א. כתבו מחלקה גנרית בשם AssociationTable המממשת טבלה אסוציאטיבית ממויינת (בסדר עולה) המיועדת לאחסון זוגות של איברים כאשר כל זוג מורכב ממפתח וערך. טיפוס המפתח יהיה טיפוס כלשהו שניתן להשוואה, כלומר, טיפוס המממש את הממשק Comparable<T>. ואילו טיפוס הערך יכול להיות טיפוס כלשהו (כל הערכים מאותו טיפוס).
המחלקה תכלול שני **בנאים**, אחד יוצר טבלה ריקה והשני מקבל מערך של מפתחות ומערך של ערכים כאשר כל איבר במערך המפתחות יהיה קשור לאיבר התואם במערך הערכים.
האיברים במערך המפתחות אינם מסודרים בסדר כלשהו. הבנאי ייצור טבלה אסוציאטיבית ממויינת. במקרה שיש מפתחות שווים הערך האחרון יחליף את הערך הקודם. במקרה שהמערכים אינם שווים בגודלם, יש לעורר מצב חריג מסוג IllegalArgumentException, (עליכם להגדיר מחלקה זו כתת-מחלקה של Exception).

המחלקה תכלול את **הפעולות** הבאות:

- add - פעולה המקבלת מפתח וערך ומוסיפה אותם לטבלה הממויינת במקומם המתאים. אם המפתח קיים כבר בטבלה, יש לעדכן את הערך הקשור אליו.
- get - פעולה המקבלת מפתח ומחזירה את הערך שמשויך למפתח שהתקבל. במקרה שהמפתח לא קיים, יש להחזיר null.
- contains – פעולה בוליאנית המקבלת מפתח ובודקת אם הוא נמצא בטבלה.
- remove – פעולה בוליאנית המקבלת מפתח ומסירה מהטבלה את המפתח והערך הקשור אליו. הפעולה מחזירה true אם המפתח היה בטבלה והזוג הוסר בהצלחה אחרת יוחזר false.
- size - פעולה המחזירה את מספר הזוגות בטבלה.
- keyIterator – פעולה המחזירה Iterator המאפשר מעבר על מפתחות הטבלה. המפתחות יתקבלו לפי סדר המיון.

ממשו את הטבלה הממויינת באמצעות המחלקה java.util.TreeMap.

ב. בדקו את המחלקה הגנרית AssociationTable עבור מפתחות מסוג Student וערכים מסוג Integer. הגדירו עבור Student מחלקה מתאימה שתכלול פרטים כגון שם פרטי, שם משפחה, ת"ז, ושנת לידה. את ההשוואה בין הסטודנטים יש לבצע לפי ת"ז. הערך מסוג Integer ייצג את הציון הממוצע של הסטודנט. כתבו במחלקה נפרדת תכנית ראשית היוצרת טבלה עם 3 סטודנטים כרצונכם. הוסיפו סטודנט חדש, עדכנו ציון ממוצע ומחקו סטודנט קיים. לבסוף הציגו בפלט הסטנדרטי את הרשימה הממויינת של הסטודנטים.