שפות תכנות - ש.ב. 3

עמרי גיא 065982415

גיא שקד 036567055

22 בדצמבר 2010

1.

2.

3.

4. אין דמיון רב בין go-routines ו־co-routines.

באופן כללי, co-routine היא רוטינה המאפשרת מספר נקודות כניסה בהתאם לנקודות בהן הרוטינה עוצרת את ריצתה. זאת בניגוד לרוטינה רגילה בה יש נקודת כניסה אחת קבועה. ב־co-routine יש בנוסף למנגנון ה־return הקיים ברוטינות רגילות גם מנגנון yield, המחזיר ערך (אופציונלית) לפונקציה הקוראת וממשיך את הריצה שלה, אך גורם לכך שבפע הבאה שה־co-routine תקרא ריצתה תמשיך מהפקודה הבאה אחרי פקודת ה־yield (ולא מתחילתה, כמו ברוטינה רגילה). מנגנון זה מאפשר יצירת איטרטורים, גנרטורים, רשימות אינסופיות, צינורות וכו'...

בשפת Go, go-routine הן פונקציות הנקראות עם המילה השמורה go לפניהן. go-routine רצה במקביל לקוד שקרא לה וערך ההחזרה שלה לא מושם למשתנה. מנגנון ה־go-routine הוא במידה רבה מנגנון של חוטים רזים, ה־go-routine רצה באותו מרחב כתובות כמו הקוד שקרא לה והתקורה שביצירתה והפעלתה אינה גדולה, היא מאפשרת ביצוע משימות במקביל העברת ערכים באמצעות ערוצים (channels) וסנכרון באמצעי סנכרון מקובלים אחרים (מנעולים, למשל).

5. (א) מתכנני השפה בחרו במנגנון simultaneous assignments המאפשר לחלק הימני של ההשמה להיות tuple או רשימת איברים, משערך את כולו ואז משים את האיבר ה־n לאיבר ה־n ברשימת האיברים שמופיעה בצד השמאלי. הדרישה היא שמספר האיברים בשני הצדדים (או - מספר האיברים ב־tuple שבצד ימין) יהיה זהה, והטיפוסים ה־nים בשני הצדדים יתאימו (לכל n). מנגנון זה מתיישב היטב עם תכונות ה־strong typing שבשפה, מממש את היתרונות שבהשמה סימולטנית (למשל - החלפה בין משתנים ללא צורך במשתנה עזר) והשימוש במזהה הריק (_) מאפשר להשתמש רק בחלק מהמשתנים שברשימה מצד ימין מבלי לפגוע בדרישות התאמת מספר המשתנים והסדר בינהם.

(ב) ניתן למשל לחשב סדרת פיבונאצ'י. בדוגמא הבאה - קוד המחשב את מספרי פיבונאצ'י עד ל־1000 -

```
first,sec := 0,1;
for (sec < 1000) {
  fmt.Println(first);
  first, sec = sec, first+sec;
}
```