הטכניון - מכון טכנולוגי לישראל **044101 מבוא למערכות תוכנה**
הפקולטה להנדסת חשמל 8.10.07

מבחן סוף סמסטר
מועד א'

משך הבחינה: שלוש שעות

נא לרשום את השם ומספר תעודת הזהות במקום המיועד במחברת הבחינה.

עליכם לענות על כל השאלות אך ורק במחברת הבחינה.

יש להגיש רק את מחברת הבחינה (אני אינני מחברת הבחינה) !

במבחן זה 8 עמודים (לא כולל עמוד זה) ו-4 שאלות.

מומלץ לקרוא כל שאלה בעיון רב לפני שניגשים לפתור אותה.

שאלות בבחינה מסודרות לפי נושאי הלימוד ולוא דווקא לפי רמת הקושי !

כל חומר עזר מותר.

בהצלחה!!!

שאלה 1 - ADT (30 נק')

בשאלה זו נממש מימוש חלקי של מבנה נתונים מופשט (ADT) בשם Rishuy, שנועד לצורכי משרד הרישוי. טיפוס נתונים זה ישמור בתוכו רשימה עדכנית של כלי הרכב השונים הרשומים במשרד הרישוי של מדינת ישראל.

הרכב שיוכנס למבנה יכול להיות מטיפוס כלשהו. נגדיר טיפוס זה כך:

```
typedef void* PVehicle;
```

קיימים שני מאפיינים משותפים לכל הרכבים המוכנסים למבנה:

- מספר שלדה, אשר הינו חלק בלתי נפרד מהרכב והינו יחודי. מספר שלדה יוחזר ע"י פונקציה מהטיפוס:

```
typedef long (*GetShildaNum)(PVehicle);
```

- אפשרות הדפסה של תכונות הרכב. הדפסת התכונות תתבצע ע"י פונקציה מהטיפוס:

```
typedef void (*PrintVehicle)(PVehicle);
```

כמו כן, בעת הגעתו של רכב חדש למדינה, המבנה Rishuy ייצר עבורו מספר רישוי חדש יחודי (מטיפוס long), מספר שילווה את הרכב כל חייו עד שיורד מהכביש.

הפעולות שנדרשות ממבנה Rishuy הינן כדלקמן:

1. CreateRishuy – יצירת המבנה. מחזירה את המצביע למבנה הריק החדש.
2. DeleteRishuy – הורסת מבנה קיים.
3. RishuyAddVehicle – מקבלת רכב, מייצרת עבורו מספר רישוי חדש ומוסיפה רכב זה למבנה. מחזירה את מספר הרישוי שנוצר.
4. RishuyRemoveVehicle – מקבלת מספר רישוי ומוחקת רכב מהמבנה בעל מספר רישוי זה.
5. RishuyGetShildaNum – מקבלת מספר רישוי ומחזירה את מספר השלדה של הרכב במבנה.
6. RishuyPrintData – עוברת על כל כלי הרכב במבנה, מדפיסה עבור כל אחד מהרכבים את מספר הרישוי שלו ואת המאפיינים שלו (ע"י שימוש בפונקציה מתאימה מטיפוס PrintVehicle).

הערות:

1. על הפעולות 3 ו-4 יש להחזיר ערך הצלחת הפעולה מסוג

```
typedef enum {Success, Failure} Status
```

כערך החזרה (return value). אם בנוסף הפונקציה אמורה להחזיר פרמטר נוסף, יש להחזיר אותו במצביע שיתקבל כפרמטר לפונקציה.

2. לצורך המימוש של Rishuy, השתמשו ב- ADT של רשימה כללית שמנשקו נתון בהמשך.

3. לצורך יצור מספר רישוי חדש, הניחו כי קיימת פונקציה גלובלית הבאה:

```
static long GenNewRishuyNum();
```

להלן תכנית שימוש לדוגמא, אשר משתמשת במבנה Rishuy:

```
typedef struct _car {
      long shilda_num;
      char model_name[255];
} Car, *PCar;
```

```
long GetCarShilda(PVehicle pv) {
      PCar pc = (PCar) pv;
      return pc->shilda_num;
}

void PrintCar(PVehicle pv)
{
      PCar pc = (PCar) pv;

      printf("Car details: shilda number = %d, model name = %s",
             pc->shilda_num, pc->model_name);

}

int main()
{
      /* create an empty Rishuy */
      PRishuy pr;
      pr = CreateRishuy();

      /* create new car */
      PCar pc;
      pc = (PCar) malloc(sizeof(Car));
      pc->shilda_num = 10101010;
      strcpy(pc->model_name, "Toyota");

      /* add this car */
      int RishuyNum;
      Status stat;
      stat = RishuyAddVehicle(pr, pc, GetCarShilda, PrintCar,
                              &RishuyNum);
      if (stat == Failure)
            printf("Failed to add a car ! \n");

      /* free memory */
      free(pc);
      DeleteRishuy(pr);

      return 0;
}
```

עליכם לענות על שלושת הסעיפים הבאים:

א) (8 נק')

כתבו את קובץ המנשק של Rishuy ADT. הקפידו לרשום את כל ההגדרות הדרושות לקובץ זה.

ב) (5 נק')

ממשו את struct _Rishuy בקובץ המימוש.

ג) (17 נק')

ממשו את הפונקציות הבאות:

CreateRishuy, RishuyAddVehicle, RishuyPrintData.

הקפידו לרשום את כל ההגדרות והפונקציות הנוספות, הדרושות למימוש שלושת פונקציות מנשק אלה.

הדרכה: על מנת להשתמש במבנה של רשימה כללית, הגדירו מבנה הבא:

```
typedef struct _item {
      ...
} Item, *PItem;
```

והכניסו לרשימה איברים מטיפוס PItem.

להלן מנשק של רשימה כללית:

```
typedef struct _list* PList;
typedef void* PKey;
typedef void* PElem;
typedef enum {Success, Failure} Status;

/*user-defined functions:*/
typedef PKey (*GetKey)(PElem); /*returns Key from the Elem*/
typedef Status (*CompareFunc)(PKey, PKey); /*compares between two
             Keys and returns Success if they match*/

/*interface functions:*/
PList CreateList(GetKey, CompareFunc); /*creates new list*/
void  DeleteList(PList); /*deletes the list*/
Status ListAddElem(PList, PElem); /*adds new element to list*/
PElem  ListFindElem(PList, PKey); /*returns element given its key*/
Status ListRemoveElem(PList, PElem); /*removes element from list*/
PElem  ListGetFirst(PList); /*gets first item in the list*/
PElem  ListGetNext(PList); /*gets next item in the list*/
```

במימוש זה של רשימה, המפתח (Key) שיכול להיות כללי, הינו חלק מהאיבר הכללי (Elem). החיפוש ברשימה נעשה ע"י הפונקציות ListFindElem, ListGetFirst ו-ListGetNext , בדומה לרשימה שמימשתם בתרגיל 5.

שאלה 2 – מבנה נתונים (22 נק')

בשאלה זו אין לכתוב קוד. יש לכתוב הסברים קצרים, ברורים ומדויקים.
הסברים קצרים ברורים ומדויקים ניתן לכתוב בעברית, אנגלית, pseudo-code או ע"י תרשימי זרימה.

בשאלה זו נעסוק בערימת מקסימום הממומשת כעץ בינארי כמעט מלא המסודר לפי חוק הערימה. העץ הינו מוקצה דינאמית וכל צומת מכילה (פרט לנתונים) מצביע לאב ולשני הבנים; מצביעים אלה יקראו father, left,right, בהתאמה.
בכל סעיף ניתן להניח כי מימשתם את הסעיפים הקודמים בהצלחה ולהשתמש בהם.

א. (6 נק')
נתון מצביע לעלה כלשהו בעץ (ומצביע לראש העץ), כתבו אלגוריתם המוחק עלה זה מהערימה בסיבוכיות O(log n) כאשר n הינו מספר האיברים בערימה. בסוף האלגוריתם על הערימה להיות ערימה תקנית. הסבירו מדוע האלגוריתם נכון ועומד בדרישות הסיבוכיות.

ב. (6 נק')
נתון מצביע לצומת כלשהו בעץ (ומצביע לראש העץ), כתבו אלגוריתם המוחק צומת זה מהערימה בסיבוכיות O(log n) כאשר n הינו מספר האיברים בערימה. בסוף האלגוריתם על הערימה להיות ערימה תקנית. הסבירו מדוע האלגוריתם נכון ועומד בדרישות הסיבוכיות.
רמז: אם תצליחו להפוך את הצומת לעלה, סיימתם.

במועדון הפופולרי "חרסנוס" ($\chi\alpha\rho\sigma\alpha\nu o\sigma$) שבכרתים נוצר בכל ערב תור ארוך של אנשים בכניסה. מדיניות בעל המועדון הינה להכניס בכל רגע נתון את האדם היפה ביותר שעומד בתור עד הרגע בו הראשונים בתור הופכים אלימים, ואז מוכנס האדם שנמצא בתור הכי הרבה זמן. הסלקטור במועדון מציע לנהל את התור ע"י מחשב. בישיבה עם בעל המועדון מוחלט כי המנשק הדרוש מורכב מ-4 פונקציות:

1. initialize – מאתחלת מבנה נתונים ריק, סיבוכיות נדרשת O(1).
2. insertPerson – מקבלת ת"ז וציון יופי (יפה יותר מקבל ציון גבוה יותר) ומכניסה את האדם למבנה. סיבוכיות נדרשת O(log n) כאשר n הינו מספר האנשים במבנה.
3. getBestLooking – מחזירה את ת"ז של האדם היפה ביותר בתור ומוציאה אותו מהמבנה. סיבוכיות נדרשת O(log n) כאשר n הינו מספר האנשים במבנה.
4. getFirstInLine - מחזירה את ת"ז של האדם הותיק ביותר בתור ומוציאה אותו מהמבנה. סיבוכיות נדרשת O(log n) כאשר n הינו מספר האנשים במבנה.

(אין צורך בפונקצית שחרור זיכרון שכן בסוף כל ערב התור ריק).

הסלקטור מציע מבנה נתונים המורכב מערימת מקסימום הממוינת לפי ציון היופי וממומשת כעץ בינארי כמעט שלם המוקצה דינאמית וכן תור FIFO הממומש כרשימה מקושרת חד-כיוונית בעלת מצביע לראש וזנב ע"מ לאפשר הכנסה והוצאה מהירים.
הערה: ניתן להניח שכל ציוני היופי שונים זה מזה.

ג. (4 נק')
האם ניתן לממש את הפונקציות לעיל בסיבוכיות הרצויה ע"י שימוש במבנה הנתונים המוצע? אם כן הסברו כיצד, אם לא מהי הסיבוכיות המינימאלית הנדרשת ע"מ לממש את 4 הפונקציות לעיל ע"י שימוש במבנה המוצע? הסברו בקצרה.

ד. (6 נק')
הצעו שינויים ו/או תוספות קלות במבנה כך שניתן יהיה לממש את הפונקציות בסיבוכיות הנדרשת. הסברו את המימוש והראו כי הוא אומר בדרישות הסיבוכיות.
רמז: שימו לב לסעיפים א' ו- ב'.

שאלה 3 – C++ (32 נק')

שאלה זו דנה בדרכי טיפול במערכים דו-מימדיים בשפת C++.

חלק א: (8 נקודות)

התבוננו בקטע קוד (חלקי) הבא:

```
class Complex { /* … */ };

void someFunction(Complex& cmp);

void manipulateMatrix(int nrows, int ncols)
{
      /* allocate a matrix of Complex, of size (nrows x ncols) */
      /* (1) */

      /* manipulate matrix */
      for (int i =0; i < nrows; ++i)
            for (int j =0; j < ncols; ++j)
                  someFunction( matrix[i*ncols + j] );


      /* deallocate a matrix */
      /* (2) */
}
```

א. (1 נק') השלימו את הקטע המקצה את המטריצה, המסומן ע"י /* (1) */.

ב. (1 נק') השלימו את הקטע המשחרר את המטריצה, המסומן ע"י /* (2) */.

ג. (2 נק') הסבירו מהן הדרישות מהמחלקה Complex על מנת שקטע קוד זה יתקמפל.

ד. (4 נק') מהי בעיתיות בקטע קוד זה ? רמז: חישבו מה קורה כאשר someFunction זורקת חריגה. פתרו את הבעיה והסבירו את הפתרון.

חלק ב: (4 נקודות)

נניח כעת כי ברצוננו להשתמש במערך דו-מימדי לא מלבני. זהו מערך שמספר האיברים בשורות אינו קבוע. התבוננו בקטע קוד (חלקי) הבא:

```
class Complex { /* … */ };

void someFunction(Complex& cmp);

void manipulateMatrix(int nrows, int ncols[])
{
      /* allocate a non-rectangular matrix of Complex */
      /* (1) */

      /* manipulate matrix */
      for (int i =0; i < nrows; ++i)
            for (int j =0; j < ncols[i]; ++j)
                  someFunction( matrix[i][j] );


      /* deallocate a matrix */
      /* (2) */
}
```

ה. (4 נק') חזרו על סעיפים א' ו-ב' במקרה זה. שימו לב כי $\text{ncols}[i]$ מציין את מספר האיברים בשורה i.

חלק ג: (20 נקודות)

כפי ששמתם לב, הקוד בחלקים הקודמים הינו בעייתי ויכול לגרום לשגיאות, והסיבה היא השימוש הישיר במצביעים. נרצה לפתור את הבעיה ע"י "הסתרה" של השימוש הישיר במצביעים בתוך מחלקה עם מנשק פשוט ובטוח. הניחו שוב כי אנו מטפלים במטריצות מלבניות (כמו בחלק א') והתבוננו בקטע קוד חדש:

```
class Complex { /* … */ };
class BadSize { };
class BoundsViolation { };

void someFunction(Complex& cmp);
void someOtherFunction(CompMatrix m);

void manipulateMatrix(int nrows, int ncols)
{
      /* define a matrices of Complex, of size (nrows x ncols) */
      try {
            CompMatrix matrix(nrows, ncols);
      }
      catch (BadSize e) {
            cout << "Bad size ! << endl;
            return;
      }


      /* manipulate matrices */
      try {
            for (int i =0; i < nrows; ++i)
                  for (int j =0; j < ncols; ++j) {
                        someFunction( matrix(i, j) );
                        // assume that Complex has this constructor
                        matrix(i, j) = Complex(0, 0);
                  }
      }
      catch (BoundsViolation e) {
            cout << "Bounds violation ! << endl;
            return;
      }
      someOtherFunction(matrix);
}
```

ו. (5 נק') הגדירו את המנשק של המחלקה CompMatrix, כפי שנדרש מהקטע הקוד הנ"ל.

ז. (10 נק') ממשו את המחלקה CompMatrix. הקפידו לממש גם את קטעי הקוד הזורקים חריגות.

ח. (5 נק') התבוננו בקטע הבא:

```
int main()
{
      const CompMatrix m(2, 2);
      cout << m(1, 1) << endl;
      return 0;
}
```

האם קטע זה יתקמפל ? אם כן, הסבירו מדוע. אם לא, הסבירו את הבעיה והציעו תיקון למנשק שכתבתם בסעיף ו' הפותר את הבעיה.

שאלה 4 - CSH (16 נק')

הפקודה who מראה מי כרגע logged-in במחשב (וכן פרטים נוספים עליו). לדוגמא:

```
% who

sharon     pts/13        Aug 15  13:02   (eem-sharon.eed.ef.technion.ac.il:0.0)
adam       pts/16        Jul 26  10:27   (ee:1.0)
michal     pts/2         Aug  2  09:15   (eem-michal.eed.ef.technion.ac.il:0)
merav      pts/19        Aug 16  10:05   (ee)
michal     pts/20        Aug  2  09:28   (eem-michal.eed.ef.technion.ac.il)
sharon     pts/44        Jul 29  10:07   (m-sharon.eed.ef.technion.ac.il:0)
Sharon     pts/31        Aug  5  07:58   (132.68.59.164)
gil        pts/42        Aug 12  17:57   (vsm-pc-100.vsm.technion.ac.il)
yoav       pts/3         Aug 16  12:33   (yoav-02.technion.ac.il)
```

שים לב שחלק מהמשתמשים מופיעים מספר פעמים מאחר ועשו login יותר מפעם אחת.

א. (2 נק')

רשום script בשם howmany, באורך שורה אחת (לא כולל את שורת הקריאה ל-/bin/csh), המקבל שם משתמש כפרמטר ומדפיס כמה פעמים עשה המשתמש הזה login.

ב. (4 נק')

רשום script בשם when באורך 2 שורות, המקבל כפרמטר שם משתמש, ומדפיס את התאריך בו עשה login ואת משך הזמן מאז היה המשתמש פעיל לאחרונה. עליך להתייחס לשורה הראשונה בפלט הפקודה who, בה מופיע משתמש זה.

למשל – עבור המשתמשת מיכל הוא ידפיס :

```
michal   Aug  2   09:15
```

ג. (10 נק')

רשום script בשם count המחשב את סכום המספרים המופיעים בעמודה השנייה בפלט הפקודה who.

בדוגמא למעלה הוא יחשב וידפיס 190.

(13+16+2+19…).

<u>הנחיה:</u> חלק לשני scripts, הראשון באורך שורה אחת והשני כ-10 שורות.

<u>הערה:</u>

בכל הסעיפים, אין צורך לבדוק את מספר או נכונות הפרמטרים ואין להשתמש בקבצי ביניים.