מרצה: נעם רינצקי מתרגל: דוד טרביז חומר: פתוח משך: שלוש שעות

מבחן בקומפילציה – 2019/20 – מועד א

המבחן מורכב מחמש שאלות. יש לענות על כולן.
מומלץ לקרוא את השאלה עד סופה לפני שמתחילים לענות.
משקל השאלה ומספרה אינה מעיד על הקושי בפתירתה.
יש לציין בראש העמוד את השאלה עליה עונים.
אין לענות על שאלות שונות באותו העמוד.
תשובה "איני יודע/ת" תזכה ב 20% מהניקוד על הסעיף הרלוונטי.

<u>שאלה 1 (20 נקודות)</u>

נתונה רשימת מאורעות. לכל אחד מהמאורעות הסבירו בקצרה (3-1 שורות)

1. האם הוא מתרחש במהלך בניית הקומפיילר/הקומפילציה /ריצת התוכנית.
2. אם בחרת בזמן קומפילציה ,הסבר מהו השלב המסוים בקומפיילר בו מתרחש המאורע ומהם מבני הנתונים הרלבנטיים.
3. אם מאורע יכול להתרחש במספר שלבים, ציינו את כולם.

א. (4 נקודות) נמצא כי התוכנית מכילה לולאות מקוננות.
ב. (4 נקודות) נמצאה כי יש בתוכנית מחרוזת קבועה המכילה יותר מ 256 תווים.
ג. (4 נקודות) נתגלה כי זיכרון המחסנית חורג מ 1024 בתים.
ד. (4 נקודות) נתגלה כי בקוד יש רק קריאות למתודות בהיסט +4 בטבלת הפונקציות הוירטואליות.
ה. (4 נקודות) נמצא כי קיימת פקודת חיבור שלא תתבצע בשום ריצה של התוכנית.

<u>שאלה 2 (20 נקודות)</u>

ענו על הסעיפים עבור הדקדוק הבא:

S ::= E$

E ::= E < T

E ::= T

T ::= int < int

T ::= int

א. (5 נקודות) הראו שהדקדוק רב משמעי
ב. (10 נקודות) ציירו את דיאגרמת המצבים (המלאה) של SLR והציגו בה את הקונפליקטים המתקבלים
ג. (5 נקודות) שכתבו את הדקדוק כך שיתקבל דקדוק 1LL שמקבל את אותה השפה

<u>שאלה 3 (15 נקודות)</u>

הוחלט להוסיף לשפת פיקאסו הסקת טיפוסים אוטומטית,
בתנאי שבזמן ההכרזה יש ערך אתחול.
תארו את השינויים הנדרשים, אם בכלל, בשלב המנתח הסמנטי ויצירת הקוד.
דוגמא:

```
auto i := 989 - 520;
auto s := “HelloWorld”;
class Person { int age; }
Person me := new Person;
auto you := me;
```

שאלה 4 (25 נקודות)

הוחלט להוסיף לשפת פיקאסו את האפשרות לעשות **inlining** לפרוצדורות עלה,
אשר אינן קוראות לפרוצדורות אחרות. תארו את השינויים הנדרשים, אם בכלל, בשלב יצירת הקוד.
דוגמא:

```
inline int bar( int t ) { int k := 17; return k-t; }
int foo1() { int t := 6; return t + bar(40); }
Int foo2() { int k := 6; return k - bar(80); }
```

שאלה 5 (20 נקודות)

הוחלט לשנות את האלגוריתם לניתוח טווחי חיות של משתנים באלגוריתם הבא:

```
for each cmd in CMDs:
        live-in [ cmd ] = live-out [ cmd ] = { }
        live-in-tag [ cmd ] = live-out-tag [ cmd ] = { }
do
{
        for each cmd in CMDs:
                live-in [ cmd ] = live-in-tag [ cmd ]
                live-out [ cmd ] = live-out-tag [ cmd ]
        for each cmd in CMDs:
                live-in-tag = use [ command ] U live-out [ command ]
                for each n in SUCCESSORS(cmd):
                        live-out-tag = live-out-tag U live-in [ n ]
                        // originally: live-out-tag = live-out-tag U (live-in [ n ] - def [ n ])
}
while ((live-in != live-in-tag) || (live-out != live-out-tag))
```

קבעו ונמקו מה הטענה הנכונה לגבי גרף ההפרעות הנוצר מאלגוריתם זה:

א. מוכל בגרף ההפרעות המקורי
ב. מכיל את גרף ההפרעות המקורי
ג. שתי התשובות הקודמות אינן נכונות בהכרח