דברים קטנים ושימושיים:

חיתוך string – str.substring(int start, int end), או רק התחלה ואז זה יהיה עד הסוף.

המרת String ל-int – Integer.parseInt(str)

המרת String ל-double – Double.parseDouble(str) (בשניהם אם לא עובד נזרק NumberFormatException)

אורך של מערך – Arr.getLength(arr) / arr.length

חיתוך מערך – Arrays.copyOfRange(start, end) – לא כולל את האחרון

יצירה של רשימה ממערך – Arrays.asList(arr)

המרה ל-String - toString()

בדיקת אובייקט – a instanceof b

Enum – הבנאי (אם יש) הוא בלי public, פשוט לשים את השם שלו

שלוש מתודות של Object – String toString(), boolean equals(Object other), hashCode()

ב-equals נבדוק תחילה שהאובייקט השני הוא בכלל מהסוג שאנחנו בודקים, ואם כן אז נבדוק אם הם שווים.

hashCode() – נותן ייצוג מספרי לאובייקט לפי המיקום בזיכרון בברירת מחדל – ניתן לדרוס ולהגדיר ייצוג שונה, העיקר שיחזיר int.

\*\*\*\*\*אם דורסים את equals חייבים לדרוס גם את hashCode, צריך לוודא כי שני ערכים זהים יקבלו את אותו הערך לגיבוב.

Objects.hash(args) – מכניסים פשוט מלא משתנים לא משנה מה וזה מחזיר int

Packages – כותבים למעלה package ואת השם שלה. מייבאים מחלקה מהחבילה עם

import packageName.className  
לא לשכוח להגדיר את המחלקה כפומבית

מימוש בממשק – באמצעות שימוש במילה השמורה default – default void…..

Super – ניתן להשתמש בשביל להגדיר גבול תחתון עם wildcard (כמו השימוש ב-extend)

Assert – זו התבנית: assert (condition): “error message” (הודעת השגיאה והנקודתיים זה אופציונאלי)

חריגות – לא לשכוח לציין על כל מחלקה / מתודה שהיא זורקת את החריגה, גם היא מגלגלת את זה אחורה.

זה חשוב:

Predicate<T>       (   T -> boolean  ) – test()

Supplier<T>        (  void -> T  ) – get()

- Consumer<T>        (  T -> void  ) – accept()

Function<T,R>      (  T -> R  )  - apply()

BiFunction<T,U,R>      (  T,U -> R  )  - apply()

Runnable – ממשק מובנה שמייצג פונקציה שלא מקבלת כלום ולא מחזירה כלום – run()

הכלה ב-String - str.contains(String other)

אם מכיל char – str.indexOf(char c) (מחזיר את האינדקס הראשון, אם לא קיים מחזיר -1)

מחלקה פנימית לא יכולה להכריז על מתודות סטטיות (בגרסה מתקדמת של ג׳אווה אפשר)