עבודה ב OOP **31695** סמסטר א' תשפ"א, עבודה מסכמת
תאריך ההגשה : *15/02/2021*
שם הסטודנט, תעודת זהות, מייל : ולדימיר סרוב, 317569341, farengate93@gmail.com
שם הסטודנט, תעודת זהות, מייל : ניב כהן, 312142524, fusioniv10@gmail.com
ציון:

תיאור התוכנית

תיאור של מחלקות התוכנית

1. Date:

מחלקה המציינת תאריך, יודעת לוודא כי הוא תקני תוך כדי התחשבות גם בשנים מעוברות. ה-constructor ה-default-י של המחלקה יוצר את התאריך הנוכחי של היום. קיימים שני constructors נוספים ליצירה מתוך מספרים ממשיים ומתוך string.

- למחלקה קיימים operator overloads לoperators : "=", "<" ו- "<<".
- מתודות אשר קיימות במחלקה:

inc_years – מקבלת מספר ממשי ומגדילה את השנים במספר זה תוך התחשבות בשנה מעוברת.

expires_in_n_years – מקבל מספר ממשי ובודקת האם התאריך פג לאחר מספר שנים זה. במידה וכן מחזירה true.

- Exceptions:

ב-constructors במידה והתאריך אינו תקני, זורק את העצם עצמו.

ב-constructor מ-string יכול גם לזרוק הודעת שגיאה במידה ומבנה ה-string אינו תקין.

inc_years - במידה והמספר שנים המתקבל לאחר פעולה הוא שלילי זורק את העצם עצמו.

2. Time:

מחלקה המציינת תאריך ושעה, אשר יורשת ממחלקת Date, יודעת גם היא לוודא כי זמן ותאריך תקניים תוך כדי התחשבות גם בשנים מעוברות. ה-constructor ה-default-י של המחלקה יוצר את התאריך והשעה הנוכחיים של היצירה. קיים constructor נוסף ליצירה מתוך string.

- למחלקה קיים operator overload לoperator : "<<".
- Exceptions:

ב-constructors במידה והתאריך והשעה אינם תקנים, זורק את העצם עצמו.

ב-constructor מ-string יכול גם לזרוק הודעת שגיאה במידה ומבנה ה-string אינו תקין.

3. Driver:

מחלקה המתארת נהג. קיימים לנהג נתונים אישיים ונתוני רישיון ומספר עבודה המשויכת (במידה ולא קיימת הערך הוא 0). קיים לנהג constructor המקבל את כל הנתונים הרלוונטיים ליצירתו.
אחד הנתונים – סוג רישיון הוא מטיפוס enum LicenseType אשר יכול לקבל רק את הערכים: {C1,C,CplusE}, בנוסף קיים עבורו operator overload לoperator "<<".

- למחלקה קיימים operator overloads לoperator : "<<"
נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

valid_dates (private) – בודקת כי כלל התאריכים שנמצאים בנתוני הנהג תקניים מבחינה כרונולוגית ואם הוא מעל 18.

assign_job – מקבלת מספר עבודה ומשייכת את המספר לנהג.

remove_job – מאפסת את מספר העבודה.

suitable_for_driving – בודקת כי הרישיון לא פג תוקף, במידה ולא מחזירה true.

str_to_LicenseType (static) – מקבלת string ומחזירה enum LicenseType.

str_to_need_glasses (static) – מקבלת string ומחזירה bool.

create_Driver_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני הנהג בהתאמה מקצה זיכרון עבורו בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructor במידה והתאריכים אינם תקינים כרונולוגית זורק הודעת שגיאה מתאימה, במידה והשם אינו תקני זורק הודעת שגיאה מתאימה, במידה ומספר ת.ז. או מספר רישיון אינם תיקנים זורק את המספר הלא תיקני.

assign_job זורקת הודעת שגיאה במידה ולנהג כבר משויכת עבודה.

str_to_LicenseType זורקת הודעת שגיאה במידה ואין התאמה לאף אחד מסוגי הרישיון האפשריים.

str_to_need_glasses זורקת הודעת שגיאה במידה וה-string לא היה YES או NO.

set_name ו-set_surname במידה והשם אינו תקני זורקות הודעת שגיאה.

set_date_of_receipt_of_latest_permit ו-set_expiraton_date זורקים הודעת שגיאה במידה והתאריך לא תקין כרונולוגית.

create_Driver_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידה שבתוכו אינם תקינים.

4. <u>DriverList</u>:

מחלקה המכילה רשימות נהגים לפי ת"ז ולפי מספר רישיון. רשימות הנ"ל ממומשות על בסיס מבנה נתונים מסוג map<int,Driver*> מתוך ה-stl. ממשנו את המחלקה על בסיס שתי maps כדי לאפשר גישה נוחה ומהירה גם דרך ת"ז וגם דרך מספר רישיון, תוך כדי שמירה על חד ערכיות של הת"ז ומספר רישיון. למחלקה constructor מ-string אשר מקבל את שם הקובץ בפורמט csv ויוצר ממנו נהגים אשר מוקצה לטובתם זיכרון. יש לה default constructor אשר מומש בכדי שיהיה אפשר ליצור עצם ללא נהגים.

- למחלקה קיימים operator overloads לoperator : "<<"
  נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

suitable_driver (private) – מקבלת *DeliveryJob ומספר ת"ז. בודקת האם הנהג מתאים למשימה שהתקבלה.

add_new_driver – מקבלת *Driver בודקת שהוא לא קיים באף אחת מהמפות ומבצעת השמה לכל אחת מהמפות.

remove_driver_by_id/license – מקבלת מספר ת"ז או רישיון נהיגה ומוחקת את הנהג במידה והוא לא משויך לעבודה.

assign_job – מקבלת מספר ת"ז ומספר עבודה. בודקת האם הנהג קיים ואם כן קוראת ל-assign_job של Driver.

remove_job – מקבלת מספר ת"ז. בודקת האם הנהג קיים ואם כן קוראת ל-remove_job של Driver.

edit methods – קודם כל בודקות האם הנהג קיים ואם כן קוראות למתודת set של Driver המתאימה.

show methods – מדפיסות את רשימת הנהגים לפי סוג רישיון, סטטוס שיוך לעבודה, תאריך תפוגה של הרישיון. הדפסת רשימת הנהגים בצורה מצומצמת וסינון נהגים על פי התאמה למשימה מסוימת.

- Exceptions:

ב-constructor מ-string במידה והקובץ שהתקבל אינו נפתח זורק את שם הקובץ. במידה ואחד הנהגים שהיו צריכים להיווצר מהקובץ לא תקין או ישנה כפילות של מספר ת"ז או רישיון. סוגר את הקובץ משחררת את הזיכרון שהוקצה עד כה וזורק את שם הקובץ.

add_new_driver במידה וקיימת כפילות של מספר ת"ז או רישיון זורק את הנהג שהתקבל.

get driver methods זורקות מספר ת"ז או מספר רישיון שהתקבל במידה ולא נמצא.

remove driver methods זורקות מספר ת"ז או מספר רישיון שהתקבל במידה ולא נמצא. במידה והנהג משויך למשימה זורק את הנהג.

assign/remove_job זורקות מספר ת"ז שהתקבל במידה ולא נמצא.

edit methods זורקות מספר ת"ז שהתקבל במידה ולא נמצא.

5. **Truck:**

מחלקה מורישה (Base Class) המתארת משאית בסיסית. קיימים למשאית נתונים טכניים ונתונים כלליים ומספר עבודה המשויכת (במידה ולא קיימת הערך הוא 0). קיים למשאית constructor המקבל את כל הנתונים הרלוונטיים ליצירתו.
אחד הנתונים – סוג משאית הוא מטיפוס enum TruckType אשר יכול לקבל רק את הערכים: {Truck,SemiTrailer,Refrigeration,Tanker}, בנוסף קיים עבורו operator overload לoperator "<<".

- למחלקה קיימים operator overloads לoperator: "<<"
  נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

valid_tests (virtual) – בודקת אם כלל הבדיקות של המשאית בתוקף, על ידי שימוש ב-expires_in_n_years של Date.

assign_job – מקבלת מספר עבודה ומשייכת את המספר למשאית.

remove_job – מאפסת את מספר העבודה.

str_to_TruckType (static) – מקבלת string ומחזירה enum TruckType.

create_Truck_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני המשאית בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructor בודק את לוחית רישוי במידה ולא תקינה זורקת אותה. בודק תקינות של שאר הנתונים במידה ואחד מהם לא תקין זורקת הודעת שגיאה מתאימה.

assign_job זורקת הודעת שגיאה במידה ולמשאית כבר משויכת עבודה.

str_to_TruckType זורקת הודעת שגיאה במידה ואין התאמה לאף אחד מסוגי המשאיות האפשריים.

set_test_date במידה והתקבל תאריך ישן יותר ממה שהיה זורק הודעת שגיאה מתאימה.

add_mileage במידה והתקבל מספר שלילי זורק הודעת שגיאה מתהימה.

create_Truck_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

6. **SemiTrailer:**

מחלקה יורשת (Derived Class) מ-Truck המתארת משאית סמיטריילר. קיימים לסמיטריילר בנוסף לנתוני משאית את נתוני הגרור. קיים לסמיטריילר constructor המקבל את כל הנתונים הרלוונטיים ליצירתו.
וקיים constructor אשר מקבל *Truck ואת נתוני הגרור המשלימים.

- מתודות אשר קיימות במחלקה:

create_SemiTrailer_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני משאית הסמיטריילר בהתאמה מקצה זיכרון עבורו בצורה דינמית ומחזירה את המצביע לכתובתו.

get_max_load_weight מחזירה את הנמוך בין השניים – העמסה מקסימלית של הגרור או העמסה מקסימלית של המשאית פחות משקל הגרור.

- Exceptions:

ב-constructors בודק את לוחית רישוי במידה ולא תקינה זורקת אותה. בודק תקינות של שאר הנתונים במידה ואחד מהם לא תקין זורקת הודעת שגיאה מתאימה.

create_SemiTrailer_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

7. **RefrigerationTruck:**

מחלקה יורשת (Derived Class) מ-Truck המתארת משאית קירור. קיימים למשאית קירור בנוסף לנתוני משאית את נתוני המקרר. קיים למשאית קירור constructor המקבל את כל הנתונים הרלוונטיים ליצירתו.
וקיים constructor אשר מקבל *Truck ואת נתוני המקרר המשלימים.

- מתודות אשר קיימות במחלקה:

create_RefrigerationTruck_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני משאית הקירור בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructors בודק את לוחית רישוי במידה ולא תקינה זורקת אותה. בודק תקינות של שאר הנתונים במידה ואחד מהם לא תקין זורקת הודעת שגיאה מתאימה.

create_ RefrigerationTruck_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

set_refrigerator_test_date במידה והתקבל תאריך ישן יותר ממה שהיה זורק הודעת שגיאה מתאימה.

8. <u>TankerTruck</u>:

מחלקה יורשת (Derived Class) מ-Truck המתארת מכלית. קיימים למכלית בנוסף לנתוני משאית את נתוני המיכל. קיים למיכלית constructor המקבל את כל הנתונים הרלוונטיים ליצירתו. וקיים constructor אשר מקבל *Truck ואת נתוני המיכל המשלימים.

- מתודות אשר קיימות במחלקה:

create_TankerTruck_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני מכלית בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

get_max_load_weight מחזירה את הנמוך בין השניים – העמסה מקסימלית של המיכל או העמסה מקסימלית של המשאית פחות משקל המיכל.

- Exceptions:

ב-constructors בודק את לוחית רישוי במידה ולא תקינה זורקת אותה. בודק תקינות של שאר הנתונים במידה ואחד מהם לא תקין זורקת הודעת שגיאה מתאימה.

create_TankerTruck_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

set_sealing_test_date במידה והתקבל תאריך ישן יותר ממה שהיה זורק הודעת שגיאה מתאימה.

set_tank במידה ואחד הנתונים לא תקין זורק הודעת שגיאה המתאימה.

9. <u>TruckFleet</u>:

מחלקה המכילה רשימת משאיות לפי לוחית רישוי. רשימה זו ממומשת על בסיס מבנה נתונים מסוג map<int,Truck*> מתוך ה-stl. למחלקה constructor מ-string אשר מקבל את שם הקובץ בפורמט csv ויוצר ממנו משאיות אשר מוקצה לטובתם זיכרון. יש לה default constructor אשר מומש בכדי שיהיה אפשר ליצור עצם ללא משאיות.

- למחלקה קיימים operator overloads לoperator: "<<"
  נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

suitable_truck (private) – מקבלת *DeliveryJob ומספר לוחית רישוי. בודקת האם המשאית מתאימה למשימה שהתקבלה.

create_Truck_from_string (private) מקבלת string בפורמט csv מוציאה ממנו את סוג המשאית וקוראת למתודה סטטית המתאימה. לאחר ביצוע מחזירה *Truck.

add_new_truck – מקבלת *Truck בודקת שהיא לא קיימת במפה ומבצעת השמה.

remove_truck – מקבלת מספר לוחית רישוי ומוחקת את המשאית במידה והיא לא משויכת לעבודה.

assign_job – מקבלת מספר לוחית רישוי ומספר עבודה. בודקת האם המשאית קיימת ואם כן קוראת ל-assign_job של Truck.

remove_job – מקבלת מספר לוחית רישוי. בודקת האם המשאית קיימת ואם כן קוראת ל-remove_job של Truck.

edit methods – קודם כל בודקות האם המשאית קיימת ואם כן קוראות למתודת set של Truck המתאימה. קיימות מתודות המיועדות לסוג משאיות ספציפיות אשר נקראות ע"י שימוש בstatic cast המתאים.

show methods – מדפיסות את רשימת המשאיות לפי סוג משאית, סטטוס שיוך לעבודה, משקל העמסה מקסימלי, קילומטרז׳ ותאריך טסט. הדפסת רשימת המשאיות בצורה מצומצמת וסינון משאיות על פי התאמה למשימה מסוימת.

- Exceptions:

ב-constructor מ-string במידה והקובץ שהתקבל אינו נפתח זורק את שם הקובץ. במידה ואחד המשאיות שהיו צריכות להיווצר מהקובץ לא תקינה או ישנה כפילות של מספר לוחית רישוי. סוגר את הקובץ משחררת את הזיכרון שהוקצה עד כה וזורק את שם הקובץ.

add_new_truck במידה וקיימת כפילות של מספר לוחית רישוי זורק את המשאית שהתקבלה.

get_truck זורקת מספר לוחית רישוי שהתקבלה במידה ולא נמצא.

remove_truck זורקת מספר לוחית רישוי שהתקבלה במידה ולא נמצא. במידה והמשאית משויכת למשימה זורק את המשאית.

assign/remove_job זורקות לוחית רישוי במידה ולא נמצא.

edit methods זורקות מספר לוחית רישוי במידה ולא נמצא. בנוסף לכך עבור מתודות שמיועדות למשאיות ספציפיות, במידה והמשאית אינה מהסוג המתאים זורקות את סוג המשאית שהתקבלה.

create_Truck_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

10. <u>DeliveryJob</u>:

מחלקה מורישה (Base Class) המתארת משימת שילוח בסיסית. קיימים למשימה נתונים כללים ומספר ת״ז ולוחית רישוי של הנהג והמשאית המשויכים (במידה ולא קיימים הערך הוא 0). קיים למשימה constructor המקבל את כל הנתונים הרלוונטיים ליצירתה.
אחד הנתונים – סוג המשא הוא מטיפוס enum Cargo אשר יכול לקבל רק את הערכים: {regular, refrigerated_goods,liquid}, בנוסף קיים עבורו operator overload לoperator ״<<״.

- למחלקה קיימים operator overloads לoperator: ״<<״
נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

assign_truck_and_driver – מקבלת מספר ת״ז ולוחית רישוי ומבצעת השמה.

suitable_truck_and_driver (virtual) – מקבלת *Driver ו-*Truck ובודקת האם הם מסוגלים להיות משויכים למשימה ספציפית זו לפי הפרמטרים שלהם מול מגבלות המשימה.

str_to_Cargo (static) – מקבלת string ומחזירה enum Cargo.

str_to_ongoing_status (static and protected) - מקבלת string ומחזירה bool.

create_DeliveryJob_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני המשימה בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructor נבדקת תקינות של הנתונים במידה ואחד מהם לא תקין זורק הודעת שגיאה מתאימה.

assign_truck_and_driver זורקת הודעת שגיאה במידה ומספר ת״ז או מספר לוחית רישוי אינם תקינים זורקת הודעת שגיאה מתאימה.

str_to_Cargo זורקת הודעת שגיאה במידה ואין התאמה לאף אחד מסוגי המשימות האפשריים.

str_to_ongoing_status זורקת הודעת שגיאה במידה וה-string לא היה STATUS_ONGOING או STATUS_PENDING.

create_DeliveryJob_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

set methods בודקות את הנתון המתקבל וזורקות הודעת שגיאה מתאימה במידה ולא תקין.

11. RefrigeratedGoodsDelivery:

מחלקה יורשת (Derived Class) מ-DeliveryJob המתארת משימת שילוח קפואה. קיים למשימת שילוח קפואה בנוסף לנתוני משימת שילוח נתון על טמפרטורת קירור. קיים למשימת שילוח קפואה constructor המקבל את כל הנתונים הרלוונטיים ליצירתו. וקיים constructor אשר מקבל *DeliveryJob ואת טמפרטורת הקירור.

- מתודות אשר קיימות במחלקה:

create_RefrigeratedGoodsDelivery_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני משימת שילוח קפואה בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructor נבדקת תקינות של הנתונים במידה ואחד מהם לא תקין זורק הודעת שגיאה מתאימה.

create_RefrigeratedGoodsDelivery_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

12. LiquidGoodsDelivery:

מחלקה יורשת (Derived Class) מ-DeliveryJob המתארת משימת שילוח של נוזלים. קיים למשימת שילוח של נוזלים בנוסף לנתוני משימת שילוח נתון על נפח הנוזלים. קיים למשימת שילוח של נוזלים constructor המקבל את כל הנתונים הרלוונטיים ליצירתו. וקיים constructor אשר מקבל *DeliveryJob ואת נפח הנוזלים.

- מתודות אשר קיימות במחלקה:

create_LiquidGoodsDelivery_from_string (static) – מקבלת string בפורמט csv ויודעת להמיר אותו לנתוני משימת שילוח של נוזלים בהתאמה מקצה זיכרון עבורה בצורה דינמית ומחזירה את המצביע לכתובתו.

- Exceptions:

ב-constructor נבדקת תקינות של הנתונים במידה ואחד מהם לא תקין זורק הודעת שגיאה מתאימה.

create_LiquidGoodsDelivery_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

set_liquid_volume במידה והנפח נוזלים שהתקבל שלילי זורק הודעת שגיאה מתאימה.

13. JobList:

מחלקה המכילה רשימות משימות שילוח לפי מספר משלוח, רשימה אחת של משימות בהמתנה ושניה של משימות בתהליך. רשימות הנ"ל ממומשות על בסיס מבנה נתונים מסוג map<unsigned int,DeliveryJob*> מתוך ה-stl. למחלקה constructor מ-string אשר מקבל את שם הקובץ בפורמט csv ויוצר ממנו משימות שילוח אשר מוקצה לטובתם זיכרון. יש לה default constructor אשר מומש בכדי שיהיה אפשר ליצור עצם ללא משימות שילוח.

- למחלקה קיימים operator overloads לoperator : "<<"
  נפרדים עבור ostream ועבור ofstream (הדפסה לקובץ היא בפורמט csv).
- מתודות אשר קיימות במחלקה:

suitable_truck (private) – מקבלת *DeliveryJob ומספר לוחית רישוי. בודקת האם המשאית מתאימה למשימה שהתקבלה.

create_DeliveryJob_from_string (private) מקבלת string בפורמט csv מוציאה ממנו את סוג משימת השילוח וקוראת למתודה סטטית המתאימה. לאחר ביצוע מחזירה *DeliveryJob.

add_new_job – מקבלת *DeliveryJob מקצה לה את המספר משימה הבא ומכניסה למפה של משימות בהמתנה.

start_job – מקבל מספר משימה, מספר ת"ז של נהג ומספר לוחית רישוי. בודק שמשימה קיימת במפה של משימות בהמתנה .מעדכן את סטטוס המשימה מבצע השמה של המספר לוחית רישוי ושל מספר ת"ז למשימה ומעביר אותה למפה של משימות בתהליך.

remove_job – מקבלת מספר משימה ומוחקת את המשימה מתוך המפה המתאימה במידה וקיימת.

assign_job – מקבלת מספר לוחית רישוי ומספר עבודה. בודקת האם המשאית קיימת ואם כן קוראת ל-assign_job של Truck.

edit methods – קודם כל בודקות האם משימת השילוח קיימת ואם כן קוראות למתודת set של DeliveryJob המתאימה. קיימות מתודות המיועדות לסוג משימות ספציפיות אשר נקראות ע"י שימוש בstatic cast המתאים. במידה והמשימה בתהליך ניתן לשנות רק את התאריך יעד משימה.

show methods – מדפיסות את רשימת משימות השילוח לפי סוג משימה, סטטוס שיוך לעבודה, משקל מטען, מרחק, תאריך יעד משימה. הדפסת רשימת משימות השילוח בצורה מצומצמת.

- Exceptions:

ב-constructor מ-string במידה והקובץ שהתקבל אינו נפתח זורק את שם הקובץ. במידה ואחת ממשימות השילוח שהיו צריכות להיווצר מהקובץ לא תקינה או ישנה כפילות של מספר משימה. סוגר את הקובץ משחררת את הזיכרון שהוקצה עד כה וזורק את שם הקובץ.

start_job במידה ומשימת השילוח לא נמצאת בהמתנה אך נמצאת בתהליך, זורק הודעת שגיאה. במידה ולא קיימת כלל זורקת מספר המשימה שהתקבלה.

get_job זורקת מספר משימה שהתקבלה במידה ולא נמצאה באף אחת מהמפות.

remove_job זורקת מספר משימה שהתקבלה במידה ולא נמצאה באף אחת מהמפות.

edit methods זורקות הודעת שגיאה מתאימה במידה והמשימה בתהליך (פרט לעדכון תאריך יעד), אם זה לא נמצא במשימות בהמתנה זורקות מספר משימה. בנוסף לכך עבור מתודות שמיועדות למשימות ספציפיות, במידה ומשימת השילוח אינה מהסוג המתאים זורקות את סוג המשימה שהתקבלה.

create_DeliveryJob_from_string זורקת הודעת שגיאה במידה ומבנה ה-string או המידע שבתוכו אינם תקינים.

14. <u>User</u>:

מחלקה המתארת משתמש באפליקציה, מגדירה את השם שלו ושם משפחה שלו. יש לה constructor מקבל את שם, שם משפחה וסוג משתמש.

- למחלקה קיים operator overloads לoperator : "<<"

15. <u>Interface</u>:

מחלקה המתארת את ממשק האפליקציה. כילה אובייקטים מסוג DriverList, TruckFleet, JobList ו-User* ומפה <map<string,string[4]> המכילה את המשתמשים האחרונים באותה פתיחה של אפליקציה עד סגירתה לטובת גישה יותר מהירה. למחלקה יש default constructor ו-constructor המקבל שלושה strings אשר כל אחת מהן היא שם קובץ המתאים לרשימה מסוימת.

- מתודות אשר קיימות במחלקה:

login - מקבלת שם משתמש וסיסמה. בודקת האם המשתמש שמנסה להתחבר התחבר לאחרונה, במידה ולא קוראת ל-search_user_in_file. במידה והוחזר true מתחבר ושומר את המידע על התחברות אחרונה במפה המתאימה.

search_user_in_file (private) – מקבלת שם משתמש, סיסמה ומערך לשמירת נתוני משתמש באורך המידע. טוענת קובץ משתמשים, ומחפשת בו את ההתאמה של שם משתמש וסיסמה. במידה ולא נמצא מחזיר false. במידה ונמצא מחזיר true ושומר את המידע במערך המועבר.

שער המתודות מתארות תפריטים שונים והצגתם על המסך. מעבר בין התפריטים נעשה על ידי קלט מהמשתמש. מתודות של הוספה, עריכה או הסרה של נהג, משאית או משימת שלוח על ידי קלט משתמש. מתודות של הצגת נהג, משאית או משימה בודדת. מתודות של הצגת רשימות בסידורים שונים של כל אחת מהרשימות של נהגים, משאיות או משימות שילוח.

תיאור של אלגוריתמים המשומשים בתוכנית

1.ביצוע של הדפסה של רשימות על פי סידור מסוים

בחלק מההדפסות המיוחדות שביצענו היה שימוש במבנה נתונים מסוג multimap. כאשר העתקנו את ה-map הקיימת לתוך ה-multimap תוך כדי החלפה של ה-key לאחד ה-values שעל פיו רצינו למיין את ה-map ולאחר מכן הדפסנו את הmultimap. חשוב לציין כי הסידור ברירת מחדל של multimap הוא לפי אופרטור "<" לכן כל key שהוכנס ל-multimap היה צורך לממש עבורו אופרטור זה. האלגוריתם מסתמך על כך שה-multimap יכול להכיל מספר key שווים והוא מסדר את הvalues לפי סדר עולה של ה-key. סיבוכיות הזמן של האלגוריתם היא O(n*log(n)) מכיוון שיש לבצעו פעולת הכנסה n פעמים וכל הכנסה בעלת סיבוכיות לוגריתמית. סיבוכיות מקום של האלגוריתם היא O(n).

2. ביצוע של הדפסה למסך ולקובץ של העצמים מסוגים שונים של משאיות ומשימות מתוך הרשימות שלהם

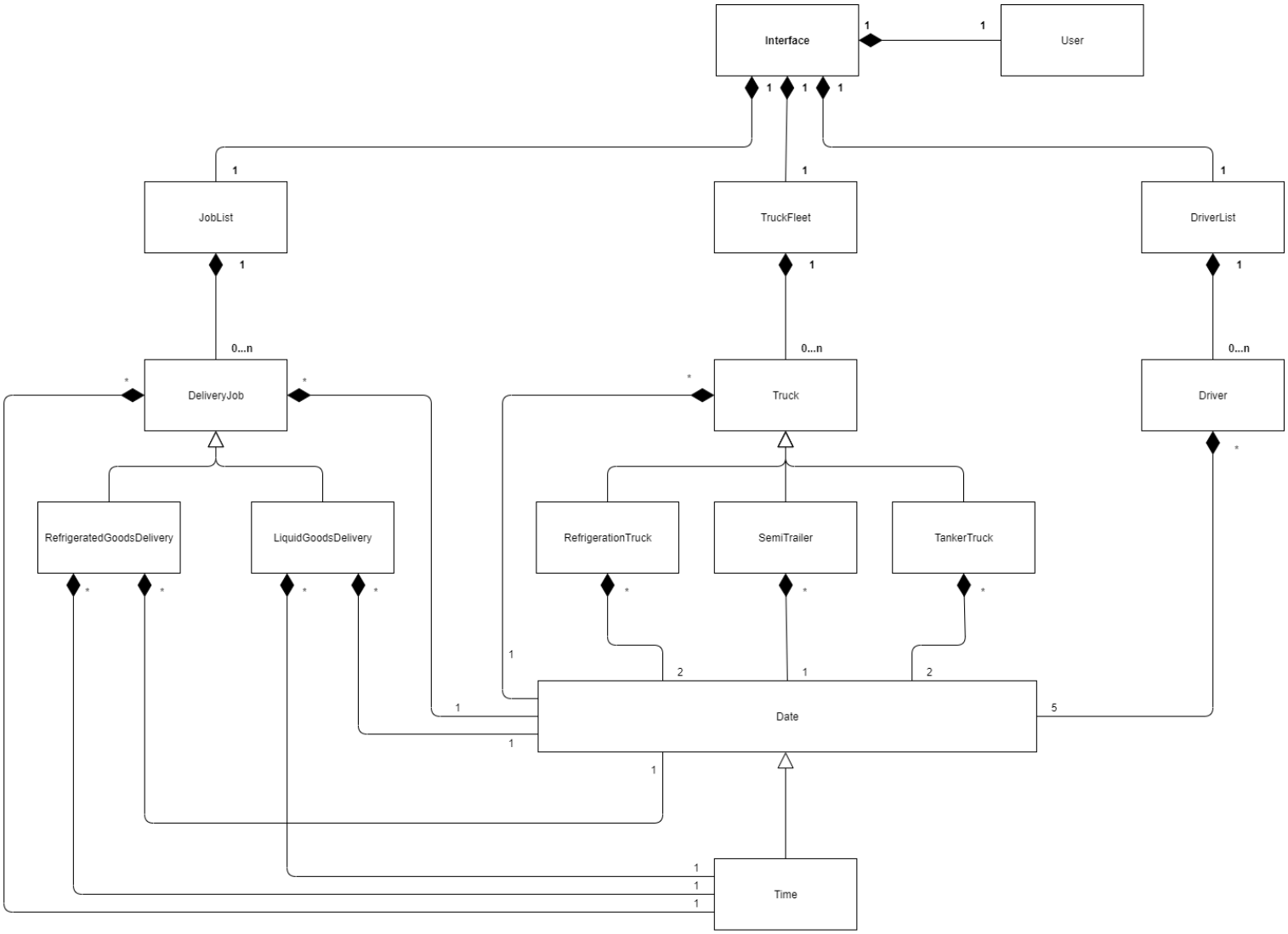
בוצע על ידי שימוש בעקרון הפולימורפיזם. בעת קריאה לאופרטור "<<" מתבצעת קריאה למתודה ווירטואלית של העצם הנדרש לפי ה-virtual table הנמצא אתו.

3. ביצוע של שמירת נתונים לתוך המסמך וטעינת נתונים מהמסמך

שמירה של נתונים לתוך קובץ בוצע בפורמט של csv, זאת אומרת שכל אחד מהאיברים מופרד באמצעות תו ',‘.

טעינה מהקובץ בוצע על ידי קריאה של שורה שלמה בתור string, המרתה ל-stringstream וחלוקה שלו לפי תווי ',‘ ולאחר מכן המרה של כל אחד מהאיברים לטיפוס המתאים על ידי פונקציות ספריה stoi,stoul,stod במקרה של טיפוסים פרימיטיביים או על ידי פונקציות אשר מומשו על ידינו וconstructors מ-string שמומשו על ידינו לשאר הטיפוסים. הדבר היה בר ביצוע מכיוון שעבדנו בפורמט csv.

יחסי גומלין בין המחלקות



ממשק למשתמש

הממשק מתבצע דרך מסך cmd בכל פעם מודפס למשתמש תפריט עם מספר אפשרויות, מתוכו הוא בוחר את אחד האפשרויות, שלוקחת אותו לתפריט חדש או מבצעת פעולה כגון הדפסה. המעבר בין התפריטים מתבצע באמצעות הקלדת המקש המתאים ולחיצה על enter. במידה והוקלד תו שאינו מתאים לאחת מהאפשרויות, המערכת מתעלמת ממנו וממתינה למקש מתאים.
הממשק כולל מערכת משתמשים משני סוגים, admin ו-user. כאשר ל-admin יש יותר הרשאות, כגון הוספה, הסרה ושינוי פרטי הנהגים ומשאיות. ההתחברות למערכת מתבצעת על ידי שם משתמש וסיסמה. בכניסה למערכת היא טוענת את הקבצים עם כלל המידע הנצבר עד כה. ביציאה מהמערכת מתבצעת שמירה אוטומטית לאותם קבצים.

מבני הנתונים בהם השתמשנו בתוכנית

מבנה הנתונים העיקרי ששימש אותנו בתוכנית הוא ה-map. מכלל מבני הנתונים הידועים לנו השתמשנו בו, מכיוון שהוא מקנה גישה מהירה לפריט ספציפי, מקנה הוספה והסרה יחסית מהירים וכל זאת תוך כדי שמירה על סדר תמידי ושמירה על חד-ערכיות של ה-key. הדבר שימש אותנו בכלל הרשימות מכיוון שהיינו צריכים חד-ערכיות של המספרי ת"ז, מספרי רישיון, מספרי לוחית רישוי של המשאיות ומספרי המשימות. בנוסף לכך שימוש ב-map הקנה לנו אפשרות להשתמש באלגוריתם הסידור שפירטנו עליו מקודם.

דוגמאות לשימוש בספרייה

הספרייה נבנתה לצורך בנייה על בסיסה אפליקציה לניהול של צי משאיות, נהגים ומשימות שילוח. ניתן להשתמש בה לטובת ניהול כל חברה אשר קיימים בה המרכיבים הללו. בהינתן אפשרות להרחבת הספרייה לסוגים נוספים של משאיות וסוגי משימות שילוח. ניתן גם להרחיב את כלל הניהול של הנהגים לצורך ניהול עובדים יותר ממוקד.
על בסיס הספרייה גם ניתן להציג דוחות כגון – רשימת משאיות עם טסטים שיש לבצע בקרוב, רשימת נהגים שיש לחדש להם את הרישיון בקרוב, רשימת נהגים שיוצאים לפנסיה בקרוב, רשימת משאיות אשר הקילומטרז' שלהן הוא גבוה מאיזו שהוא מספר ויש להחליפן וכו' .

מקורות

1. שימוש ב-map : https://www.cplusplus.com/reference/map/map/
2. שימוש ב-multimap : https://www.cplusplus.com/reference/map/multimap/
3. שימוש ב-string : https://www.cplusplus.com/reference/string/string/
4. שימוש ב-stringstream : https://www.cplusplus.com/reference/sstream/stringstream/
5. שימוש ב-ofstream : https://www.cplusplus.com/reference/fstream/ofstream/
6. שימוש ב-ifstream : https://www.cplusplus.com/reference/fstream/ifstream/
7. טבלאות UML 1 : http://www.umich.edu/~eecs381/handouts/UMLNotationSummary.pdf
8. טבלאות UML 2 : https://en.wikipedia.org/wiki/Class_diagram