**מבוא לתכנות מערכות 10010 סמסטר א׳ – תשפ"ד**

**פרויקט סיום**

המערכת תנהל ליגת כדורסל כאשר הליגה בנויה מקבוצות ושחקנים שקיימים במאגרים.

למערכת יש 3 קבצים עיקריים:

* ליגת הכדורסל (קובץ טקסט וקובץ בינארי)
* מאגר שחקנים (קובץ טקסט וקובץ בינארי)
* מאגר קבוצות (קבוץ טקסט וקובץ בינארי).

בעת העלאת המערכת המשתמש בוחר אם לטעון כל אחד מהקבצים באחת משתי דרכים: מקובץ בינארי או מקובץ טקסט. במידה וטעינה לא הצליחה המשתמש יאתחל באופן ידני את הרכיב שהעלאתו לא צלחה.

התפריט הראשי כולל:

* גישה לתפריט הפנימי של הליגה
* גישה לתפריט הפנימי של מאגר השחקנים
* גישה לתפריט הפנימי של מאגר הקבוצות

תפריט פנימי של הליגה כולל:

* הדפסה של טבלה של הליגה הכוללת : (מיקום הקבוצה בליגה, שם הקבוצה, קוד הקבוצה, מספר נקודות, מספר ריבואנדים, מספר אסיסטים).
* הדפסת קבוצה ספציפית מתוך הליגה (הדפסת כל נתוני הקבוצה)
* הדפסת כל השחקנים בליגה ונתוניהם
* טרייד בין שני שחקנים
* סיבוב ליגה

תפריט פנימי של מאגר השחקנים כולל:

* הדפסת מאגר השחקנים
* מיון השחקנים לפי נתון מסוים (שם השחקן, תעודת זהות, מדינה, עמדה על המגרש, מספר על הגופייה, משכורת, מספר נקודות, מספר ריבאונדים, מספר אסיסטים).
* הוספה של שחקן למאגר

תפריט פנימי של מאגר הקבוצות כולל:

* הדפסת מאגר הקבוצות
* מיון הקבוצות לפי נתון מסוים (שם הקבוצה, קוד הקבוצה, שם המאמן, תקציב, תקציב חופשי (תקציב פנוי), שם ההיכל הביתי).
* חיפוש קבוצה לפי נתון מסוים (שם הקבוצה, קוד הקבוצה, שם המאמן, שם ההיכל הביתי).
* הוספת קבוצה למאגר.

מכל תפריט פנימי ניתן לחזור לתפריט הראשי ולבחור אופציה נוספת.

לאחר שהמשתמש בוחר לצאת מן התוכנית, כל הרכיבים (ליגה, מאגר שחקנים, מאגר קבוצות) יישמרו ל2 סוגי קבצים (קובץ טקסט וקובץ בינארי).

הישויות העיקריות במערכת זו יהיו:

שחקן , Player, שעבורו נשמור :

1. שם השחקן
2. מספר זהות (חייב להיות מספר ייחודי בעל 5 ספרות)
3. האם השחקן נבחר לקבוצה בליגה
4. מדינה
5. עמדת השחקן על המגרש (נשמור 5 עמדות כדורסל באמצעות enum).
6. מספר השחקן (המספר שמופיע לו על הגופייה)
7. שכר
8. נתוני השחקן
9. שיוך השחקן לקבוצה (במידה והוא משויך לקבוצה יישמר קוד הקבוצה, במידה ולא משויך

יישמר "----"

*פונקציות:*

* initPlayer – מאתחלת את כל השדות של השחקן.
* למערכת יש פונקציית השוואה (מקבלת שני מצביעים לשני שחקנים בתור void\* וממירה אותם בתוך הפנקצייה Player\*) לפרמטרים הבאים: שם השחקן, מספר זהות, שכר, מדינה, נקודות שקלע, ריבואנדים שלקח, אסיסטים שמסר, עמדה על המגרש, מספר על הגופייה, קוד קבוצה אליה השחקן משוייך).
* למערכת יש פונקציית השמה (מקבלת מצביע לשחקן ומאתחלת את השדה הרצוי לנתון שהמשתמש הכניס) לפרמטרים הבאים: שם השחקן, מספר זהות, שם השחקן, מספר זהות, שכר, מדינה, נתונים סטטיסיטים (נקודות, ריבאונדים ואסיסטים. הפונקציה מזמנת פונקציה במבנה Stats) עמדה על המגרש, מספר על הגופייה, קוד קבוצה אליה השחקן משוייך).
* מספר פונקציות הדפסה: פונקציה המדפיסה את כל נתוני השחקן, פונקציה המדפיסה רק שחקן שלא נבחר לקבוצה).

מנהל שחקנים, PlayerMananger, שעבורו נשמור:

1. מערך שחקנים, כאשר כל תא במערך הוא מצביע לשחקן.
2. מספר השחקנים בטורניר
3. האם המאגר ממויין

*פונקציות:*

* initPlayerManager - (מקבלת מצביע למנהל השחקנים, שם של קובץ טקסט ושם של קובץ בינארי) – שואלת את המשתמש אם הוא רוצה לקרוא את המאגר מקובץ טקסט או מקובץ בינארי. במידה והטעינה שבחר לא מצליחה המשתמש יאתחל את המאגר בצורה ידנית באמצעות הפונקציה setPlayersList (מקבלת מצביע למנהל שחקנים).
* setPlayer - פונקציית השמה לכל שחקן (מקבלת מצביע למנהל שחקנים ולשחקן) – קוראת לכל אחת מפונקציות ההשמה במבנה Player ומאתחלת כל שחקן במאגר.
* checkIfPlayerInThePlayersList - פונקציה הבודקת האם שחקן נמצא במאגר (מקבלת מצביע למנהל שחקנים ולשחקן).
* printPlayersList - פונקציית הדפסה (מקבלת מצביע למנהל שחקנים ומצביע לפונקציה) – מדפיסה את כל השחקנים והנתונים שלהם.

נתונים, Stats, שעבורו נשמור:

1. נקודות
2. ריבאונדים
3. אסיסטים

*פונקציות:*

* initStats – מקבלת מצביע לStats מאתחלת את כל השדות ל0.
* פונקציות השמה – מקבלת מצביע לStats ומאתחלת אותו לפי בחירת המשתמש.
* setStatsRandomly – מקבלת מצביע לStats ומאתחלת את השדות באופן רנדומלי (נקודות בין 10-20, ריבאונדים ואסיסטים בין 0-10).
* פונקציות השוואה לכל אחד מהשדות (מקבלת שני מצביעים void\* עושה להם השמה לStats\* ומשווה ביניהם).
* printStats – מקבלת מצביע לstats ומדפיסה את הנתונים הסטטיסיטים.

סגל, Squad, שעבורו נשמור:

1. רשימה של השחקנים בקבוצה
2. מספר השחקנים בקבוצה
3. הוצאות הקבוצה (שכר מאמן ושכר שחקנים)

*פונקציות:*

* initSquad - מקבלת מצביע לSquad ומאתחלת את כל השדות ל0.
* addPlayerToSquad – מקבלת מצביע לSquad, מצביע לPlayerManager, התקציב הפנוי של הקבוצה, האם מכניסים את השחקן לסגל באופן ידני או רנדומלי (בוחרים אותו באופן רנדומלי ממאגר השחקנים או באופן ידני).
* choosePlayerManually – מחזירה שחקן שנבחר באופן ידני ממאגר השחקנים שאותו נכניס לסגל הקבוצה .
* choosePlayerRandomly - מחזירה שחקן שנבחר באופן רנדומלי ממאגר השחקנים שאותו נכניס לסגל הקבוצה .
* deletePlayerFromSquad – מוחקת את השחקן מסגל הקבוצה.
* printSquad – מדפיסה את כל השחקנים בקבוצה.

מאמן, Coach, שעבורו נשמור:

1. שם המאמן
2. שכר
3. מספר שנות ניסיון

*פונקציות:*

1. initCoach – מקבלת מצביע למאמן ומאתחלת את כל השדות שלו.
2. פונקציות השמה שמאתחלות את השדות המאמן לפי בחירת המשתמש (SetCoachName, SetCoachSalary, SetCoachYearsOfExperience).
3. compareCoachName – מקבלת שני מצביעים (void\* וממירה אותם לCoach\*) ומשווה לקסיקוגרפית בין שמות המאמנים.
4. printCoach – מדפיסה את כל נתוני המאמן.

היכל ביתי, Arena, שעבורו נשמור:

1. קיבולת (מספר המושבים בהיכל)
2. שם ההיכל

*פונקציות:*

* initArena – מקבלת מצביע לArena ומאתחלת את כל השדות של ההיכל.
* setArena – מקבלת מצביע לArena ומאתחלת את כל השדות של ההיכל לפי בחירת המשתמש.
* IsCapacityOk – בודקת האם קיבלת ההיכל גדולה מ1000.
* compareArenaName - מקבלת שני מצביעים (void\* וממירה אותם לArena\*) ומשווה לקסיקוגרפית בין שמות ההיכלים.
* printArena – מדפיסה את כל נתוני ההיכל.

קבוצה, Team, שעבורו נשמור:

1. מיקום הקבוצה בליגה
2. שם הקבוצה
3. קוד הקבוצה
4. סגל השחקנים (רשימה של שחקנים)
5. מאמן
6. נתוני הקבוצה
7. היכל ביתי
8. הוצאה של הקבוצה (מבחינה תקציבית)
9. תקציב פנוי
10. תקציב
11. האם הקבוצה נבחרה לליגה
12. האם הקבוצה נבחרה למשחק

*פונקציות:*

* initTeam– מקבלת מצביע לTeam ומאתחלת את כל השדות של הקבוצה.
* פונקציות השמה (setTeamName, setTeamSquad, setTeamCoach, setTeamStats, setTeamBudget).
* updateTeamAllBudgets – מעדכנת את כל תקציבי הקבוצה (הוצאות הקבוצה ותקציב פנוי) בהתאם למצב הנוכחי (שכר השחקנים והמאמן).
* פונקציות השוואה (compareTeamName, compareTeamCode, compareTeamByCoachName, compareTeamByPoints, compareTeamByRebouns, compareTeamByAssists, compareTeamByArenaName, compareTeamByBudget, compareTeamFreeByBudget) – מקבלות שני מצביעים (void\*) ממירות אותם לTeam\* ומשווות בין שתי הקבוצות על בסיס הפרמטר הרצוי.
* מספר פונקציות הדפסה – printTeamForLeagueTable (מדפיסה רק את שם הקבוצה, קוד הקבוצה והנתונים הסטטיסיטים), printAllTeamParameters (מדפיסה את כל נתוני הקבוצה).

מנהל קבוצה, TeamManager, שעבורו נשמור:

1. מערך של הקבוצות בטורניר (נשמור בתור Team\*\*)
2. מספר הקבוצות בטורניר
3. האם הליגה ממויינת

*פונקציות:*

* initTeamManager - מקבלת מצביע למנהל הקבוצות, שם של קובץ טקסט ושם של קובץ בינארי) – שואלת את המשתמש אם הוא רוצה לקרוא את המאגר מקובץ טקסט או מקובץ בינארי. במידה והטעינה שבחר לא מצליחה המשתמש יאתחל את המאגר בצורה ידנית באמצעות הפונקציה setTeamsList (מקבלת מצביע למנהל קבוצות).
* addTeam – מקבלת TeamManaegr\* וTeam\* ומוסיפה קבוצה באופן ידני (המשתמש בוחר את הפרמטרים) למאגר.
* sortTeam – מקבלת TeamManager\* ומבצעת מיון באמצעות qsort לפי הפרמטרים הבאים: שם הקבוצה, קוד הקבוצה, שם המאמן, תקציב, תקציב פנוי, שם ההיכל הביתי).
* FindTeamFromMenu - מקבלת TeamManager\* ומבצעת חיפוש באמצעות bsearch לפי הפרמטרים הבאים: שם הקבוצה, קוד הקבוצה, שם המאמן, שם ההיכל הביתי. לבסוף מדפיסה את נתוני הקבוצה שנבחרה.
* printTeamsList – מדפיסה את מאגר הקבוצות.

משחק, Match, שעבורו נשמור:

1. קבוצה ביתית
2. קבוצת חוץ
3. היכל בו המשחק נערך
4. מספר נקודות שקבוצת הבית קלעה
5. מספר נקודות שקבוצת החוץ קלעה

*פונקציות:*

* initMatch - מקבלת מצביע לMatch ומאתחלת את כל השדות של המשחק.
* SetStatsRandomTeam – מאתחלת את הנתונים הסטטיסטים של כל קבוצה במשחק ומאתחלת אותם רנדומלית באמצעות setStatsRandomly.
* SetMatchArena – מאתחלת את ההיכל שבו נערך המשחק (ההיכל של הקבוצה הביתית).
* printMatch – מדפיסה את הקבוצות ששיחקו במשחק.

ליגה, LeagueTable, שעבורו נשמור:

1. מערך של קבוצות הליגה (נשמור בתור Team\*\*).
2. מספר הקבוצות בליגה (חייב להיות כפולה של 2)

*פונקציות:*

* initLeagueTable - מקבלת מצביע לליגה, מצביע למנהל השחקנים, מצביע למנהל הקבוצות, שם של קובץ טקסט ושם של קובץ בינארי) – שואלת את המשתמש אם הוא רוצה לקרוא את הליגה מקובץ טקסט או מקובץ בינארי. במידה והטעינה שבחר לא מצליחה המשתמש יאתחל את המאגר בצורה ידנית באמצעות הפונקציה setLeagueTable (מקבלת מצביע לליגה, מצביע למנהל השחקנים ומצביע למנהל הקבוצות).
* updateStandings – ממיינת את הליגה לפי נקודות הקבוצות (באמצעות qsort) ומעדכנת כמובן את המיקום של כל קבוצה בליגה.
* EnterPlayersToTeam – מכניסה שחקנים לסגל של כל קבוצה בליגה לפי בחירת המשתמש (באופן ידני או באופן רנדומלי). כמובן שאותו שחקן לא יכול להיות בשתי קבוצות שונות.
* tradePlayers – שואלת את השחקן מאיזו קבוצה הוא רוצה להעביר שחקן ואיזה שחקן הוא רוצה להעביר ולאיזה קבוצה הוא רוצה להעביר את השחקן ובמקום איזה שחקן (אותו שחקן עובר לקבוצה המקורית). הפונקציה בודקת שהשחקנים יכולים לעשות את ה"חילוף" מבחינה תקציבית של הקבוצות. הפונקציה מעדכנת את תקציבי הקבוצות בהתאם ואת מיקומי הקבוצה בליגה (לאחר השינוי).
* LeagueRound – הפונקציה מבצעת "סיבוב ליגה". בוחרת ברנדומליות 2 קבוצות בליגה ומבצעת Match ביניהן. הפונקציה מבצעת: מספר הקבוצות בליגה חלקי 2 משחקים וכל המשחקים זה מזה. לאחר ה"סיבוב" הפונקציה מעדכנת את הליגה מחדש בהתאם לתוצאות המשחקים (בכל תחילת משחק הנתונים הסטטיסיטים של הקבוצה מתאפסים ונבחרים רנדומלית במהלך המשחק, קבוצה בייתית זכאית ליתרון של 10 נקודות).
* printPlayersListInLeague- מדפיסה את כל השחקנים בליגה ונתוניהם.
* printLeagueTable – מדפיסה את טבלת הליגה.
* printSpecificTeamInTheLeague – מדפיסה קבוצה ספציפית מהליגה שאותה המשתמש בוחר.

\*כמובן שלכל מבנה יש פונקציות free בהתאם.