הטכניון - מכון טכנולוגי לישראל
הפקולטה להנדסת חשמל

044101 מבוא למערכות תוכנה
11.3.07

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

מומלץ לקרוא כל שאלה בעיון רב לפני שניגשים לפתור אותה.

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

כל חומר עזר מותר.

בהצלחה!!!

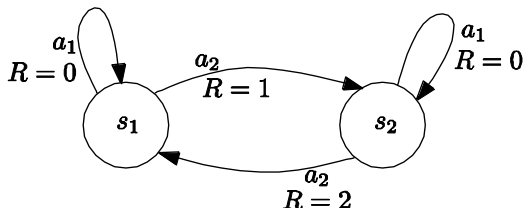
שאלה 1 - ADT (30 נק')

מכונת מצבים סופית מורכבת מהמרכיבים הבאים:

- קבוצת מצבים סופית $S = \{s_1, s_2, ..., s_n\}$.
- קבוצת פעולות סופית $A = \{a_1, a_2, ..., a_m\}$. זהו אוסף הפעולות שניתן לבצע בכל מצב.
- פונקציית מעבר $T : S \times A \to S$. בהינתן מצב נוכחי והפעולה שבוצעה במצב זה, פונקציה זו תחזיר את המצב הבא.
- פונקציית רווח $R : S \times A \to \mathbb{R}$. כלומר, $R(s,a)$ זהו מספר ממשי המציין את הרווח בביצוע הפעולה $a \in A$ במצב $s \in S$.

מדיניות (policy) הינה פונקציה $P : S \to A$, המגדירה את הפעולה שיש לבצע בכל מצב.

דוגמא:



כאן:

$$A = \{a_1, a_2\}\ , S = \{s_1, s_2\}$$
$$T(s_1, a_1) = s_1,\ T(s_1, a_2) = s_2,\ T(s_2, a_1) = s_2,\ T(s_2, a_2) = s_1$$
$$R(s_1, a_1) = 0,\ R(s_1, a_2) = 1,\ R(s_2, a_1) = 0,\ R(s_2, a_2) = 2$$

המדיניות יכולה להיות למשל: $P(s_1) = P(s_2) = a_2$, כלומר מדיניות של מעבר ממצב נוכחי למצב אחר.

מטרתנו בשאלה זו לכתוב ADT בשם FSM אשר יבצע סימולציה למכונות מסוג הנ"ל, ויתמוך בפעולות הבאות:

- FSMCreate - יצירת מכונה חדשה. הפונקציה תקבל את קבוצת המצבים S, קבוצת הפעולות A, פונקציית מעבר T ופונקציית הרווח R ותחזיר את המבנה המאותחל, כאשר המצב הנוכחי מאותחל למצב הראשון בקבוצת המצבים.
- FSMSetCurrentState - הפונקציה תקבל את המכונה ואת המצב s ותקבע את המצב הנוכחי של המכונה ל- s.
- FSMGetCurrentState - הפונקציה תקבל את המכונה ותחזיר את המצב הנוכחי.
- FSMStartSimulation - הפונקציה תקבל את המכונה, את הפונקצית המדיניות P ואת מספר צעדי סימולציה K. הפונקציה תבצע סימולציה למכונה נתונה ע"פ מדיניות P החל מהמצב הנוכחי במשך K צעדים ותחזיר את הרווח הכולל שהתקבל במהלך הסימולציה.
- FSMContinueSimulation - הפונקציה תקבל את המכונה, את הפונקצית המדיניות P ואת מספר צעדי סימולציה K. הפונקציה תמשיך את הסימולציה מהמצב הנוכחי, ותחזיר את הרווח הכולל (המצטבר).
- FSMBestAction - הפונקציה תקבל את המכונה, ותחזיר את הפעולה המיטבית מהמצב הנוכחי (כלומר, הפעולה המניבה את הרווח המקסימלי במצב נוכחי).
- FSMDestroy - משחררת את הזיכרון שתופסת המכונה.

הערות:

1. המצבים והפעולות יכולות להיות מסוג כלשהו.
2. על כל הפונקציות הנ"ל (פרט ל- FSMDestroy) להחזיר ערך הצלחת הפעולה מסוג

```
typedef enum {SUCCESS, FAILURE} Result
```

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

3. לצורך המימוש, השתמשו ב- ADT של קבוצה כללית (לנוחותכם, מנשקו נתון בדף הבא).

א. רשמו את קובץ המנשק של FSM . הקפידו לרשום את כל ההגדרות הדרושות לקובץ זה.

ב. רשמו את הגדרת המבנה struct _fsm {...} .

ג. ממשו את הפונקציות FSMContinueSimulation ו- FSMBestAction .

```
#ifndef SET_H
#define SET_H

/* Definitions and types */
/* Set pointer definition */
typedef struct _set *pSet;
/* general element definition */
typedef void* pElement;
/* useful types */
typedef enum {FALSE, TRUE} bool;
typedef enum {FAILURE, SUCCESS} Result;

/* Function Types */
/* Compare function - Returns 0 iff elements are equal */
/* Returns < 0 iff first < second */
/* Returns > 0 iff first > second */
typedef int (*cmp)(pElement, pElement);
/* Copy function - creates a copy of the element */
typedef pElement (*cpy)(pElement);
/* Delete Function - frees the memory of the element */
typedef void (*del)(pElement);

/* Interface functions */
/* Create an empty set*/
pSet SetCreate(cmp, cpy, del);
/* Add an element to the set */
/* Returns SUCCESS, if after the usage given element is in the set.
   Returns FAILURE in case of some memory failure */
Result SetAdd(pSet, pElement);
/* Remove an Element from the set */
/* Returns FAILURE if the element does not exist
   or in some memory failure, otherwise - SUCCESS */
Result SetRemove(pSet, pElement);
/* Check if an element exists in the set */
/* Return TRUE if element is in set, else (and in case of some failure)
   returns FALSE */
bool SetIsIn(pSet, pElement);
/* Returns the number of elements in the set */
/* Returns -1 in case of failure */
int SetSize(pSet);
/* Returns the first element in the set, NULL if empty*/
pElement SetFirst(pSet);
/* Returns the next element in the set, NULL if reached the end of the set*/
pElement SetNext(pSet);
/* Free all memory used by the set including its elements */
void SetDestroy(pSet);

#endif
```

שאלה 2 – מבנה נתונים (20 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.

ברצוננו לתכנן בסיס נתונים עבור משרד התעבורה, שיעזור בניהול מידע על עבריינים תנועה, בהתאם לשיטת הניקוד החדשה.

השיטה:
בהתאם לחומרת העבירה, הנהג מקבל לחובתו בין 2 ל-12 נקודות על עבירה שביצע. הנקודות הנצברות הן תמיד מספרים **שלמים וזוגיים**.
לכל נהג ישנה תקופת צבירה, שיכולה להיות שנתיים או ארבע שנים.
תקופת הצבירה ההתחלתית הינה של שנתיים.
אם סך כל הנקודות בתקופת צבירה (כלשהיא) עולה על 11 נקודות, על הנהג לבצע קורס בסיסי בנהיגה מונעת.
אם סך כל הנקודות בתקופת הצבירה עולה על 21 נקודות, תקופת הצבירה עולה לארבע שנים ונשארת כך לכל אורך חיי הנהג.
אם סך כל הנקודות בתקופת הצבירה עולה על 23 נקודות, הנהג מזומן לקורס שני, מתקדם, בנהיגה מונעת.
כשהעבריין הסדרתי מגיע ל-36 נקודות בתקופת הצבירה, יישלל רישיונו לתקופה של 3 חודשים (כאן נפסיק למרות שהחוק ממשיך ומחמיר עם עבריינים סדרתיים).

שימו לב:

- תקופת הצבירה הנוכחית של נהג אינה סטטית אלא חלון-זמן נע באורך שנתיים או ארבע שנים, בהתאם לניקוד שצבר.
- מספר החודשים בתקופת הצבירה אינו קבוע לצורכי חישוב הסיבוכיות.

א. (8 נק')
תכננו מבנה שייקרא Driver, שינהל את צבירת הנקודות של נהג ספציפי. בנוסף, המבנה ישמור את מס' ת.ז. של הנהג.
הפעולות המוגדרות על המבנה ודרישות הסיבוכיות הן כדלקמן:

שם הפעולה	פרמטרים	סיבוכיות נדרשת	הסבר
Init	ת.ז. של הנהג	O(1)	יצירת מבנה חדש עבור נהג
AddPointsThisMonth	מצביע ל-Driver, מספר נקודות שנצברו בחודש המסוים	O(1)	הוספת נקודות שנצברו ע"י הנהג בחודש הנוכחי. הניחו כי פונקציה זו נקראת פעם בחודש, בסוף כל חודש.
GetSum	מצביע ל-Driver	O(1)	מחזירה את מספר הנקודות בתקופת הצבירה הנוכחית.
GetId	מצביע ל-Driver	O(1)	מחזירה את תעודת הזהות של הנהג.

ב. (12 נק')
כעת נתכנן מבנה חדש, שירכז את המידע על עבירות כל נהגי ישראל, תוך שימוש במבנה Driver שבנינו בסעיף א'. מוצא לממש את המבנה שייקרה Transport בעזרת עץ חיפוש בינארי ממוין, Tree, שניתן לתמוך בו ע"י מבנה/י עזר. כל צומת בעץ (Node) כולל מצביע למבנה Driver (העץ ממומש כמבנה non-intrusive).
יש להסביר באיזה מבנה/מבנים השתמשתם וכיצד מימשתם את הפעולות.

הפעולות המוגדרות על המבנה ודרישות הסיבוכיות הן כדלקמן:

בהינתן n מספר הנהגים:

שם הפעולה	**פרמטרים**	**סיבוכיות נדרשת**	**הסבר**
Init	מערך n מצביעים למבני Driver	O(nlogn) בממוצע	יצירת מבנה חדש עבור כלל הנהגים
AddPointsThisMonth	מצביע ל- Transport, מצביע ל- Node הספיציפי, מספר נקודות שנצברו בחודש המסוים	O(logn) בממוצע	הוספת נקודות שנצברו ע"י הנהג (שנמצא בתוך ה-Node) בחודש הנוכחי. הניחו כי פונקציה זו נקראת פעם בחודש, בסוף כל חודש.
GetCourseA	מצביע ל- Transport	O(m) m – מספר הנהגים שהפונקציה מדפיסה.	מדפיסה רשימה של מס' ת.ז. של כל הנהגים שעליהם לעבור קורס בסיס, ממוינת לפי מספר הנק' שצברו מהקטן לגדול (כלומר כל הנהגים שלחובתם יותר מ-11 נק' ופחות מ-24 נק')
GetCourseB	מצביע ל- Transport	O(m) m – מספר הנהגים שהפונקציה מדפיסה.	מדפיסה רשימה של מס' ת.ז. של כל הנהגים שעליהם לעבור קורס מתקדם, ממוינת לפי מספר הנק' שצברו מהקטן לגדול (כלומר כל הנהגים שלחובתם יותר מ-23 נק' ופחות מ-36 נק')
GetCancelLicense	מצביע ל- Transport	O(m) m – מספר הנהגים שהפונקציה מדפיסה.	מדפיסה את מס' ת.ז. של כל הנהגים שרישיונם צריך להיפסל, ממוינת לפי מספר הנק' שצברו מהקטן לגדול (כלומר כל הנהגים שלחובתם יותר מ-35 נק')

תזכורת: גובה ממוצע של עץ חיפוש בינארי הינו O(logn).

רמזים:

1. זכרו כי בעץ בינארי ממוין מתקיים הבן השמאלי קטן **<u>או שווה</u>** לאב.

2. עבור עץ בינארי ממוין של מספרים שלמים, חישבו איך נוכל ב-O(m) להדפיס לפי סדר עולה את כל המספרים בתחום [12,23], כאשר m הוא מספר הנהגים שהפונקציה מדפיסה.

שאלה 3 – C++ (30 נק')

א. נתון הקובץ A.H בו מוגדרת המחלקה A הבאה:

```
class A{
public:
      void set_n(const int n){_n = n; return;}
      int get_n(void) const {return _n;}
private:
      A(void):_n(0){}
      int _n;
};
```

האם ניתן ליצור אובייקטי ם מהמחלקה A? אם כן הראה כיצד ואם לא הסבר מדוע. (3 נק')

ב. נתונה התוכנית הבאה:

```
#include <iostream>
#include "A.H"

using std::cout;
int main{
      A *pa,*pb;
      pa = A::get_new_A();
      pb = A::get_new_A();
      pa->set_n(3);
      pb->set_n(5);
      cout << "a::_n = "<<pa->get_n()<<", b::_n = " <<pb->get_n()<<endl;
      delete pa;
      delete pb;
      return 0;
}
```

הוסף את ההצהרה והמימוש של הפונקציה get_new_A כך התכנית תפעל ותדפיס את הפלט

a::_n = 3, b::_n = 5;

יש לכתוב **במפורש** היכן הפונקציה get_new_A מוגדרת (5 נק').

רמז: חשוב כיצד יכולה הפונקציה get_new_A ליצור אובייקט חדש ולהחזיר אותו.

ג. בסעיף ב' מימשנו פונקציה היוצרת אובייקט חדש ממחלקה A, אך זו לא הדרך היחידה ליצור אובייקט ממחלקה A.

- שנו את השורה

*pb = A::get_new_A();

כך ש pb יצביע על אובייקט חדש מטיפוס A ללא שימוש ב- get_new_A. (6 נק')
רמז: חשבו איזה פונקציות יש למחלקה A (לא בהכרח אלו שכתובות במפורש)

- איזו תוספת יש להוסיף להגדרת המחלקה A כך שניתן יהיה ליצור אובייקטים חדשים אך ורק דרך הפונקציה get_new_A יש לכתוב במפורש את הקוד הנדרש וכן היכן הוא ממוקם. (2 נק').

כעת יש בידינו מחלקה בה ניתן ליצור אובייקט חדש אך ורק ע"י גישה לפונקציה נתונה. כלומר, אנו שולטים על האפשרות ליצור אובייקטים חדשים של המחלקה וניתן למשל להחליט שאסור ליצור יותר אובייקטים (זאת לא ניתן לעשות ע"י כתיבת constructor מתאים שכן constructor **תמיד** יוצר אובייקט חדש) . באפליקציות רבות נרצה כי לא ייוצר יותר מאובייקט יחיד ממחלקה מסוימת (לדוגמא: error_handler המטפל בכל השגיאות של תוכנית גדולה או logger המעביר הודעות ל-log file).

ד. שנו את המחלקה A כך שבפעם הראשונה בה נקראת הפונקציה get_new_A ייוצר אובייקט חדש מטיפוס A ויוחזר מצביע אליו ואילו בפעמים הבאות שתקרא הפונקציה get_new_A יוחזר המצביע אל האובייקט שנוצר בפעם הראשונה. יש לכתוב את הקובץ A.H החדש ואת הקובץ A.C (אם יש בו צורך). (10 נק')
שימו לב לנקודות הבאות

- כיצד המחלקה תדע האם כבר הוקצה אובייקט כלשהוא ? האם זו תכונה של אובייקט ספציפי או של כל המחלקה ?
- שימו לב לשחרור הזיכרון. מה יקרה אם נשתמש ב-delete כמו בסעיף ב' ? לכן יש למנוע מהמשתמש לעשות delete ולספק פונקציה שתשחרר את הזיכרון בצורה נכונה (נקרא לה delete_A). ניתן להניח שהמשתמש יפעיל את הפונקציה delete_A עבור כל עותק של האובייקט שנמצא ברשותו (זאת משום שעותקים שונים יהיו אצל חלקים שונים בתוכנה ולא ניתן לדעת מי יסיים אחרון).

ה. השלימו את התוכנית הבאה ע"מ שתעבור בצורה נכונה, וכתבו מהו הפלט.(4 נק')

```
#include <iostream>
#include "A.H"
using std::cout;
int main{
      A *pa,*pb;
      //allocate pa and pb
      …
      pa->set_n(3);
      pb->set_n(5);
      cout << "a::_n = "<<pa->get_n()<<", b::_n = " <<pb->get_n()<<endl;
      // free memory
      …
      return 0;
}
```

<u>הערה:</u> המחלקה שיצרתם היא תבנית תכנון (design pattern) סטנדרטית הידועה בשם Singleton.

שאלה 4 - CSH (20 נק')

א. (5 נקודות)

הפקודה "**finger** user-name" רושמת לפלט הסטנדרטי את הפרטים של המשתמש user-name העובד כרגע במערכת. לדוגמא:

```
% finger ronit

Login name: ronit                In real life: Ronit Tirosh
Office:  EE1155,   4655              Home phone:  8222333
….   (more details)
```

רשום script קצר (לא יותר מ-4 שורות) בשם phone המקבל את שם המשתמש (בדוגמא - ronit) ומדפיס את המשפט:

Home phone: xxxxxx

כאשר xxxx הינו מספר הטלפון בבית. בדוגמא זו יודפס:

Home phone: 8222333

הנח כי המשתמש נמצא כרגע במערכת.

ב. (4 נקודות)

הפקודה "finger" ללא פרמטרים מדפיסה את שמות ופרטים נוספים של המשתמשים הנמצאים כרגע logged-on במחשב.
לדוגמא:

```
% finger

Login     Name            TTY       Idle   When          Where
idit      Cohen Idit      pts/75    20:    Sun 12:57     ees-cohen.eed.ef.te
dubi      Dov Manor       pts/57    1d     Thu 14:55     eem.eed.ef.te
jacob     Jacob Levy      pts/10    6d     Mon 14:22     star.technion.ac.il
shoham    Nahum Soham     pts/88    11     Wed 10:50     132.68.48.156
```

רשום script קצר (עד 3 שורות) בשם tty, המקבל כפרמטרים שם ושם משפחה של המשתמש, משתמש בפקודה "finger" ומדפיס את ה- TTY ממנו עשה המשתמש login. למשל בדוגמא למעלה, אם נרשום

tty Dov Manor

נקבל כפלט:

Dov Manor pts/57

(זאת בהנחה שדב נמצא כרגע במערכת)

ג. (5 נקודות) חזור על הסעיף הקודם כאשר tty מקבל כפרמטרים מספר כלשהו של זוגות "שם שם-משפחה" למשל:

tty Nahum Shoham Idit Cohen Jacob Levy

ידפיס :

Nahum Soham pts/88
Cohen Idit pts/75
Jacob Lev pts/10

(זאת בהנחה שאף אחד מהשלושה לא עשה logout). בסעיף זה ניתן להגדיל את התוכנית ב- 5 שורות נוספות לכל היותר.

ד. (6 נקודות) רשום script קצר בשם print-match המקבל כפרמטר ראשון דגל ואחריו מספר לא ידוע מראש של מחרוזות.
אם הדגל הינו "print-" עליו לעבור על כל הקבצים בספריה הנוכחית ולהדפיס את שמות הקבצים בהם מופיעה אחת מהמחרוזות הנתונות. בכל מקרה אחר עליו לצאת מהתוכנית.

לדוגמא:

print-match -print ab cd ef

ידפיס את שמות כל הקבצים אשר בשמם מופיע ab, cd או ef.
(12-15 שורות קצרות).

הערות כלליות:

בכל הסעיפים אין צורך לבדוק את תקינות או מספר הפרמטרים.
אין להשתמש בקבצי בינים.
אין צורך ביותר מ- script אחד <u>קצר</u> בכל סעיף.