מבני נתונים 1 (234218) – רטוב 1# חלק יבש

מגישים:

- **שם:** דניאל מודריק **ת.ז:** 212243257
- **שם:** ניקיטה ליסוקון **ת.ז:** 332684190

תאריך הגשה:

25/5/2021

תכנון והגדרת המערכת:

מבנה הנתונים שלנו, CarDealershipManager, יכיל את המידע הבא:

CarDealershipManager		
AvlTree<CarType, int> car_types - עץ AVL שכל צומת בו מכיל: • משתנה Key מטיפוס int שמכיל את מספר המזהה של סוג הרכב. • משתנה Data מטיפוס CarType שמכיל את הנתונים של סוג הרכב (יפורט בהמשך) עץ זה יכיל את כל טיפוסי הרכב השונים של המערכת ובעזרתו נוכל לגשת לכל טיפוס ב(O(log(n)).		
AvlTree<int, GradeKey> model_grades - עץ AVL שכל צומת בו מכיל: • משתנה Key מטיפוס GradeKey שמכיל את מספר סוג הרכב, מספר הדגם וציונו במערכת. • משתנה Data מטיפוס int שמכיל את מספר הדגם של הרכב. עץ זה יכיל את כל הציונים של הדגמים שנמכרו לפחות פעם אחת. בעזרתו נוכל להדפיס את K הדגמים עם הציונים הקטנים ביותר ב(O(K.		
AvlTree<List<GradeKey>*, int> unsold_models - עץ AVL שכל צומת בו מכיל: • משתנה Key מטיפוס int שמכיל את מספר המזהה של סוג הרכב • משתנה Data מטיפוס *<List<GradeKey> שמכיל מצביע לרשימה של מפתחות ציונים של דגמים שעוד לא נמכרו, ששייכים לסוג הרכב שמספרו נמצא בKey (יפורט בהמשך) עץ זה יכיל מצביעים לרשימות של דגמים שלא נמכרו מעולם ובעזרתו נוכל לעבור על כל הדגמים שלא נמכרו ביעילות.		
TreeNode<List<GradeKey>*, int>* left_most_unsold_list - מצביע לצומת בעץ unsold_models אשר מכיל את הצומת השמאלי ביותר (כלומר מצביע לרשימת הדגמים שלא נמכרו של הטיפוס הקטן ביותר בעץ) ובעזרתו נוכל לעבור על K דגמים שלא נמכרו ב(O(K.		
TreeNode<int, GradeKey>* worst_model - מצביע לצומת בעץ model_grades אשר מכיל את הצומת השמאלי ביותר (כלומר הדגם בעל הציון הנמוך ביותר בעץ) ובעזרתו נוכל להדפיס את K הדגמים עם הציונים הקטנים ביותר ב(O(K.		
AvlTree<int, SaleKey> car_types_best_sellers - עץ AVL שכל צומת בו מכיל: • משתנה Key מטיפוס SaleKey שמכיל את מספר סוג הרכב, מספר הדגם וכמות המכירות שלו במערכת (ובעצם זה הדגם הנמכר ביותר של סוג הרכב הנתון) • משתנה Data מטיפוס int שמכיל את מספר הדגם של הרכב הנמכר ביותר מסוג הרכב ששמור בSaleKey.		
best_seller_type - משתנה מטיפוס int שיכיל את מספר סוג הרכב של הדגם הנמכר ביותר במערכת	**best_seller_count -** משתנה מטיפוס int שיכיל את כמות המכירות של הדגם הנמכר ביותר במערכת	**best_seller -** משתנה מטיפוס int שיכיל את מספר הדגם הנמכר ביותר במערכת
int total_model_amount – מספר סך כל הדגמים שנמצאים במערכת		

המחלקה CarType אשר מתארת סוג רכב תכיל את המידע הבא:

<table>
<tr><th colspan="3">CarType</th></tr>
<tr><td>unsold_models_amount –
משתנה מטיפוס int אשר יכיל את מספר הדגמים שמעולם לא נמכרו</td><td>models_amount –
משתנה מטיפוס int אשר יכיל את מספר הדגמים של הרכב</td><td>Id -
משתנה מטיפוס int אשר יכיל את מספר סוג הרכב</td></tr>
<tr><td colspan="3">SmartPtr<List<GradeKey> > unsold_list –
מצביע "חכם" (מאפשר למספר מצביעים להצביע לאותו מקום בזיכרון ולדאוג שהזיכרון ישוחרר פעם אחת בלבד), והוא מכיל:
• מספר המצביעים אל הכתובת ששמורה במצביע.
• כתובת של רשימה מקושרת דו-כיוונית שכל איבר בה מכיל משתנה data מטיפוס GradeKey אשר מכיל את מספר סוג הרכב, מספר הדגם וכמות מכירות של 0 (כלומר הרשימה מחזיקה את כל הדגמים שלא נמכרו אפילו פעם אחת).
נשמור מצביע לרשימה זו בעץ unsold_models במחלקה CarDealershipManager.</td></tr>
<tr><td colspan="3">ListNode<GradeKey>** unsold_models_ptrs –
מערך באורך models_amount של מצביעים לאיברים ברשימה המקושרת unsold_list.
אם הדגם i לא נמכר מעולם, אז התא ה-i במערך יכיל:
• מצביע לאיבר ברשימה unsold_list אשר מכיל מפתח ציון (GradeKey) ששייך לדגם i.
• אחרת אם הדגם i נמכר לפחות פעם אחת, ערך המצביע יהיה NULL.
בעת מכירת דגם, נוכל להוציא אותו מהרשימה בסיבוכיות זמן O(1) בעזרת המצביע לאיבר.</td></tr>
<tr><td colspan="3">int *models_sell_count –
מערך באורך models_amount של איברים מטיפוס int, כאשר התא ה-i במערך מכיל את כמות המכירות של הדגם ה-i. בעזרתו נוכל לעדכן את הדגם הנמכר ביותר של סוג הרכב ביעילות.</td></tr>
<tr><td colspan="3">Int *model_grades –
מערך באורך models_amount של איברים מטיפוס int, כאשר התא ה-i במערך מכיל את הציון של הדגם ה-i. באתחול המערכת, ציון הדגם יהיה 0.</td></tr>
<tr><td colspan="2">best_model_count –
משתנה מטיפוס int אשר מכיל את כמות המכירות של הדגם הנמכר ביותר של סוג הרכב</td><td>best_model –
משתנה מטיפוס int אשר מכיל את מספר הדגם הנמכר ביותר של סוג הרכב</td></tr>
</table>

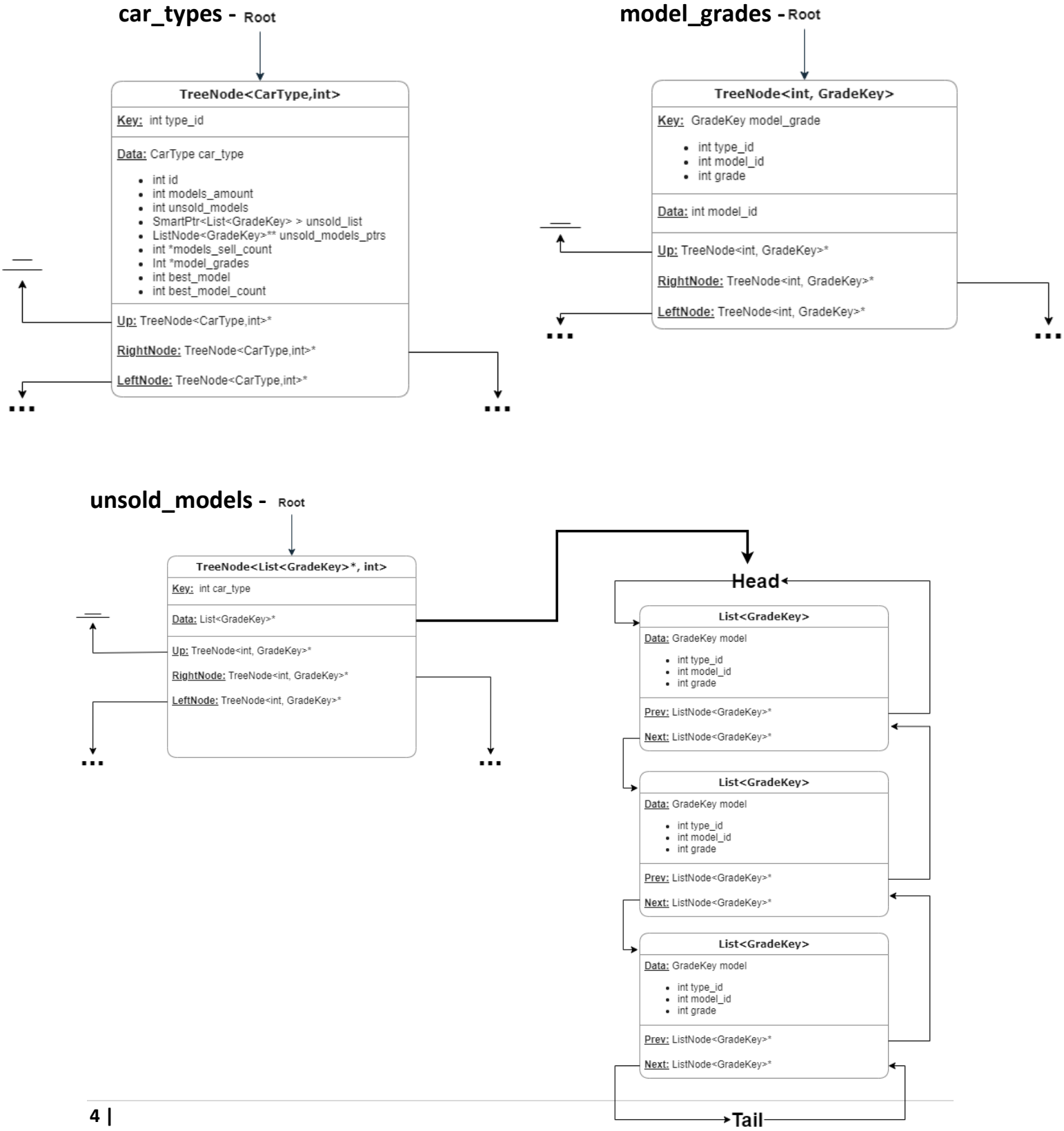
מחלקות מפתחות העצים GradeKey ו- SaleKey יכילו את המידע הבא:

GradeKey	
int type –	מזהה טיפוס הרכב
int model –	מספר דגם הרכב
int grade –	ציון הדגם במערכת

SaleKey	
int type –	מזהה טיפוס הרכב
int sale_count –	מספר מכירות הדגם

תרשים המערכת:

CarDealershipManager:



מימוש הפוקנציות:

1. **void* Init():**

נשתמש בבנאי ברירת המחדל של המחלקה CarDealershipManager ונאתחל את העצים להיות עצי AVL ריקים, את המצביעים לאיברים בעצים נאתחל בNULL, ואת המשתנים ל0.

סיבוכיות זמן: O(1) **סיבוכיות מקום:** O(1)

2. **StatusType AddCarType(void* DS, int typeID, int numOfModels):**

ראשית ניצור אובייקט מסוג CarType אשר יכיל מצביע חכם לרשימה מקושרת unsold_list שאיבריה מייצגים את הדגמים שעוד לא נמכרו, ונאתחל 3 מערכים באורך numOfModels אשר יחזיקו מידע לגבי הדגמים השונים שנמצאים ברשימה **((O(m) זמן, O(m) מקום)**.

כעת נוסיף אובייקט זה לעץ טיפוסי הרכבים car_types, ונשמור מצביע ל unsold_list עם המפתח typeID בתוך העץ unsold_models **(((O(log(n)) זמן, ((O(log(n)) מקום).**

לבסוף נעדכן את המצביע לצומת השמאלי ביותר בunsold_models **(((O(log(n)) זמן)**,
ונוסיף את הדגם 0 עם המפתח מטיפוס SalesKey המכיל את typeID וכמות מכירות השווה ל- 0 לעץ car_type_best_sellers **(((O(log(n)) זמן, ((O(log(n)) מקום)**.

סיבוכיות זמן: O(log(n) + m) **סיבוכיות מקום:** O(log(n)+m)

3. **StatusType RemoveCarType(void* DS, int typeID):**

ראשית נבדוק האם הטיפוס הנתון קיים בעץ הטיפוסים car_types **(((O(log(n)) זמן)**, אם כן אז נסיר את המצביע לרשימת הunsold_list של הטיפוס מהעץ unsold_models **(((O(log(n)) זמן, ((O(log(n)) מקום)**, ונעדכן את המצביע לצומת השמאלי ביותר left_most_unsold_list **(((O(log(n)) זמן).**

כעת נסיר כל דגם שנמכר לפחות פעם אחת מהעץ model_grades **(((O(m*log(M)) זמן, O(log(M) מקום)**, נסיר את הדגם הנמכר ביותר של הטיפוס מהעץ car_types_best_sellers **(((O(log(n)) זמן, ((O(log(n)) מקום)**, ונעדכן את המשתנים שמחזיקים את הדגם הנמכר ביותר **(((O(log(n)) זמן)** ואת המצביע לדגם עם הציון הנמוך ביותר, שהוא הצומת השמאלי ביותר של model_grades **(((O(log(M)) זמן)**.
לבסוף נסיר את הטיפוס מעץ הטיפוסים car_types **(((O(log(n)) זמן, O(log(n) מקום)**.

סיבוכיות זמן: O(log(n) + m*log(M)) **סיבוכיות מקום:** O(log(n)+log(M)

4. `StatusType SellCar(void* DS, int typeID, int modelID)`:

ראשית נבדוק האם הטיפוס נמצא בעץ car_types **(O(log(n)) זמן)**, והאם מספר הדגם חוקי.
לאחר מכן נבדוק האם הדגם כבר נמכר בעבר **(O(1) זמן)** ואם כן נסיר את הציון הישן שלו מעץ הציונים model_grades **(O(log(M)) זמן, O(log(M) מקום)**.

כעת נמכור את הדגם ונעדכן במערכים של הטיפוס את הציון ואת כמות המכירות **(O(1) זמן)**, ואם זו מכירה ראשונה של הדגם נסיר אותו מהרשימה unsold_list ע"י שימוש במצביע שנמצא במערך unsold_models_ptrs **(O(1) זמן)**. לאחר מכן נבדוק אם מספר הדגמים שלא נמכרו שווה ל0, אם כן אז נסיר את המצביע לרשימה unsold_list מהעץ unsold_models ונעדכן את המצביע לצומת השמאלי ביותר של העץ unsold_models **(O(log(n)) זמן, O(log(n) מקום)**.

לבסוף נוסיף את הדגם עם הציון החדש אל עץ הציונים model_grades ונעדכן את המצביע לצומת השמאלי ביותר של model_grades **(O(log(M)) זמן, O(log(M) מקום)**. אם הדגם הנוכחי הפך להיות הדגם הנמכר ביותר של הטיפוס, נסיר את הדגם הישן מעץ הדגמים הנמכרים ביותר car_types_best_sellers ונוסיף את הדגם הנ"ל **(O(log(n)) זמן, O(log(n) מקום)**.

סיבוכיות זמן: O(log(n) + log(M)) **סיבוכיות מקום:** O(log(n)+log(M))

5. `StatusType MakeComplaint(void* DS, int typeID, int modelID, int t)`:

ראשית נבדוק אם הטיפוס קיים בעץ car_types **(O(log(n)) זמן)** ואם מספר הדגם חוקי.
לאחר מכן נסיר את ציון הדגם הישן מהעץ model_grades **(O(log(M)) זמן, O(log(M) מקום)**
ונעדכן את הציון של הדגם במערך הציונים ששייך לטיפוס typeId **(O(1) זמן)**.
לבסוף נוסיף בחזרה את ציון הדגם לעץ הציונים model_grades ונעדכן את המצביע לצומת השמאלי ביותר בעץ model_grades **(O(log(M)) זמן, O(log(M) מקום)**.

סיבוכיות זמן: O(log(n) + log(M)) **סיבוכיות מקום:** O(log(n)+log(M))

6. `StatusType GetBestSellerModelByType(void* DS, int typeID, int* modelID)`:

אם modelID = 0, נבדוק אם העץ ריק **(O(1) זמן)**, ואם הוא לא, נחזיר את הערך best_seller שדאגנו לעדכן בכל פעם שביצענו מכירה של דגם \ הסרנו טיפוס **(O(1) זמן)**.
אחרת, נמצא את הטיפוס המבוקש **(O(log(n)) זמן, O(log(n)) מקום)** ונחזיר את הערך best_model שנמצא בתוך הטיפוס CarType **(O(1) זמן)**.

סיבוכיות זמן: O(log(n)) **סיבוכיות מקום:** O(1)

7. `StatusType GetWorstModels(void* DS, int numOfModels, int* types, int* models)`:

ראשית נבדוק האם יש מספיק דגמים במערכת **((O(1) זמן)**, ואז ניגש לצומת השמאלי ביותר של העץ model_grades השמור בworst_model **((O(1) זמן)**, וכל עוד הציון של הצומת הנוכחי קטן ממש מ-0, נשמור אותו במערכים ונעבור לצומת שנמצא אחרי הצומת הנוכחי בסיור In Order **(ומכיוון שזה עץ AVL פעולה זו תיקח O(1) זמן)**.

אם הגענו לדגם עם ציון גדול או שווה ל-0, נתחיל לעבור במקביל על העץ unsold_models (עץ אפסים) מהצומת השמאלי ביותר ששמור ב left_most_unsold_list **((O(1) זמן)**, כאשר בכל איטרציה נשווה בין הדגמים בעצים ונבחר את הקטן מביניהם, ונקדם את הצומת בעץ המתאים לצומת שבא אחריו בסיור In Order **((O(1) זמן)**. נעבור במקביל על שני העצים עד שנגיע לסוף של עץ האפסים ואז נעבור רק על model_grades, או עד שבאיטרציה על model_grades הגענו לדגם עם ציון גדול מ-0, ואז נעבור על עץ האפסים עד סופו (או שmodel_grades נגמר).

לבסוף, אם עץ האפסים unsold_models נגמר, נמשיך לעבור בסיור In Order על model_grades עד שנמלא את המערכים. שמירת כל צומת במערכים לקחה O(1) ולכן סה"כ הפעולה תהיה בסיבוכיות זמן O(numOfModels).

סיבוכיות זמן: O(m) **סיבוכיות מקום:** O(1)

8. `void Quit(void** DS)`:

כל עץ AVL שנמצא בCarDealershipManager נמחק בעזרת סיור Post Order על צמתיו מבלי להחזיר את העץ לאיזון בכל מחיקה **((O(n) זמן, ((O(log(n) מקום, כאשר n - מס' הצמתים בעץ).**

את הטיפוס CarType שנמצא בעץ car_types ניתן למחוק ב(O(numOfModels זמן ע"י מחיקת המצביע לרשימה שמחזיקה את כל הדגמים שלא נמכרו, סה"כ מחיקת car_types תיקח O(n+M) זמן ו- O(log(n)) מקום.

GradeKey ו-SaleType נמחק ב(O(1 זמן, לכן מחיקת model_grades תיקח O(M) זמן ו-O(log(M)) מקום, ומחיקת car_types_best_sellers O(n) זמן ו-O(log(n)) מקום.

סיבוכיות זמן: O(n + M) **סיבוכיות מקום:** O(log(n)+log(M))