שפות תכנות, 234319

חורף תשע"ח

****

תרגיל בית 5

*ML sequences, Storage, Commands*

**תאריך פרסום**: 1.1.2018

**מועד אחרון להגשה**: 15.01.2018

**מועד אחרון להגשה מאוחרת**: 18.01.2018

**מתרגל אחראי**: ליאור בן-ימין

**אי-מייל**: lior.b@cs.technion.ac.il

**בפניה בדוא"ל, נושא ההודעה (subject) יהיה "PLW17-EX5" (ללא המרכאות).**

**תרגיל בית זה מורכב משני חלקים, חלק יבש וחלק לא יבש. לפני ההגשה, ודאו שההגשה שלכם תואמת את הנחיות ההגשה בסוף התרגיל.**

**תיקונים והבהרות יפורסמו בסוף מסמך זה, אנא הקפידו להתעדכן לעתים תכופות.**

# **חלק יבש**

## שאלה 1 - RTTI

1. **קראו את** [**פרק 5.6 בחוברת השקפים**](https://webcourse.cs.technion.ac.il/234319/Winter2017-2018/ho/WCFiles/_handout-5-6-run-time-type-information.pdf) **בנושא RTTI.**

**מהו מנגנון RTTI? סכמו בשני משפטים.**

המנגנון RTTI, זהו רפרנס לתיאור הטיפוס המשותף לכל הערכים של הטיפוס (אורך, האם מחולק והדרך לפענח כל חלק):

* Dynamic typing – בדיקות טיפוס בזמן ריצה.
* Dynamic / static typing -איסוף אשפה, deep cloning.

1. **האם בשפת C יש מנגנון RTTI?**

**אם כן - תארו אותו.**

**אם לא - הסבירו למה.**

בשפת C אין מנגנון RTTI בגלל עיקרון ה"אין מחיר נסתר" בשפה. מכיוון שה-RTTI הוא מנגנון הפועל "מאחורי הקלעים", ה-RTTI נוגד את עקרונות השפה ב-C.

1. **הסבירו: כיצד מנגנון איסוף אשפה משתמש במנגנון RTTI?**

מנגנון RTTI שומר מידע מהיכן המשתנה הגיע ומי מחזיק רפרנס אליו. ללא מנגנון ה-RTTI לא נוכל לדעת אילו משתנים הם אשפה ואילו לא, ולא נוכל לאסוף אותם לאשפה.

1. **תארו שימוש אחר (שאינו איסוף אשפה) למנגנון RTTI.**

שימוש נוסף הוא, לדוגמא ב-C++ המנגנון מאפשר ביצוע dynamic\_cast (המרה בטוחה בין מצביעים), כלומר, בדיקה של עצם נתון ולאיזה טיפוס הוא משתייך.

## 

## שאלה 2 - KOTLIN

**ענו על הסעיפים הבאים בהקשר של שפת Kotlin:**

1. **מהם הפקודות האטומיות בשפה?**

פקודה ריקה, רצפים.

אין פקודת אטומית להשמה בשפה – מכיוון שהשמה ב-kotlin קוראת לפונקצית השמה, כלומר השמה ב-kotlin היא command expression.

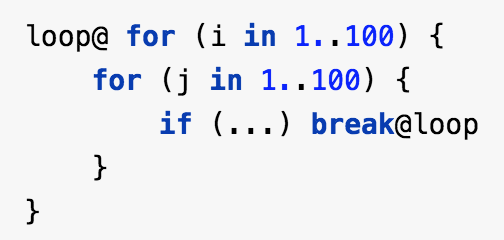
1. **מהם בנאי הפקודות בשפה? לאילו בנאים תיאורטים הם מתאימים?**

If ב-kotlin הוא ביטוי המחזיר ערך, לכן הוא command expression והבנאי התאורטי שלו הוא condition command.

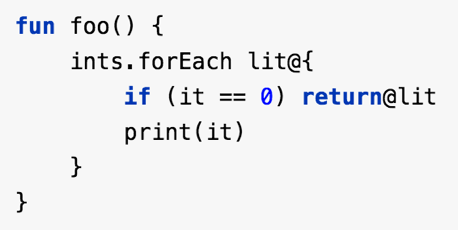
1. **מהם ה-Sequencers של השפה?**

בדומה לC/C++ קיימות הפקודות: Return,Break,Continue המשמשות לשינוי רצץ התוכנית באופן דומה בKotlin.

בנוסף בKotlin קיים הסימון ״@״ היכול להיות מצורף לכל ביטוי בKotlin כך שהוא יהפוך להיות “Label” שיהיה באפשרות המתכנת ל״קפוץ״ למקום הזה ע״י break@/continue@:



בנוסף יש return@ בפונקציות למדה(lambda function), באופן הבא:



קיימת גם מערכת ניהול שגיאות(Exception) בדומה לC++ ע״י המילים השמורות try/catch/finally.

1. **מהם סוגי המערכים הקיימים בשפה? מה קורה בחריגה מגבולות מערך?**

Array – נסווג אותו תחת מערכים דינאמי – הגודל נקבע בזמן ריצה, לא ניתן לשנות את הגודל לאחר יצירה מוקצה בheap.

1. **האם קיימים מנגנוני איסוף אשפה בשפה? אם כן, פרטו על אחד מהם.**

מנגנון הGarbage Collector בKotlin זהה לזה של שפת הJava.

[mark-and-sweep algorithm](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)):

* מאפסים את הסימונים של מקטעי זיכרון שהוקצו, באמצעות ערימה שמחזירה מצביע לכל קטע קוד שהוקצה.
* נתחיל מהשורשים, נבצע חיפוש בגרף ערכים הנגישים מהשורש ונסמן אותם.
* נעבור שוב על כל מקטעי הזיכרון שהוקצו, ו"נאסוף" את כל מי שלא סומן. תוך כדאי נבצע איחוד של קטעי זיכרון חופשיים לבלוק אחד.

## 

## שאלה 3 - BASH

**ענו על הסעיפים הבאים בהקשר של שפת bash:**

1. **סווגו את מערכת הטיפוסים שלה לפי הקריטריונים שנלמדו בתרגול ובהרצאה.**

* Existance: כל המשתנים בשפה הם מאותו סוג: מערך של מחרוזות. לכן אין ב

BASH מערכת טיפוסים, כלומר השפה היא Untyped. (הערה: נשים לב

שפעולות אריתמטיות על משתנים אפשרית רק כאשר המחרוזת היא מספר, אך מחרוזות כאלו אינן מהוות טיפוס נפרד. כלומר הסמנטיקה נקבעת לפי הערך ולא לפי הטיפוס).

* + Strength: מכיוון שאין מערכת טיפוסים (כל המשתנים מאותו טיפוס) אז אין צורך

באכיפה, או בהמרות מסוימות לפני ביצוע פעולות כלשהן בין משתנים. התנהגות כזאת מכונה Weak Typing.

* + Time of checking: השפה מאופיינת ב Dynamic Typing. כפי שהוזכר לעיל

אין מערכת טיפוסים, לכן אין צורך לבדוק טיפוסיות של משתנים לפני ריצה. בזמן ריצה תיווצר שגיאה כאשר ננסה להפעיל פעולות אריתמטיות על מחרוזות שאינן מייצגות מספרים.

* + Type Equivalence: קיים רק טיפוס אחד לכן כל הערכים שקולים מבחינת

הטיפוס.

* + Flexibility: כתוצאה מהעדר טיפוסיות למתכנת יש חופש פעולה רחב מאוד.

1. **חזרו על** [**שאלה 2**](#_lpn2gxak62tj) **עבור bash.**

2.1. מהם הפקודות האטומיות בשפה?

פקודת הצבה, ביטול הגדרה של משתנה, sequencers, הפקודה הריקה.

2.2. מהם בנאי הטיפוסים בשפה? לאילו בנאים תיאורטיים הם מתאימים?

בנאי הפקודה בשפה הם:

בנאי תנאי – לדוגמא if/then/else

בנאי לולאה – לדוגמא for, while, until.

2.3. מהם ה-sequencers של השפה?

בBash יש continue וגם break. בנוסף יש אפשרות לשנות את כמות האיטרציות של לולאה ע״י “continue N”. עבור חזרה מפונקציה קיים השימוש בreturn בדומה לC/C++.

2.4. מהם סוגי המערכים הקיימים בשפה? מה קורה בחריגה מגבולות מערך?

ב-bash יש מערך יחיד מסוג flexible array, גודלו נקבע בזמן ריצה, וניתן לשינוי לאחר יצירה. כמו כן, ניתן לגשת לאיבר רק באמצאות אינדקס מסוג int. בנוסף, השפה לא מטפלת בשגיאת חריגה מגבולות מערך, ולא זורקת שגיאות.

2.5. האם קיימים מנגנוני איסוף אשפה בשפה? אם כן, פרטו על אחד מהם.

לשפת Bash אין Garbage Collector.

1. **כיצד נקבעת נקודת התחילה של התכנית בשפת bash? מהם "גבולות" התכנית? האם השפה אוטרקית?**

נקודת ההתחלה של קטע קוד בשפת BASH היא הפקודה הראשונה בקובץ. באותו אופן, סוף התכנית מוגדר להיות הפקודה האחרונה בקובץ - אלו הם גבולות התכנית. משום שגבולות אלו מצויים בקובץ אחד ויחיד השפה עונה על ההגדרה של שפה אוטרקית - שפה שבה ניתן לדעת בדיוק איזה קוד יתבצע מהסתכלות בתכנית בלבד. כמו כן, אין צורך לייבא ספריות חיצוניות על מנת להשתמש בפונקציות קלט\פלט שונות אשר מובנות בשפה, כמו echo.