# תכנות מתקדם 2

## מועד א' תשע"ב

15.7.12

המחלקה למדעי המחשב

89-211

| | | |
|---|---|---|
| 1 | 35 נק' | |
| 2 | 15 נק' | |
| 3 | 20 נק' | |
| 4 | 10 נק' | |
| 5 | 20 נק' | |
| סה"כ | 100 נק' | |

**מרצה:** יריב טל.

**מתרגל:** איגור רוכלין.

**משך הבחינה:** שלוש שעות. אין הארכה.

**חומר עזר:** אסור להכניס כל חומר עזר.

**הנחיות כלליות:** רצוי לענות בגוף הבחינה. אחרי כל שאלה יש מקום לתשובות.
אם עניתם על שאלה במחברת – ציינו זאת בגוף הבחינה !
חובה להגיש את המחברת ביחד עם טופס הבחינה בסיום הבחינה.
חובה לענות על השאלות בעברית, אלא אם יש אישור מהדיקן.

**הנחיות טכניות:** במידה ונדרשתם לתת נימוק, אזי הוא חובה. כלומר, תשובה לא מנומקת לא תקבל נקודות כלל.
נימוק לא נכון יגרור פסילת השאלה.
בכל השאלות – מספר השורות שניתנו לפתרון אינו מרמז על אורך התשובה.

בהצלחה ☺

## שאלה 1: Code Improvements (35 נקודות)

**נתון הקוד הבא:**

```
class CharStringParser
{
public:
    // str is a c-string (terminated with a '\0' character)
    CharStringParser(char *str) : m_str(str) { }

    bool isEmpty() {
        if (m_str == NULL || *m_str == '\0') {
            return true;
        }
        else {
            return false;
        }
    }

    long tryParsePositiveNumericValue() {
        long numericValue;
        if (NULL != m_str) {
            skipLeadingSpaces();

            assert(NULL != m_str);
            for (numericValue=0;isdigit(*m_str);++m_str) {
                numericValue *= 10;
                numericValue += (*m_str - '0');
            }
        }

        return numericValue;
    }

private:
    // advance m_str until the first non-space character
    void skipLeadingSpaces() {
        assert(NULL != m_str);
        while (isspace(*m_str))
            m_str = m_str + 1;
    }

    char *m_str;
};
```

**א.** (4 נקודות) האם ניתן להגדיר את m_str כטיפוס **const** char * (במקום char *), למרות שהפונקציות skipLeadingSpaces() ו-tryParsePositiveNumericValue() משנות את מיקום המצביע? נמקו!

_____

_____

_____

_____

ב. (4 נקודות) אילו פונקציות יכולות להיות מוגדרות כפונקציות const? נמקו והדגימו כיצד תראה החתימה של הפונקציות לאחר הוספת ה-const.

ג. (9 נקודות) האם השימוש ב-assert בפונקציה skipLeadingSpaces() מוצדק או לא? נמקו. אם השימוש מוצדק הסבירו מדוע בפונקציה tryParsePositiveNumericValue() אין assert דומה.
רמז: שימו לב להבדל ברמת ה-access (private/public) בין הפונקציות.

ד. (18 נקודות) איכות הקוד הנתון שנויה במחלוקת...
הציעו לפחות 6 שיפורים אשר עשויים להפוך את הקוד ליותר קריא או להוריד את הפוטנציאל שלו לבאגים.
נמקו את הצורך בכל שיפור!
היו ברורים היכן בקוד יש לבצע את השינוי.
שימו לב – הצעה של שינוי שאינו משפר את הקוד *עלולה* לגרור הורדת ניקוד!

## שאלה 2: Templates (15 נקודות)

נתון הקוד הבא:

```
template< unsigned long N >
struct Magic
{
  enum { value = (N % 10) + 2 * magic <N / 10>::value } ;
};

template<>
struct Magic<0>
{
  enum { value = 0 } ;
};
```

**א.** (5 נקודות) כיתבו אילו אינסטנסיאציות יבצע הקומפיילר ומה יודפס כתוצאה מביצוע שורת הקוד הבאה (נמקו!):

```
std::cout << Magic<0>::value << std::endl;
```

**ב.** (5 נקודות) כיתבו אילו אינסטנסיאציות יבצע הקומפיילר ומה יודפס כתוצאה מביצוע שורת הקוד הבאה (נמקו!):

```
std::cout << Magic<1>::value << std::endl;
```

**ג.** (5 נקודות) מה מבצע ה-template?

רמז: העזרו באינסטנסיאציות הבאות בכדי להגיע למסקנתכם:

```
std::cout << Magic<10>::value << std::endl;
std::cout << Magic<11>::value << std::endl;
```

## שאלה 3: Exception Safety (20 נקודות)

נתון הקוד הבא:

```
class Yard {
public:
      Bone *digupBone() {
            Hole *pHole = dig();
            if (pHole && (pHole->hasBone())) {
                  Bone *pBone = pHole->getBone();
                  releaseHole(pHole);
                  return pBone;
            }
            else {
                  if (pHole) {
                      releaseHole(pHole);
                  }
                  throw NoBoneException();
            }
      }

private:
      // Allocates a hole.
      // Freeing an unused hole is done by calling releaseHole (do not directly
      // delete the hole! You *must* call releaseHole instead).
      Hole *dig();
      void releaseHole(Hole *pHole) throw();

      std::vector<Hole*> m_holes;
};
```

א. (5 נקודות) נתונות ההנחות הבאות:

- הפונקציות releaseHole(), hasBone(), getBone() מקיימות nothrow guarantee.
- שחרור של Hole שהוקצה ע"י dig() מתבצע ע"י קריאה ל-releaseHole() (ולא ל-delete)
- הפונקציה dig() מקיימת strong exceptions guarantee.

איזה guarantee מקיימת הפונקציה digupBone()? נמקו!

ב. (5 נקודות) איזו בעייה עלולה להיווצר אם הפונקצייה getBone() (שנקראת מהפונקציה digupBone()) יכולה לזרוק exception?

ג. (10 נקודות) כל Hole שמוחזר ע"י ()dig יש לשחרר כשמסיימים להשתמש בו ע"י קריאה ל-releaseHole().
בתוכנית הנתונה שחרור זה מתבצע ידנית (ישנן קריאות ל-()releaseHole במקומות המתאימים).
כיתבו Guard אשר יהיה אחראי לשחרור ה-Hole במקום הקריאות המפורשות ל-()releaseHole. הדגימו היכן וכיצד יש לבצע את איתחול ה-Guard בפונקצייה ()digupBone.

## שאלה 4: Java (10 נקודות)

א. הסבירו בקצרה: מה זה **Servlet**, מה זה **JSP** ומה ההבדלים ביניהם.

ב. הסברו בקצרה מה זה **Reflection** ומה השימוש במנגנון הזה.

## שאלה 5: Java (20 נקודות)

א. (5 נקודות)

הסבירו בקצרה למה נועד תרשים המחלקות (class diagram)?

למה נועד תרשים זרימה (flow diagram)?

ב. מחלקת התקצוב של חברה "X" מעוניינת להקים מערכת מידע אוטומטית ככל הניתן שתתמוך בהפקת דו"חות שכר וחישוב רווח או הפסד שנתי.
המערכת מקבלת מידע אודות עובדי החברה (מס' זהות, שם תפקיד, תאריך התחלת עבודה) ומעבדת אותו באופן הבא: בתחילת כל חודש, המערכת מפיקה דו"ח משכורות העובדים. המשכורות מחושבות על סמך טבלת תפקידים (המתאימה לכל תפקיד סכום שקלי) וטבלת ותק (המתאימה לתקופת ותק מסוימת מקדם הכפלה, לדוגמא: עובד עם ותק בין 3 ל 5 שנים, משכורתו תוכפל ב- 1.2).

בנוסף מערכת המידע מנהלת את תקציב החברה.
לכל תקציב יש: מספר תקציב, תקופת תקציב, סכום התחלתי ואחראי או אחראים לתקציב, שהם עובדי החברה.
כל עובד יכול להיות אחראי למספר תקציבים. אחראי תקציב יכול להגיש בקשות להוצאות סכומים מתקציבו.
כל בקשה תכיל את סיבת הבקשה, תאריך והסכום המבוקש.

- (4 נקודות) מיהם ישויות (actors) של המערכת?

- (4 נקודות) מהם הטרנזקציות (use cases) העיקריות של המערכת?

(7 נקודות) צרו תרשים המחלקות של המערכת (class diagram). עבור כל מחלקה, רשום את תכונותיה והקשרים המבניים למחלקות אחרות.