פרוייקט סוף: מנהל קורסים

מגישים: גונורצקי אביטל ת.ז 211622956
עומר מרואני ת.ז 206944795

מרצה: יעל ארז

מבוא

כיום ,ניהול יעיל של קורסים, סטודנטים ומורים הוא חיוני להצלחת מוסדות אקדמאיים.

ספריית "מנהל קורסים" נועדה לספק פתרון גמיש ויעיל לניהול כל המתחולל במוסד האקדמי

ספרייה זו מבוססת על שפת התכנות C++ ומציעה שימוש של כלים פונקציונליים המאפשרים למורים ולתלמידים לנהל את התכנים שלהם בצורה נקייה ויעילה.

בין הפיצ'רים שלה ניתן למצוא את היכול להוסיף, למחוק ולהציג קורסים ,לנהל שיעורים והרצאות. ובניית מערכות של קורסים ,הצגתם ושינויים ע"י הסטודנט.
נגישות: לספק ממשק פשוט ונוח לשימוש עבור מנהלי הקורסים והסטודנטים.

גמישות: לאפשר ניהול מותאם אישית של קורסים , הרצאות תרגולים ומעבדות בהתאם לצורך כגון: שינוי שעת קורס ,שינוי כיתה ועוד.

אחסון: לאחר כל יציאה מהמערכת , כל הנתונים ששונו במהלך פעילות המערכת יישמרו.

באמצעות ספרייה זו , מוסדות אקדמאיים יכולים לייעל את ניהול הקורסים והפעילויות הנלוות להם, תוך שיפור הבקרה והמעקב אחר הניהול השוטף של המוסד האקדמי.

מחלקות התוכנית

Class Person:

שימוש בפולימורפיזם ע"י מחלקת בסיס Person ומחלקות יורשות Student, Teacher.

שדה נתונים:

- תעודת זהות
- שם
- סיסמא

פונקציות עיקריות:

- CTR, CCTR, DTR
- get, set
- קריאת נתונים מקובץ CSV
- כתיבת נתונים לקובץ CSV
- פונקציות וירטואליות לממשק משתמש: כניסה למערכת, הדפסת מחלקה, הדפסת תפריט פעולות, שימוש בפעולות.
- Operator <<, operator=, operator[]

Class Student:

מחלקה זו יורשת מ- Person, ומוסיפה עוד משתנה בשם Schedule,ומשתנה סטטי שסופר כמות סטודנטים .

שדה נתונים:

- מספר סטודנטים
- מערכת(הסבר למטה).

פונקציות עיקריות:

- CTR, CCTR, DTR
- get, set
- קריאת נתוניSchedule מקובץCSV
- כתיבת נתוני Scheduleלקובץ CSV
- פונקציות וירטואליות במחלקת Person לממשק משתמש: כניסה למערכת, הדפסת מחלקה, הדפסת תפריט פעולות, שימוש בפעולות.

Class Teacher:

מחלקה זו יורשת מ- Person, מוסיפה משתנה סטטי שסופר כמות מורים .

שדה נתונים:

- מספר מורים

פונקציות עיקריות:

- CTR, CCTR, DTR
- get, set
- Serialization
- פונקציות וירטואליות במחלקת Person לממשק משתמש: כניסה למערכת, הדפסת מחלקה, הדפסת תפריט פעולות, שימוש בפעולות.

Class Course:

במחלקה זו יש את כל הנתונים הקשורים לקורס , בשדה הנתונים יש שימוש במחלקה אחרת הנקראת Lesson (הרצאות, תרגולים ומעבדות) .

שדה נתונים:

- מספר קורס
- שם הקורס
- שם מרצה אחראי
- מספר נקודות זכות
- הרצאות
- תרגולים
- מעבדות

פונקציות עיקריות:

- CTR, CCTR, DTR
- get, set
- קריאת נתונים מקובץ CSV
- כתיבת נתונים לקובץ CSV, Serialization
- operator<<, operator=

Class Lesson:

מחלקה זו מתארת שיעור בכלליות, ניתן להשתמש במחלקה זו לתיאור הרצאה\תרגול\ מעבדה.

למחלקה זו אין כתיבת וקריאה מקובץ, זה נעשה בפונקציה חיצונית כדי שתמומש ע"י template.

שדה נתונים:

- מספר קבוצה
- יום
- שעת התחלה
- משך השיעור
- שם המורה
- כיתה

פונקציות עיקריות:

- CTR, CCTR, DTR
- get, set
- Serialization
- operator<<, operator=, operator==

Class Manage:

המחלקה הראשית שמנהלת את המערכת ,היא מחזיקה את כל הרשומות של האנשים והקורסים .

הקריאה והכתיבה לקבצים תיעשה בתחילת ובסוף התוכנית.

שדה נתונים:

- מפה לא מסודרת, **מפתח**: ת"ז, **ערך**: מצביע ל - Persons (פולימורפיזם)
- מפה לא מסודרת, **מפתח**: מספר קורס, **ערך**: Course
- משתנה שתפקידו לדעת אם כבר הודפסו 10 קורסים ראשונים וכו'

פונקציות עיקריות:

- CTR, DTR
- קריאת נתונים מקובץ CSV
- כתיבת נתונים לקובץ CSV

פונקציות Interface:

- הפעלת המערכת
- כניסה למערכת
- בדיקת ת"ז חוקי
- בדיקת סיסמא סטודנט
- בדיקת סיסמא מורה(אם יש)

פונקציות כלל המשתמשים:

- הדפסת קורס לפי מספר קורס
- הדפסת עשר קורסים ראשונים
- הדפסת 10 קורסים הבאים
- הדפסת אנשים ???

פונקציות עבור סטודנט:

- הדפסת מערכת
- הדפסת כל המערכות
- הוספת מערכת
- מחיקת מערכת
- חיפוש קורס לפי מספר קורס

פונקציות עבור מורה:

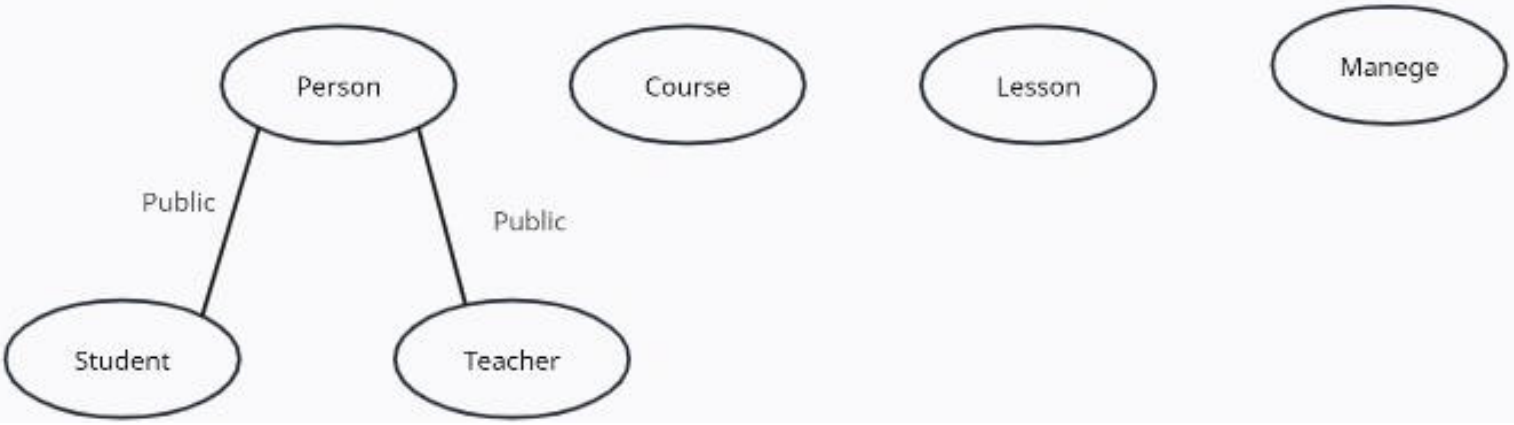
- הוספת קורס\ מרצה\סטודנט
- מחיקת קורס\ מרצה\סטודנט
- חיפוש כללי (חיפוש קורס, חיפוש שיעור וחיפוש בן אדם)

פונקציות חיצוניות:

- בדיקת זמן חוקי (בהדפסת מערכת לבדוק האם יום ושעת השיעור הם כמו בלולאה)
- הדפסת מערכת (מימוש ההדפסה)
- רישום שורות , ימים לטבלה.
- חיפוש קורס מסוים , פונקציית template לחיפוש לפית"ז, שם, נקודות זכות.
- חיפוש שיעור, שימוש בtry catch.

דיאגרמת UML

מחלקות ראשיות:



PERSON

Private:
id, name, password

Public:
constructor, copy constructor, destructor
get, set
reading from csv
writing to csv (מחזיר מפה)
Operator <<, operator=, operator[]
virtual functions :
join(כניסה למערכת) ,
print_actions_menu(הדפסת תפריט פעולות),
use_actions(שימוש בפעולות) ,
print(מדפיס סיסמה במידה ויש) ,
serialization(דחיסת נתונים ל למחרוזת בהפרדה בפסיקים)

Student

Private:
Inherited Privates from Person.
Number_student (משתנה סטטי)
Schedules(ווקטור שמחזיק מפות כאשר כל מפה זה מערכת נפרדת)

Public:
Constructor, Copy Constructor, Destructor
get , set.
Read_schedule (קריאת נתוני Schedule מקובץ CSV)
Write_schedule(כתיבת נתוני Schedule לקובץ CSV)

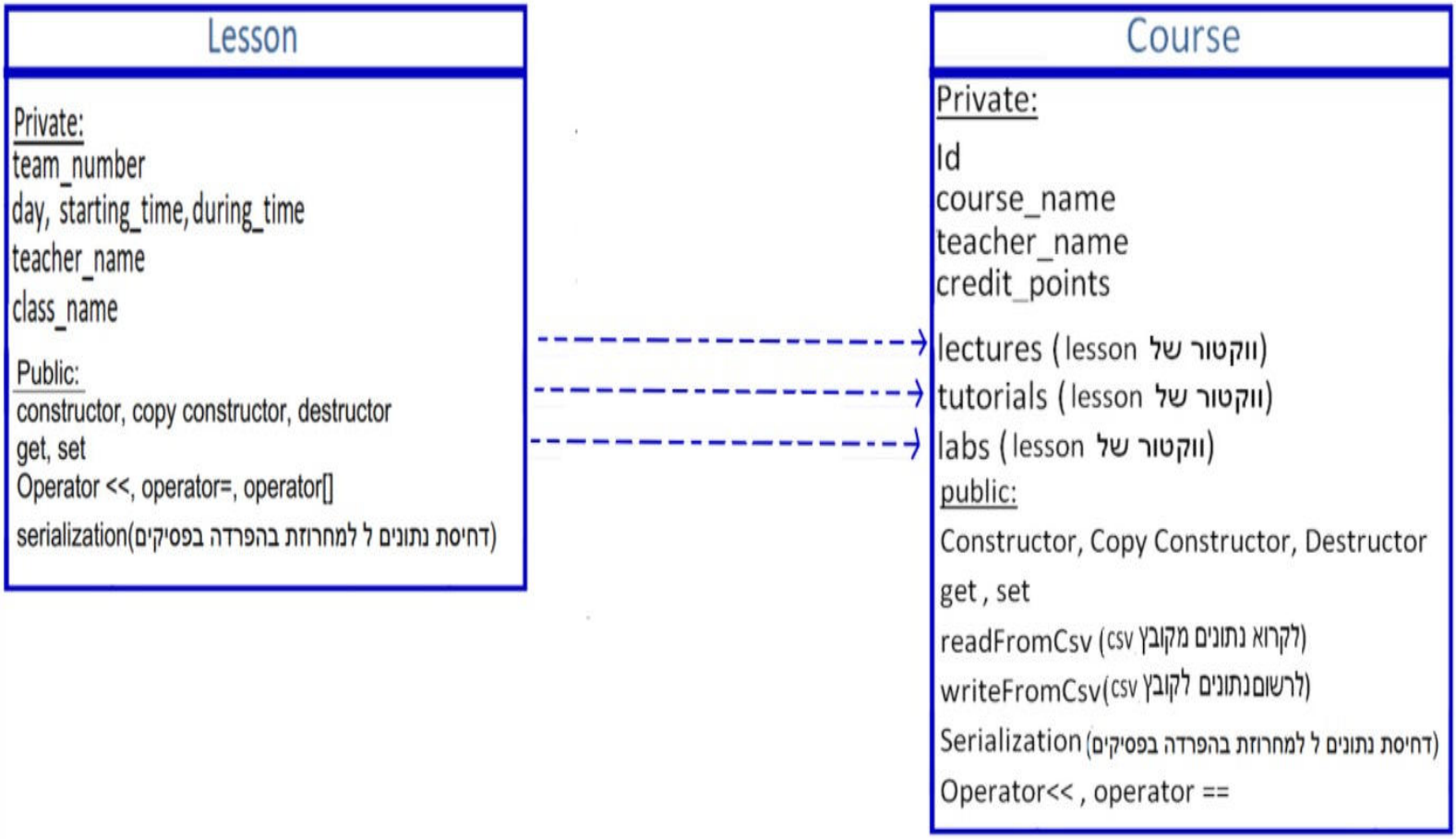
Virtual functions from Person
Join(כניסה למערכת)
Print(מדפיס סיסמה במידה ויש)
Serialization(דחיסת נתונים ל למחרוזת בהפרדה בפסיקים)
Print_action_menu (הדפסת תפריט פעולות)
Use_actions(שימוש בפעולות)

Teacher

Private:
Inherited Privates from Person.
number_teachers (משתנה סטטי)

Public:
Constructor, Copy Constructor, Destructor
get , set.

Virtual functions from Person
Join(כניסה למערכת)
Print(מדפיס סיסמה במידה ויש)
Serialization(דחיסת נתונים ל למחרוזת בהפרדה בפסיקים)
Print_action_menu (הדפסת תפריט פעולות)
Use_actions(שימוש בפעולות)


Lesson
Private:
team_number
day, starting_time, during_time
teacher_name
class_name
Public:
constructor, copy constructor, destructor
get, set
Operator <<, operator=, operator[]
serialization(דחיסת נתונים ל למחרוזת בהפרדה בפסיקים)
Course
Private:
Id
course_name
teacher_name
credit_points
lectures (וקטור של lesson)
tutorials (וקטור של lesson)
labs (וקטור של lesson)
public:
Constructor, Copy Constructor, Destructor
get , set
readFromCsv (לקרוא נתונים מקובץ csv)
writeFromCsv(לרשום נתונים לקובץ csv)
Serialization (דחיסת נתונים ל למחרוזת בהפרדה בפסיקים)
Operator<< , operator ==

Manage

Private:

persons (מפה לא מסודרת של אנשים) ← PERSON

courses (מפה לא מסודרת של קורסים) ← Course

ten_courses (משתנה שסופר אם הודפסו כבר 10 קורסים)

public:

Constructor, Copy Constructor, Destructor

get , set

readFromCsv (לקרוא נתונים מקובץ csv)

writeFromCsv(לרשום נתונים לקובץ csv)

Interface:

Start(הפעלת מערכת)

Join(כניסה למערכת)

Id_valid(בדיקת ת"ז חוקי)

Password_valid (בדיקת סיסמא סטודנט)

Password_admin (בדיקת סיסמא מורה (אם יש))

User funcs:

Print_by_course_id()

Print_ten_courses()

Print_more_ten()

Print_persons()

Student func:

print_schedule()

print_all()

add_schedule()

rm_schedule()

search_course_id() "by id"

Teacher func:

add_course() , rm_course()

add_lecturer() , rm_lecturer()

add_student)_, rm_student()

searc() "mostly everything"

הממשק למשתמש

1. **מסך התחברות:** במסך ההתחברות המשתמש נדרש לבחור כניסת student or admin .
אם בחר כניסת סטודנט עליו להקליד ת"ז וסיסמא, לאחר מכן המערכת בודקת האם המשתמש אכן קיים במערכת ומאפשרת את כניסתו.
אם בחר כניסת אדמין,עליו להקליד ת"ז , בכניסה הראשונה \ לא שינה סיסמא , אין צורך להזין סיסמא . המערכת בודקת האם ה-ת"ז קיים במערכת, לאחר הכניסה תהיה למורה אופציה להחליף סיסמה אם יבחר להחליף, כניסתו הבאה תהיה דומה לכניסת הסטודנט.

2. **תפריט ראשי:** בתפריט הראשי ישנם אופציות זהות לכל המשתמשים כגון: הדפסת קורס לפי מספר קורס, הדפסת כל הקורסים (10 ראשונים), הדפסת 10 קורסים הבאים ברשימה .
<u>אופציות סטודנט</u>: בניית מערכת, הדפסת מערכת,הדפסת כל רשומות של קורס לפי מספר קורס.
<u>אופציות מורה</u>:הוספת קורס, מחיקת קורס , הוספת מרצה, מחיקת מרצה, הוספת סטודנט, מחיקת סטודנט,חיפוש טקסט(קורסים, מרצים וסטודנטים).

3. **קלט:** בקליטת הקלט המשתמש יצטרך להזין את מה שהוא התבקש, למשל אם בחר פונקצית הוספת קורס , לאחר מכן יתבקש להזין את הקלט מספר קורס,שם קורס, שם מרצה אחראי , מספר נקודות זכות.
הכנסת נתונים בצורה הזאת ניתן לדעת לפי שם הפקודה,למשל: AddCourse.
בעוד שיש פונקציות שבשורת הפקודה מזינים נתונים למשל:
Add <schedule_id><course_id><group_id>

4. **תצוגת נתונים:** לאחר ביצוע הפקודה הקשורה לתצוגת נתונים ,המשתמש יקבל על המסך הודעה הקשורה לפקודה שהכניס, למשל: חיפוש קורס,אם הקורס קיים על המסך יוצג המידע על הקורס.

מבני נתונים

- מבני הנתונים העיקריים הם מספריית STL כגון: וקטור ,מפה לא מסודרת, מערך.
- וקטור: במחלקת Course , יש 3 וקטורים שונים של מחלקת Lesson , בשם הרצאות תרגולים ומעבדות, השימוש ווקטור הוא קל לתחזוקה ונגיש ,הכנסת והוצאת איבר ,הגדלת והקטנת הווקטור, ומאחר שאין צורך לחיפוש שיעור מסוים לפי ערך מפתח כלשהו, וקטור היא ההחלטה היעילה ביותר פה.
- מפה לא מסודרת: שימוש במחלקת Manage , לייצוג מפה של אנשים, ומפה של קורסים. השימוש במפה זו היא יעילה ביותר כיוון שרוב העבודה בשדות אלו שנעשית היא חיפוש במפות סטודנט\ קורס מסוימלפי ת"ז\מספר קורס שאלו הם גם המפתחות, ויעילות זו היא O(1). החיסרון הוא שבפונקצית חיפוש של המורה, צריך לחפש לפי שדות שונים של המחלקה , לכן שם אנו מבצעים חיפוש ע"פ איטרטורים על כל המפה.
- מערך: לייצוג schedule student אנו משתמשים במשתנה שמכיל בתוכו ,array<int,3> בחירה הכי יעילה כי גודלו קבוע, (מספר קבוצה הרצאה,תרגול,מעבדה) וקל לשימוש.

מערכת Schedule

למשתמש סטודנט יש בשדה נתונים משתנה בשם Schedule, שהוא מסוג:
Vector <unordered_map< int, array<int, 3>>> Schedule;
בדרך זו יש יעילות הכי גבוהה בשמירת נתונים על מערכות הסטודנט.
נסביר כיצד זה פועל, כרגע עבור איבר יחיד בתוך הווקטור (מערכת בודדת):
עבור מערכת , יש מפה לא מסודרת , **מפתח**: מספר קורס, **ערך:** מערך 3 מספרים.
המערך שנמצא בערך המפה מכיל את שלושת המספרים הקשורים לאותו הקורס שהם מספרי קבוצה של הרצאה ,תרגול ומעבדה.
בתוך מחלקה Manage, ברגע שנצטרך להשתמש בכל מה שקשור ל schedule, נוכל להמיר את הנתונים של הסטודנט לקורסים הרלוונטיים וכך להשתמש בהם היכן שצריך.
מה שאנו חוסכים מפה הוא הרבה משתנים מסוג Course עבור כל סטודנט.
הווקטור מייצג מערך של מערכות לפי בקשת הסטודנט.

מקורות

https://isocpp.org/wiki/faq/pointers-to-members

https://www.geeksforgeeks.org/cin-in-c/

https://www.geeksforgeeks.org/stringstream-c-applications/