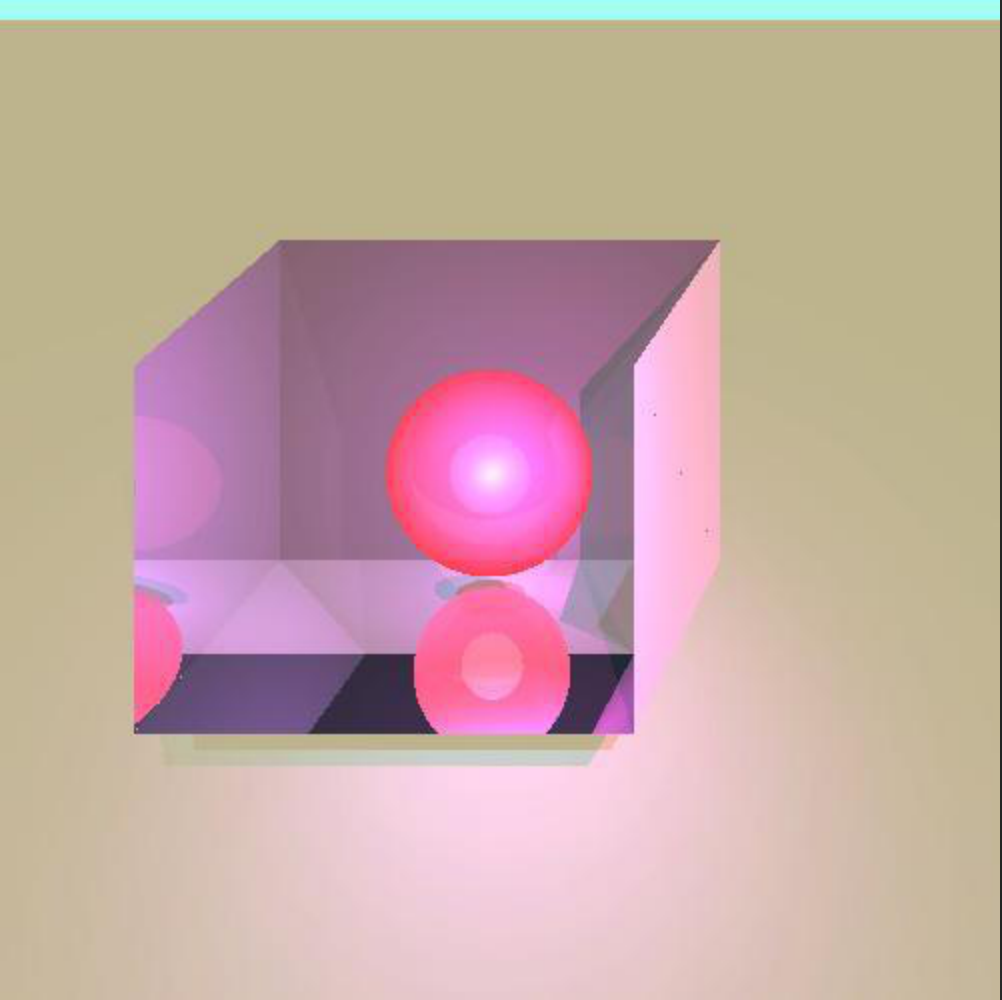
**Test of spot light**



**Test of point light**



**Test 1 of recursion**



**Test 2 of recursion**

**שלב 4:**

**בעיה:**

הצל לא אמיתי בתמונה מכיוון שהוא אינו מבצע ריכוך.

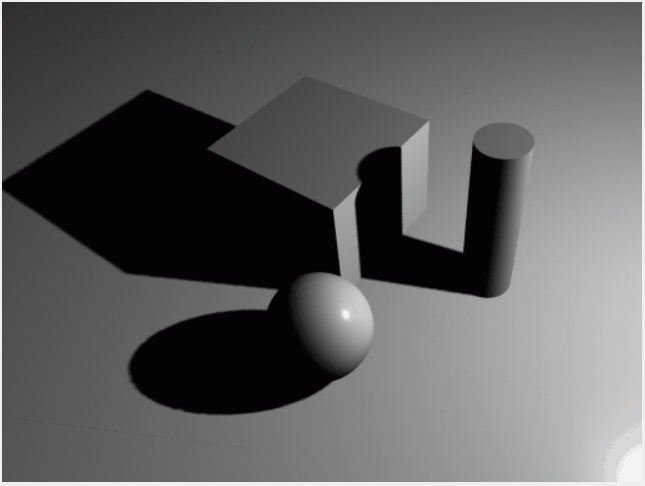
**פתרון:**

שליחת מערך של קרניים לבדיקת החיתוך ובמקום החזרה של ערך בוליאני מחזירים מספר שבו נבצע בדיקה כמה חיתוך (השפעת אור) יש על הגיאומטריה.

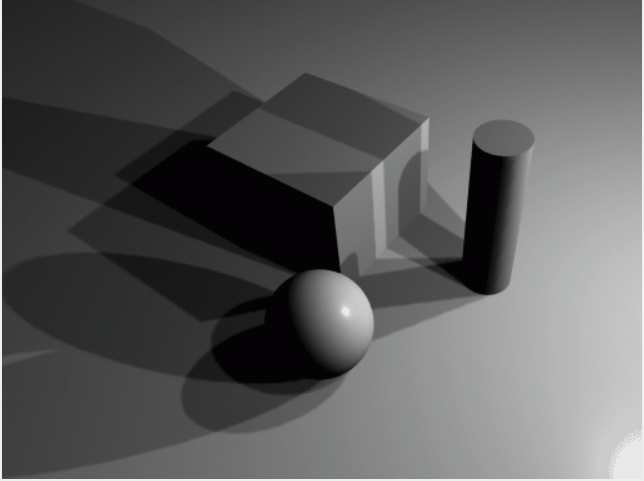
השפעת הצל הסופית תהיה באחוזים לפי כמות הקרניים שאכן יגיעו לתאורה וכמות הקרניים שלא יפגעו בתאורה מכיוון שמשהו מסתיר להם אותה.

**נראה כמה שלבים כיצד מגיעים לצל הרך:**

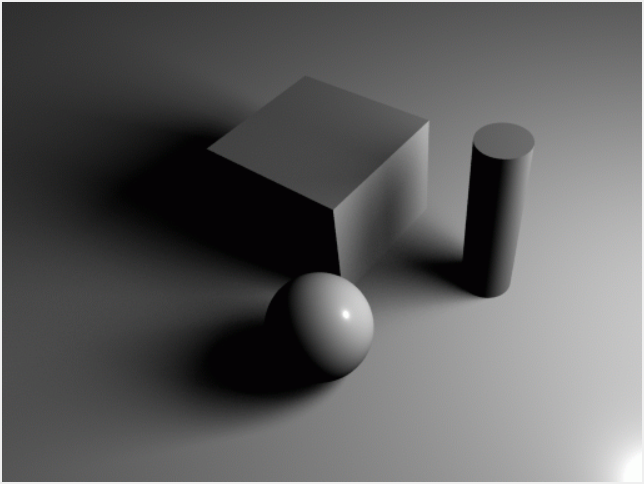
1. לפנינו Hard shadow



1. אם נירה כמה קרניים הרי שנקבל הצללות בכמה מוקדים



**שלב 3:** לאחר שנסכום את כל השתנויות של הצל ונחזיר ערך אחד שמחבר את הכל יחד נקבל בעצם במקום פיזור של כמה הצללות דעיכת הצללה בצורה שתראה לנו אמיתית.



העשרה:

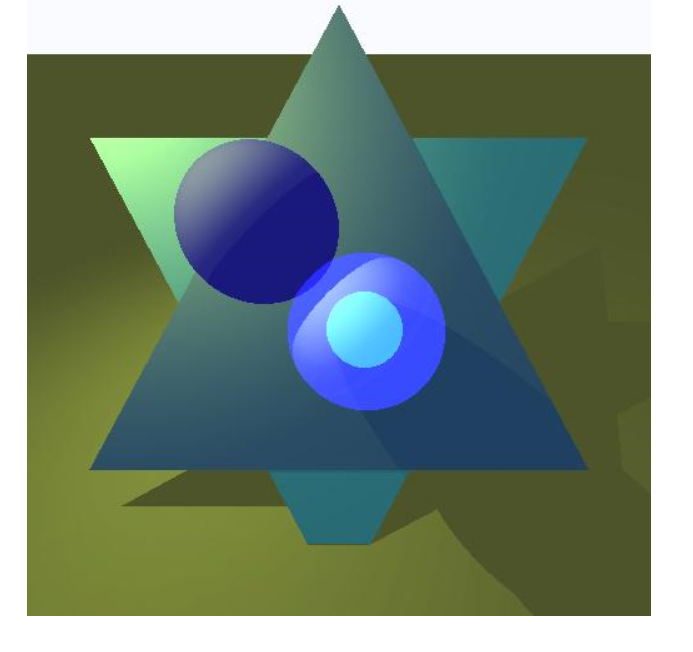
כיצד יתכן שחלק מאזורי התמונה יכולים לקבל צל קשה ואילו חלק מאזורי התמונה מקבלים צל רך(במציאות)

הסיבה לעניין זה נעוצה במרחקים מהמשטח שעליו נזרק הצל וכן זוויות התאורה ועוד.

להסבר מעמיק מצורף קישור לכתבה בנושא: <https://blog.demofox.org/2017/07/01/why-are-some-shadows-soft-and-other-shadows-hard/>

**הגנה שלב 4:**

**סצינה עם Hard Shadow :**



**סצינה עם Soft Shadow:**

