מספר זהות:

סמסטר א' תשע"ב
מועד: א' 10/2/2012
משך הבחינה: 3½ שעות
חומר עזר: כל חומר עזר כתוב מותר

בחינה בקורס: תכנות מכוון עצמים ושפת C++

מרצים: **אמיר קירש, ד"ר אלון שקלר**

מדבקת ברקוד

הנחיות כלליות לבחינה:

- שימו לב, השאלון מודפס משני צדדיו.
- המבחן מורכב משני חלקים:
  חלק א' כולל 7 שאלות אמריקאיות. משקל כל שאלה 7 נקודות, סה"כ: 49 נק'.
  חלק ב' כולל שאלת תכנות שמשקלה הכולל 51 נק'.
- חובה לתעד בשאלת התכנות כל פעולה לא ברורה שנעשית.
- בשאלות האמריקאיות יש לסמן תשובה אחת לכל שאלה בטבלה המצורפת. במידה ומספר תשובות נראות נכונות יש לסמן את התשובה הנכונה ביותר.
- בסיום המבחן יש לרשום מס' ת.ז. על גבי טופס המבחן, לוודא שטבלת התשובות האמריקאיות נמצאת יחד עם טופס המבחן, ולהגישם בתוך מחברת הבחינה.
- המבחן הינו עם חומר פתוח. כל חומר עזר מותר למעט מכשירים אלקטרוניים למיניהם. אין להעביר חומר עזר בין תלמידים במהלך המבחן.
- נא לכתוב בכתב קריא ולא מחובר.

בהצלחה!

טבלת תשובות לחלק האמריקאי

חובה לספק הסבר עבור תשובה ו׳

	א	ב	ג	ד	ה	ו
שאלה 1						
שאלה 2						
שאלה 3						
שאלה 4						
שאלה 5						
שאלה 6						
שאלה 7						

הסברים לתשובות

חובה לספק הסבר במידה ונבחרה תשובה ו׳ (אף תשובה אינה נכונה).

מותר לצרף הסבר גם עבור תשובות אחרות. אמנם **רק** תשובה נכונה תזכה בניקוד עבור כל שאלה, אולם ניתן יהיה להסתמך על ההסבר במסגרת ערעור, אם יידרש. מומלץ לצרף הסבר לתשובה אמריקאית במיוחד במקרים בהם נראה לך שתשובתך דורשת הסבר או נימוק.

שאלה 1 :

שאלה 2 :

שאלה 3 :

שאלה 4 :

שאלה 5 :

שאלה 6 :

שאלה 7 :

חלק א' – שאלות אמריקאיות (49 נק' – 7 נק' לכל שאלה)

שאלות 1-4 מתייחסות לקטע הקוד הבא:

```
1.  template<class T>class A
2.  {
3.     class A_Impl
4.     {
5.        T var;
6.        int version;
7.     public:
8.        A_Impl(T var1): var(var1), version(1){}
9.        A_Impl(const A_Impl& a): var(a.var), version(1){}
10.       const A_Impl& operator=(const A_Impl& a)
11.          {var = a.var; ++version; return *this;}
12.       operator T()const{return var;}
13.       friend class A<T>;
14.    };
15.
16.    A_Impl a;
17.
18. public:
19.       A(T var):a(var){}
20.       A_Impl& get(){return a;}
21.       int version(){return a.version;}
22. };
23.
24. int main()
25. {
26.    A<int> a1(5);
27.    cout  << "a1 = " << a1.get() << endl
28.          << ", version = " << a1.version() << endl;
29.
30.    a1.get() = 7;
31.    cout  << "a1 = " << a1.get() << endl
32.          << ", version = " << a1.version() << endl;
33.
34.    A<int> a2 = a1;
35.    a1 = 13;
36.    cout  << "a2 = " << a2.get() << endl
37.          << ", version = " << a2.version() << endl;
38.    cout  << "a1 = " << a1.get() << endl
39.          << ", version = " << a1.version() << endl;
40.
41.    return 1;
42. }
```

נתון שהתוכנית לעיל עוברת קומפילציה בהצלחה.

<u>להלן שורות פלט אפשריות בתוכנית</u>:

(A) a1 = 5
(B) a1 = 7
(C) a1 = 13
(D) a2 = 7
(E) a2 = 13
(F) version = 0
(G) version = 1
(H) version = 2
(I) version = 3
(J) version = 4

<u>שאלה 1</u>

מה יודפס מתחילת ה-main ועד שורה 33? (=> מסמן ירידת שורה).

א. (A) => (B) => (F) => (G)

ב. (A) => (G) => (B) => (H)

ג. (A) => (F) => (A) => (H)

ד. (A) => (G) => (A) => (H)

ה. (A) => (G) => (A) => (G)

ו. אף תשובה אינה נכונה.

<u>שאלה 2</u>

מה יודפס משורה 34 ועד סוף ה-main? (=> מסמן ירידת שורה).

א. (D) => (G) => (C) => (I)

ב. (D) => (I) => (C) => (J)

ג. (E) => (I) => (C) => (J)

ד. (E) => (G) => (C) => (J)

ה. (E) => (I) => (C) => (I)

ו. אף תשובה אינה נכונה.

שאלה 3

מה היה קורה במידה והיינו מוסיפים את המילה explicit ל-c'tor בשורה 8?

א. ניתן להוסיף explicit רק ל-copy c'tor לכן זה בלתי אפשרי.

ב. לא ניתן להוסיף explicit לבנאי של מחלקה פנימית, לכן זה בלתי אפשרי.

ג. שורה 30 לא תעבור קומפילציה.

ד. שורה 35 לא תעבור קומפילציה.

ה. גם שורה 30 וגם שורה 35 לא יעברו קומפילציה.

ו. אף תשובה אינה נכונה.

שאלה 4

מורן שירן ולירן התכוננו למבחן ועברו על הקוד לעיל.

לירן טען : T - פרמטר ה-Template של המחלקה – חייב שיהיה לו אופרטור ++.

שירן טען : שורה 12 היא אופרטור casting. שורה 21 היא לא אופרטור casting.

מורן טען : בתוכנית יש זליגת זכרון מכיון שאף אחד לא משחרר את המשתנה שהמחלקה A מעבירה למחלקה הפנימית שלה בשורה 19.

מי מהשלושה צודק ?

א. לירן (הוא גם מצטיין דיקאן).

ב. לירן ומורן.

ג. שירן ומורן.

ד. שלושתם צודקים.

ה. שלושתם טועים.

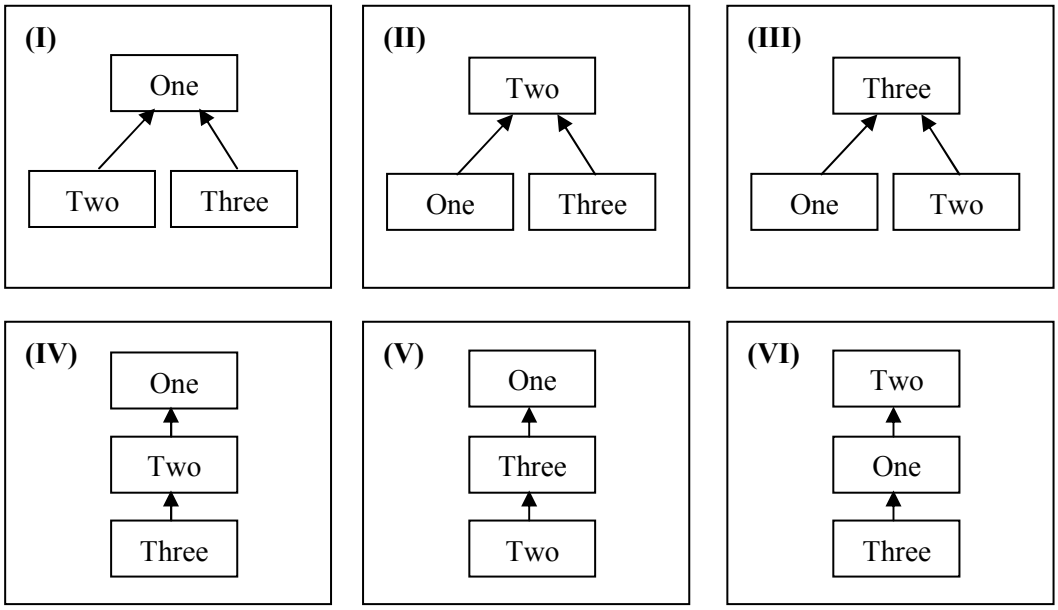
ו. אף אחת מהתשובות אינה נכונה.

שאלות 5-7 מתייחסות לקטע הקוד הבא:

```
1.  int main()
2.  {
3.      One* pOne = new Two();
4.      Three& three = pOne->doIt();
5.      *three = *pOne;
6.      three = *pOne;
7.      return 1;
8.  }
```

נתון שהקוד עובר קומפילציה בהצלחה.

להלן היררכיות הורשה אפשריות:



<u>שאלה 5</u>

איזה מההיררכיות לעיל **<u>יכולות להיות נכונות</u>** (כלומר – להתאים ל-main שהוצג)?

א. (I), (II)

ב. (II), (III), (V)

ג. (II), (III), (V), (VI)

ד. (I), (IV), (V)

ה. (III), (V), (VI)

ו. אף תשובה אינה נכונה.

שאלה 6

הנח כעת כי נתון שלא נכתב מפורשות אופרטור השמה באף אחת מהמחלקות, וכן כי לא נכתב מפורשות באף אחת מהמחלקות בנאי שמקבל כפרמטר אובייקט מהמשפחה One, Two, Three.
כמו כן הנח שאף אחת מהמחלקות לא מימשה אופרטור casting.

תחת ההנחות לעיל, איזה מהיררכיות ההורשה לעיל **יכולות להיות נכונות**?

א. (I)

ב. (III)

ג. (II), (III)

ד. (V)

ה. (VI)

ו. אף תשובה אינה נכונה.

שאלה 7

מורן שירן ולירן התכוננו למבחן ועברו על הקוד לעיל.

לירן טען: במחלקות One ו-Three בהכרח מימשו אופרטור *.

שירן טען: למחלקה Two יש Empty C'tor.

מורן טען: אם Three הינה מחלקת בסיס ישירה או עקיפה של One, אז יתכן שהמחלקה One מימשה את הפונקציה doIt פשוט ע״י שורת קוד אחת שמחזירה את this* כ-ByRef.

מי מהשלושה צודק ?

א. לירן (הוא גם מצטיין דיקאן).

ב. לירן ומורן.

ג. שירן ומורן.

ד. שלושתם צודקים.

ה. שלושתם טועים.

ו. אף תשובה אינה נכונה.

חלק ב׳ – שאלת תכנות (51 נק׳)

חברת BetterParking מעוניינת לכתוב מערכת לניהול חניונים.

המטרה היא שלקוח שמגיע לחניון יבחר מאפייני חניה ויקבל כרטיס שמפנה אותו למקום החניה המיועד עבורו. מערכת החניון תדע להפנות את הנהג למקום החניה המתאים לפי צרכיו, וכאשר הלקוח חוזר לקחת את רכבו המערכת תדע לומר לו לפי מספר הרכב היכן הרכב נמצא וכמו כן תדע לחשב את עלות החניה ללקוח.

שירותים / אילוצי החניה לדוגמה שלקוח יכול לבקש:

- שירות רחיצה
- שירות טעינה לרכב חשמלי
- חניה מוצלת אילוץ מחייב לכל משך החניה
- החזרת רכב מהירה (אסור שהרכב יהיה חסום – אילוץ מחייב לכל משך החניה)
- רכב מונע בגז (יכול לחנות רק באיזורים מסויימים – אילוץ מחייב לכל משך החניה)
- רכב גבוה (יכול לחנות רק באיזורים מסויימים – אילוץ מחייב לכל משך החניה)

שימו לב: החניון אינו חייב לתמוך בכל אפשרויות לעיל. כמו כן, יתכן בעתיד שחניונים ירצו לתמוך בשירותים ואילוצים נוספים (למשל: אילוץ לחניה עם מרחב גדול לפתיחת דלתות, או – שירות של קישוט לרכב חתן-כלה). לכן התוכנה צריכה להתייחס לשירותים / אילוצי חניה באופן כללי שלא ידרוש שינויים או תוספות בתוכנה, ככל שהדבר אפשרי. אם ישנם מאפיינים שכדאי לתארם בקוד התוכנית ממש, ניתן לעשות זאת, אך עדיין לשמור על האפשרות להוסיף שירותים ואילוצים נוספים בעתיד ככל הניתן ללא צורך בכתיבת מחלקות נוספות.

הלקוח יכול לבקש מספר כלשהו של אילוצים ושירותים, למשל: טעינת חשמל + רחיצה, או: חניה מוצלת + טעינה + רחיצה. רק שירות אחד יכול להתבצע בזמן נתון, לכן הקדימות בין שירותים נדרשים נקבעת ע״י הלקוח בעת הזנת בקשתו למערכת בכניסה לחניון. יש להתייחס רק לשירות הראשון שיש לספק. הנח כי כאשר השירות הראשון יסתיים המערכת תחפש מקום מתאים לפי השירות השני שנדרש ותיתן הוראה לעובד חניון להעביר את הרכב בהתאם, אך אין זה באחריותך.

אילוצים הנוגעים למקום החניה עצמו מחייבים לכל משך החניה וכאשר ישנם מספר אילוצים הרי הם במצטבר וכל המקומות שבהם הרכב יחנה – כולל בעת קבלת שירותים – חייבים לענות לכל האילוצים כולם.

לא תמיד ניתן לספק את בקשת הלקוח. במידה ולא נותרו מקומות פנויים בחניון המאפשרים את הקומבינציה של השירות הראשון שדרש הלקוח ביחד עם כל האילוצים הנדרשים, על המערכת להודיע בחזרה לממשק המשתמש שהצירוף המבוקש לא אפשרי (רצוי באופן שיצביע על מה היתה המגבלה). ממשק המשתמש יוציא הודעה ללקוח ויבקש בחירה מחדש אך אין זה באחריותך.

דוגמאות למצבים שבהם לא ניתן יהיה לספק את בקשת הלקוח: לקוח שביקש חניה מוצלת + רחיצה, כאשר כל עמדות השטיפה הפנויות הינן בשמש. או בקשה לחניה מוצלת + טעינה, כאשר אין עמדת טעינה מוצלת פנויה. כמו כן, האפשרות: רכב גז + טעינה, אינה חוקית ויש למנוע אותה, באם המשתמש איכשהו הצליח להזין אותה בממשק. בכל מקרה, כל האילוצים המחייבים את כל משך החניה חייבים להתקיים בכל מקומות החניה שבהם הרכב ישהה, לרבות בעת קבלת שירותים.

תעריפי תשלום:

מחיר החניה מתחלק לשניים: תשלום עבור החניה עצמה וכן תשלום עבור שירותים נוספים. השירותים ברשימה הנוכחית הינם רחיצה וטעינה חשמלית של הרכב, אך בעתיד יכולים להתווסף שירותים אחרים נוספים.

המחיר עבור השירותים אינו תלוי במשך החניה, ויכול כמובן להיות שונה בין שירות לשירות. המערכת מניחה שכל השירותים שהלקוח ביקש התקיימו ומחייבת עבור כל השירותים מראש.

המחיר עבור משך החניה מחושב לפי זמן החניה בפועל. תעריף החניה יכול להיות שונה עבור סוגי החניות השונים (מקום חניה לרכב גבוה, מקום חניה מוצל, חניה עם החזרה מהירה, וכו׳ וכו׳). החניון יכול להחליט על התעריפים למקטעי זמן לפי חלוקה לבחירתו. קביעת מקטעי הזמן חייבת להיות אחידה לכל סוגי החניה והתעריפים (למשל: שעה ראשונה, שעה שניה, שעתיים הבאות, שלוש שעות הבאות, כל שעה לאחר מכן). תעריפי החניה לסוגי החניה השונים ייקבעו על מקטעי הזמן הזהים לכולם. מכיון שבד״כ חניה עם אילוצים או תוספות (למשל – רכב גבוה) לא אמורה להיות זולה יותר מחניה ללא אילוצים, מעוניינים שהפונקציה לקביעת תעריף לא תאפשר קביעת תעריף זול יותר עבור חניה עם אילוצים, או תעריף יקר יותר לחניה רגילה ללא אילוצים, אלא אם מעבירים פרמטר אחרון נוסף עם הערך true, המסמן שאנחנו יודעים שאנחנו מזינים תעריף מוזר אבל מאשרים זאת. בחניה עם מספר אילוצים (למשל: רכב גבוה מונע בגז בחניה מוצלת) תעריף החניה יהיה הגבוה מבין האילוצים.

סיום חניה וחישוב הסכום לתשלום:

כאשר לקוח מגיע למשוך את רכבו המערכת מפנה אותו למקום החניה המתאים, מחשבת את עלות החניה הכוללת ומסמנת את מקום החניה כפנוי (מתוך הנחה שהלקוח יתפנה בהקדם).

הנח כי קיימות הפונקציות הגלובליות הבאות:

1. `long getTime()` – מחזירה ערך מספרי כלשהו שמייצג את הזמן הנוכחי.
2. `int getMinutesDiff(long from, long to)` – מחזירה את משך הזמן בדקות שעבר בין שני ערכים שחזרו מהפונקציה `getTime`.

עליך להציג ולממש את המחלקות השונות הנדרשות למערכת לניהול חניון, ללא ממשק משתמש.

שים לב:

יש לענות על סעיפי השאלה באופן מסודר לפי הסעיפים בעמוד הבא.

סעיף א׳ (25 נקודות)

כתוב את הגדרת המחלקות הנדרשות לפתרון (כל ה-prototypes ,ללא מימושים).

הקפד על שימוש נכון ב-public, protected, private ו-const.

הצעת עזר לפתרון (מומלץ אך לא חובה):

חניון מאופיין באופן הבא:

- לחניון ישנם מספר מקטעים, בדרך כלל נשתמש במקטעים על-מנת להגדיר קומות שונות בחניון או איזורים בעלי מאפיינים שונים.
- למקטע בחניון יהיה סימון ע״י מספר רץ, כמו כן יינתן לו ע״י מנהל החניון מספר קומה ואות כלשהי באנגלית.
- כל מקטע בחניון מתאפיין באוסף המאפיינים הספציפיים שלו, כלומר – לאיזה אילוצים הוא עונה ואילו שירותים הוא יכול לתת: האם מותר להכניס למקטע זה רכבים מונעי גז, האם המקטע מוצל או פתוח לשמש, האם זהו מקטע רחיצת מכוניות וכו׳. יש לשים לב שרצוי מאוד שסוגי המאפיינים השונים האפשריים לא יהיו בקוד התוכנית ממש, כלומר לא יהיו ידועים לתוכנה, אלא יתבססו על ערכים של שדות כך שניתן יהיה להוסיף עוד אפשרויות מתוך קונפיגורציה.
- מקטע חניון מכיל כמות כלשהי של חניות, כאשר לכל חניה יש מספר סידורי, מידע האם זוהי חניה חסומה, ואם כן – ע״י איזה מספר סידורי של חניה (חניה יכולה להיות חסומה רק ע״י חניה אחת מאותו מקטע. יתכן שהחניה החוסמת גם היא חסומה), ובנוסף – מספר רכב שכרגע חונה בחניה זו, אם ישנו.
- כאשר לקוח רוצה לחנות הוא יעביר רשימה של אילוצים / שירותים נדרשים.
  אפשר להחליט, אם רוצים, שמקבלים בנפרד שני פרמטרים לפונקציה: את רשימת השירותים הנדרשים, ובנפרד את רשימת אילוצי החניה.
- רצוי שניתן יהיה במהירות ובקלות להגיע ממספר רכב למיקומו בחניון בלי צורך לסרוק את כל החניון.
- לחניון ישנם גם תעריפים. התעריפים מתחלקים לתעריפי החניה ולתעריפי השירותים.
  את תעריפי החניה אפשר להחזיק באמצעות רשימה של צמדים: (א) אורך זמן, (ב) מיפוי בין סוג החניה לתעריף של חניה זו עבור מקטע זמן זה (או חלק ממנו); הצמד האחרון במערך יהיה התעריפים עבור שארית הזמן, במידה ועברנו את כל שאר מקטעי הזמן.

סעיף ב׳ (16 נקודות)

ממש באופן מלא את כל הפונקציות הנדרשות לקבלת רכב בחניון וטיפול בהשמתו לחניה בהתאם לאילוצים ולשירותים שמבקש הלקוח.

סעיף ג׳ (10 נקודות)

ממש באופן מלא את הפונקציות הנדרשות לניהול תעריפי החניון ולחישוב מחיר החניה הכולל.

סוף !