תרגיל 4, מבוא לתכנות מערכות, אביב 2020

הגשה **בזוגות או ביחידים** דרך המודל
התרגיל הוא להגשה עד ליום חמישי, 13/5/2020 בשעה 23:55
בתרגיל זה תתרגלו תכנות מודולרי בשפת **C**, עבודה עם מבנים, עבודה עם קבצים וניהול זיכרון.

ליגת כדורגל

ליגת כדורגל מורכבת ממספר קבוצות (המספר יכול להשתנות בין ליגה לליגה) אשר משחקות אחת נגד השניה ומקבלות נקודות על הישגיהן. משחק כדורגל יכול להסתיים בניצחון של אחת הקבוצות (הקבוצה שמבקיעה יותר שערים) או בתיקו (אם שתי הקבוצות לא הבקיעו שערים או הבקיעו מספר שווה של שערים). ברב הליגות מקובל להעניק 3 נק׳ על ניצחון, נקודה אחת על תיקו ו-0 נקודות על הפסד.
לדוגמא: שלב הבתים בליגת האלופות האירופית מורכב מ-32 קבוצות אשר משחקות ב-8 בתים, כאשר כל בית הוא למעשה ליגה של 4 קבוצות. בית E בליגת האלופות כלל השנה את הקבוצות ליברפול, נאפולי, זלצבורג וגנק. כל קבוצה שיחקה ששה משחקים, שניים נגד כל קבוצה (פעם במגרש שלה ופעם במגרש של הקבוצה היריבה). התוצאות שהושגו הן:

Napoli	Liverpool	2-0	Napoli	Salzburg	1-1
Salzburg	Genk	6-2	Liverpool	Genk	2-1
Genk	Napoli	0-0	Liverpool	Napoli	1-1
Liverpool	Salzburg	4-3	Genk	Salzburg	1-4
Salzburg	Napoli	2-3	Napoli	Genk	4-0
Genk	Liverpool	1-4	Salzburg	Liverpool	0-2

כלומר: נאפולי ניצחה את ליברפול 2-0, זלצבורג ניצחה את גנק 6-2, גנק ונאפולי סיימו בתיקו 0-0 וכו.
בליגת האלופות מוענקות 3 נק׳ על ניצחון, נקודה על תיקו ו-0 נקודות על הפסד ולכן אחרי 6 משחקים הטבלה נראית כך:

Team	Games played	Wins	Ties	Losses	GF	GA	Points
Liverpool	6	4	1	1	13	8	13
Napoli	6	3	3	0	11	4	12
Salzburg	6	2	1	3	16	13	7
Genk	6	0	1	5	5	20	1

הטבלה היא טבלה מפורטת ומכילה, מלבד שם הקבוצה (עמודה שמאלית) ומספר הנקודות (עמודה ימנית) גם את מספר המשחקים שהקבוצה שחקה (Games played), מספר הניצחונות (Wins), תיקו (Ties) והפסדים (Losses), וכן את מספר השערים שהקבוצה הבקיעה (GF = Goals For) ומספר השערים שהקבוצה ספגה (GA = Goals Against). פרמטרים אלה יכולים להיות חשובים במידה וקיים שיוויון בנקודות בין הקבוצות.

משימות

עליכם לממש את שלושת המודולים הבאים:

קבוצה (Team)

מודול זה צריך להכיל מידע על הקבוצה, לצורך תרגיל זה המודול יכיל רק את שם הקבוצה. למעט השדה ששומר את שם הקבוצה אין להגדיר שדות נוספים.
עבור קבוצה ממשו את הפעולות הבאות:

- TeamCreate - יצירת קבוצה חדשה, הפונקציה צריכה להחזיר מצביע לאובייקט החדש שנוצר. הקלט מורכב משם הקבוצה.
- TeamDestroy – הריסת אובייקט קבוצה

משחק (Match)

מודול המייצג משחק צריך להכיל ארבעה שדות: קבוצה ראשונה, קבוצה שנייה, מס׳ השערים שהבקיעה הקבוצה הראשונה, מס׳ השערים שהבקיעה הקבוצה השנייה. שדות הקבוצות צריכות להיות מסוג מצביע למבנה קבוצה. מלבד שדות אלו אין לשמור שדות נוספים.

הפעולות שיש צורך לממש:

- MatchCreate – יצירת משחק, על סמך מצביעים לשתי הקבוצות וכן מספר השערים לכל קבוצה, כל הנתונים מועברים כארגומנטים לפונקציה. על הפונקציה להחזיר מצביע לאוביקט שנוצר.
- MatchDestroy – הריסת אובייקט מסוג Match שנוצר ע״י MatchCreate
- team_participated – פונקציה שבודקת האם קבוצה השתתפה במשחק. הפונקציה תחזיר true אם כן או false אחרת. true ו-false הם שני הערכים האפשריים של הטיפוס bool שמוגדר ב-stdbool.h.
- match_tied – פונקציה שמחזירה true במידה והמשחק הסתיים בתיקו ו-false אחרת.

בפונקציות הבאות ודאו שהקבוצה השתתפה במשחק. אם לא, הוציאו הודעת שגיאה שמציינת את מיקום הקוד בקובץ ואת שם הקובץ וכן את סיבת השגיאה.

- team_won – פונקציה המקבלת משחק ומצביע לקבוצה ומחזירה true אם הקבוצה נצחה במשחק או false אחרת.
- team_lost – פונקציה המקבלת משחק ומצביע לקבוצה ומחזירה true אם הקבוצה הפסידה במשחק או false אחרת.
- GF – פונקציה המקבלת משחק ומצביע לקבוצה ומחזירה את מספר השערים שהקבוצה הבקיעה במשחק.
- GA – פונקציה המקבלת משחק ומצביע לקבוצה ומחזירה את מספר השערים שהקבוצה ספגה במשחק.

ליגה (League)

ליגה מורכבת מקבוצות וממשחקים. מלבד השדות שבהם ישמרו המשחקים והקבוצות וכן שדות המכילים את מספר הקבוצות ומספר המשחקים אין לשמור מידע נוסף.

פעולות:

- LeagueCreate – יצירת ליגה. הפונקציה מחזירה את הכתובת של האובייקט שנוצר.
- LeagueDestroy – הריסת אובייקט ליגה שנוצר על ידי LeagueCreate.
- read_teams – טעינת הקבוצות. פונקציה זו תקבל שם של קובץ שבו קיימת רשימה של הקבוצות אותן יש ליצור. השם של כל קבוצה נתון בשורה נפרדת ומספר הקבוצות לא נתון מראש בקובץ.
  לצורך הפשטת התרגיל הניחו כי שמות הקבוצות לא מכילות רווחים; במקום רווח נשתמש בקו תחתי (׳_׳).
  ניתן להניח שקבצי השמות נתונים בפורמט נכון.
  שימו לב: פונקציה זו היא היחידה שיכולה לייצר אובייקטים של קבוצות! לכל קבוצה יהיה בדיוק אובייקט אחד בתוכנית.
- read_matches – טעינת משחקים. כל משחק נתון בשורה נפרדת והפורמט צריך להיות
  <team1> <team2> <goals_team1> <goals_team2>

  למשל:

  Genk Liverpool 1 4

  בדקו שהפורמט לו אתם מצפים אכן נכון, הוציאו הודעת שגיאה וצאו מהתוכנית אם לא.
  פונקציה זו היא היחידה שיכולה לייצר אובייקטים של משחקים, לכל משחק יהיה בדיוק אובייקט אחד.
- num_wins – מחזירה את מספר הניצחונות של קבוצה לפי מצביע לאובייקט הקבוצה
- num_ties – מחזירה את מספר המשחקים שקבוצה סיימה בתיקו לפי מצביע לאובייקט הקבוצה
- num_losses – מחזירה את מספר המשחקים שקבוצה הפסידה בהם לפי מצביע לאובייקט הקבוצה
- num_matches – מחזירה את מספר המשחקים שקבוצה שיחקה לפי מצביע לאובייקט הקבוצה
- num_GF – מחזירה את מספר השערים שקבוצה הבקיעה לפי מצביע לאובייקט הקבוצה

- num_GA – מחזירה את מספר השערים שקבוצה ספגה לפי מצביע לאובייקט הקבוצה
- print_table – הדפסת טבלה. הטבלה צריכה להיות מודפסת בפורמט מפורט כמו בדוגמא בעמוד הבא. לצורך מיון הטבלה לפי מספר הנקודות וכו כתבתי עבורכם פונקציה בשם sort_league אותה ניתן למצוא בקבצים sort_league.h ו-sort_league.c. השתמשו בקבצים האלה אך אל תשנו אותם.

תכנית הרצה

לאחר תכנון ומימוש המודולים כתבו את הפונקציה main שמקבלת את שמות הקבצים מהמשתמש ב-command line, בונה את מבני הנתונים ולבסוף מדפיסה את הטבלה. למשל:

```
~ $./league

Usage: ./league <teams-file> <matches-file>

~ $./league teams.txt matches.txt
Team                Games   Wins    Ties    Losses  GF      GA      Points
Liverpool           6       4       1       1       13      8       13
Napoli              6       3       3       0       11      4       12
Salzburg            6       2       1       3       16      13      7
Genk                6       0       1       5       5       20      1
~ $cat teams.txt
Napoli
Salzburg
Liverpool
Genk
~ $cat matches.txt
Napoli     Liverpool  2     0
Salzburg   Genk  6     2
Genk  Napoli     0     0
Liverpool  Salzburg   4     3
Salzburg   Napoli     2     3
Genk  Liverpool  1     4
Napoli     Salzburg   1     1
Liverpool  Genk  2     1
Liverpool  Napoli     1     1
Genk  Salzburg   1     4
Napoli     Genk  4     0
Salzburg   Liverpool  0     2
```

דגשים

- עבור כל אחד משלושת המודולים שהוזכרו עליכם ליצור קובץ כותר (header file) וקובץ מימוש נפרדים, h/c בהתאמה, כפי שנלמד.
- יש לבדוק תקינות קלטים לכל הפונקציות, ולהציג הודעות שגיאה בהתאם.
- במקרים של שימוש בזיכרון דינאמי, יש לוודא כי ההקצאות אכן ניתנו ע"י מערכת ההפעלה, וכן יש לנהל בקפידה את הזיכרון ולדאוג שבתום התכנית כל זיכרון דינאמי אכן משוחרר.
- במימוש פונקציות, מומלץ לעבוד עם מצביעים (כתובות) למבנים הנדרשים, הן כארגומנטים והן כערכי חזרה, כפי שנלמד.
- יש לוודא כי התכנית עוברת קומפילציה <u>ללא כל שגיאות או אזהרות כלשהן</u>, ורצה בהצלחה.
- רצוי לבצע בדיקות רבות של המודולים שמימשתם, מעבר לבדיקה שמבצעת תכנית ההרצה.

הגשה

- **הכינו קובץ בשם README.txt הכולל את שמות ותעודות הזהות של הסטודנטים המגישים**; בקובץ זה אתם גם מוזמנים לכלול הערות ותיעוד כללי לגבי המימוש שלכם.
- **הגשה אלקטרונית**: יש להגיש במערכת Moodle קובץ tar בשם **ex4.tar** או **ex4.zip** (אבל לא 7zip!) המכיל את כל קבצי הממשק, המימוש ותכנית ההרצה, וכן את הקובץ **README.txt**.

בהצלחה!