סמסטר ב' תשס"ד
מועד: ב' 16/7/2004
משך הבחינה: 3½ שעות
חומר עזר: כל חומר עזר מותר

בחינה בקורס: **תכנות מכוון עצמים ושפת C++**

מרצה: **אמיר קירש**

הנחיות כלליות לבחינה:

- 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. אם נבחרה תשובה ר' (אף תשובה אינה נכונה), חובה לספק הסבר במקום המיועד לכך.
- בסיום המבחן יש לרשום מס' ת.ז. במקום המיועד לכך בטופס התשובות, לוודא שטופס התשובות נמצא יחד עם טופס המבחן ולא ניתק ממנו, ולהגישם בתוך מחברת הבחינה.
- המבחן הינו עם חומר פתוח. כל חומר עזר מותר למעט מחשבים ניידים. 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'

מס' ת.ז. : _____

	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 1-7 מתייחסות לקטע הקוד הבא:

```
1.   class ReferenceCounter
2.   {
3.     int* m_rc;
4.   public:
5.     ReferenceCounter():m_rc(new int(1)){}
6.     ReferenceCounter(const ReferenceCounter& rc)
7.       {attach(rc);}
8.     virtual ~ReferenceCounter(){}
9.     virtual void free()=0;
10.    virtual const ReferenceCounter& operator=
11.      (const ReferenceCounter& rc)
12.    {
13.      if(this != &rc) {
14.         detach();
15.         attach(rc);
16.      }
17.      return *this;
18.    }
19.    virtual void attach(const ReferenceCounter& rc)
20.    {
21.      cout<<"in attach"<<endl;
22.      m_rc = rc.m_rc;
23.    }
24.    virtual void detach()
25.    {
26.      if(--(*m_rc)<=0) {
27.         delete m_rc;
28.         free();
29.      }
30.    }
31.  };
32.
33.  class String: public ReferenceCounter
34.  {
35.    class innerString
36.    {
37.      char* m_str;
38.      int m_size;
39.    public:
40.
```

```
41.         innerString(const char* c_str):m_size(strlen(c_str))
42.         {
43.            m_str = new char[m_size+1];
44.            cout<<"allocate: "<<(void*)m_str<<endl;
45.            strcpy(m_str, c_str);
46.         }
47.         void free()
48.         {
49.            cout<<"free: "<<(void*)m_str<<endl;
50.            delete []m_str;
51.         }
52.         friend ostream& operator<<
53.          (ostream& out, const innerString& str)
54.         {
55.            return out<<(void*)str.m_str<<": "<<str.m_str;
56.         }
57.      };
58.
59.      innerString m_innerString;
60.
61.   public:
62.      String(const char* c_str=""):m_innerString(c_str){}
63.      virtual ~String(){detach();}
64.      void free(){m_innerString.free();}
65.      friend ostream& operator<<
66.         (ostream& out, const String& str)
67.      {
68.         return out<<str.m_innerString;
69.      }
70.   };
71.
72.   void main()
73.   {
74.         String str1 = "Hello";
75.         String str2 = str1;
76.         String str3 = "Hi";
77.         str3 = str2;
78.         cout<<str1<<endl;
79.         cout<<str2<<endl;
80.         cout<<str3<<endl;
81.   }
```

השאלות עצמן מתחילות בעמוד הבא.

שאלה 1

נניח ששורה מס' 74 הדפיסה: allocate: 0x002F0848

מה יודפס בשורה 75 ?

א. allocate: 0x002F0848

ב. allocate: <some other address>

ג. allocate: null

ד. allocate: null <= in attach

ה. לא יודפס דבר.

ו. אף אחת מהתשובות אינה נכונה.

שאלה 2

מה יודפס בשורות 76 ו-77 ?

א. 76: allocate: <some address X> – 77: free: <some other address Y>

ב. 76: allocate: <some address X> – 77: in attach <= free: <same address X!>

ג. 76: allocate: <some address X> – 77: allocate: null <= free: <same address X!>

ד. 76: allocate: <some address X> – 77: תעופה של התוכנית!

ה. בשתי השורות לא יודפס דבר.

ו. אף תשובה אינה נכונה.

שאלה 3

שירן טוענת שהעובדה ששכחו להעלות את ה-Reference Counter בפונקציה attach תגרום לתעופה של התוכנית בשורה 77.

נירן טוען שאכן זו שגיאה לא להעלות את ה-Reference Counter בפונקציה attach, אבל שגיאה זו תגרום לתעופה רק ברגע שנגיע ל-detach עם מספר מחרוזות שמצביעות לאותה כתובת, ולכן התוכנית תתרסק בשורה 81 – ממש בסיום ה-main.

דירן מסכים עקרונית עם נירן (טוב, הוא חבר שלו), אבל הוא טוען שב-main הנתון התעופה שמתאר נירן לא תתרחש מכיון שממילא ה-Reference Counter יעצור באפס.

מורן לעומתם טוען שאין חובה להעלות את ה-Reference Counter בפונקציה attach. במקום זאת המחלקה מימשה אופרטור השמה וזה בסדר גמור.

מי מהחבורה צודק?

א. שירן (היא גם מצטיינת דיקאן!).

ב. נירן (הוא אמנם עצלן וגם קצת עקשן, אבל הוא בחור לעניין).

ג. דירן (הוא חתיך כמו דוגמן).

ד. מורן (הוא תמיד מגיע מוכן לכל מבחן, חבל על הזמן).

ה. למרבית ההפתעה כולם טועים.

ו. אף תשובה אינה נכונה.

שאלה 4

האם אפשר היה לרשום בשורה 68:

```
return out<<str.m_innerString.m_str;
```

א. כן, אבל ההדפסות בשורות 78-80 היו משתנות.

ב. כן, וההדפסות בשורות 78-80 היו נשארות בדיוק אותו דבר.

ג. לא ניתן – יגרום לשגיאת קומפילציה

ד. לא ניתן – יגרום לשגיאת Linker.

ה. לא ניתן – יגרום לתעופה בזמן ריצה.

ו. אף תשובה אינה נכונה.

שאלה 5

כמה פעמים נעבור בשורה 28 במהלך הריצה של התוכנית ?

א. 0

ב. 1

ג. 2

ד. 3

ה. 4

ו. אף תשובה אינה נכונה.

שאלה 6

מורן שירן דירן ונירן התווכחו מה יודפס בשורות 78-80 (בתוכנית המקורית ללא שינויים).

מורן טען שבכל שלושת השורות תודפס המחרוזת Hello אבל עם כתובת שונה בכל פעם.

נירן טען שיודפסו שלוש מחרוזות שונות אבל עם אותה כתובת.

שירן טענה שלא יכול להיות שיודפסו שלוש מחרוזות שונות עם אותה כתובת (היא ציינה בהקשר זה שנירן חזר מחופשה ביוון והוא שכח לקחת כובע). לטענתה יודפסו שלוש מחרוזות שונות עם כתובות שונות.

דירן טען שכל שלושת השורות ידפיסו את המחרוזת Hi ואת אותה כתובת

מי מהחבורה צודק?

א. מורן.

ב. נירן.

ג. שירן.

ד. דירן.

ה. כולם טועים!

ו. אף תשובה אינה נכונה.

שאלה 7

נניח שהיינו מוחקים את שורות 75 עד 80 ונשארים עם שורה 74 בלבד ב-main. מה היה מודפס בסיום ה-main (כלומר בשורה של הסוגר המסולסל) ?

א. free: <some address>

ב. free: Hello

ג. free: <some address> => תעופה של התוכנית.

ד. לא יודפס דבר בשל תעופה של התוכנית.

ה. לא יודפס דבר, אבל התוכנית לא תעוף אלא תסתיים באופן נקי.

ו. אף אחת מהתשובות אינה נכונה.

חלק ב' – שאלת תכנות (51 נק')

סעיף א' (27 נקודות)

כתוב את הפונקציות הגלובליות הבאות, שידעו לנהל שמירה וקריאה מקובץ של כל Container סדרתי המכיל איברים מעורבים.

ה-Syntax של הפונקציות צריך להיות:

```
template<class iterator>
void save(ofstream& fout, iterator begin, iterator end);

template<class iterator>
void load(ifstream& fout, iterator end);
```

אין צורך לממש את המחלקה iterator.

מותר להניח את ההנחות הבאות לגבי iterator:

- קיימות עבורו הפונקציות המסורתיות: ++, *, ==.
- קיימת עבורו הפונקציה insert(T& item), הדוחפת איבר חדש למבנה הנתונים של האיטרטור לפני מיקום האיטרטור (כלומר, אם האיטרטור המפעיל הוא במקרה end, הפונקציה insert דוחפת איבר לסוף מבנה הנתונים).

בנוסף הנח שקיימת פונקציה גלובלית לצורך יצירת אובייקטים ע"פ שמם:

```
template<class T>
T* getNewObject(const string& className);
```

השימוש בפונקציה מתבצע באופן הבא:

נניח שיש לנו היררכיית הורשה של Shapes כאשר מחלקת הבסיס Shape היא אבסטרקטית וממנה נורשים צורות שונות שנשמרות לקובץ. לצורך יצירת אובייקט מתאים לפי זיהוי האובייקט כפי שנשמר לקובץ ניתן לקרוא לפונקציה באופן הבא:

```
Shape* pShape = getNewObject(ObjectTypeFromFileAsString);
```

אם למשל המחרוזת שנשלחה היא "Circle", תערך המוחזר יהיה אובייקט חדש מסוג Circle. המחלקה Circle צריכה במקרה זה להיגזר כמובן מהמחלקה Shape, בנוסף צריך שיהיה לה בנאי שלא דורש פרמטרים (empty c'tor) כיון שהפונקציה getNewObject דורשת בנאי כזה.

שים לב – הפונקציה מחזירה הקצאה דינמית. באחריותו של מי שקורא לפונקציה זו לשחרר את הזיכרון שהוקצה.

הצג פתרון שיאפשר שימוש פשוט וקל בפונקציות שתכתוב, עם מינימום דרישות מהמשתמש.

הסבר באופן מילולי מהן הדרישות ממחלקות שנזעוניינות להשתמש בפונקציות שתכתוב.

סעיף ב' (24 נקודות)

כתוב תוכנית שתייצר רשימה (מסוג list<Shape*> של ה-STL), של משולשים (Triangle) ועיגולים (Circle) – לפי בחירת המשתמש. עבור כל צורה שהמשתמש בוחר עליו לספק את הקלט המתאים.

התוכנית צריכה להציע למשתמש את התפריט הבא:

- קריאת רשימת צורות מקובץ.
- שמירת רשימת צורות לקובץ.
- הוספת צורה לרשימה.
- הדפסת נתוני הצורות שברשימה.
- סיום.

יש להקפיד על שחרור נכון של הזיכרון בתוכנית.

הניקוד בסעיף זה מתחלק באופן הבא:

- כתיבת המחלקות הנדרשות באופן מלא – 15 נקודות.
- main + ניהול התפריט - 9 נקודות.

(אין צורך לכתוב מחלקה מיוחדת לצורך ניהול התפריט, אפשר לעשות זאת ישירות ב-main)

סוף !