3. שאלות יבשות על SortedList

1. הדרישות ההכרחיות שטיפוס T חייב לקיים הן:
קיום אופרטור > (קטן) בין שני אלמנטים מטיפוס T – זאת כדי לשמור על המבניות של הרשימה הממויינת שהיא לא יורדת.

קיום של בנאי העתקה לטיפוס מסוג T - כי כשנידרש ליצור רשימה חדשה בבנאי ההעתקה של SortedList אנו נרצה *להעתיק* את התוכן שבטיפוס T, מבלי שיהיו תלויות בין הרשימות.

קיום הורס לטיפוס מסוג T – כי כשנידרש להרוס רשימה של איברים מטיפוס T אנו חייבים להרוס כל אחד מהאיברים ונעשה זאת ע"י הפעלת ההורס של T.

2. הבעיה שעלולה להיווצר כשנחזיר רפרנס לא קבוע לטיפוס מסוג T היא:
המשתמש יוכל לשנות את הערכים של הרשימה ולא נוכל להבטיח את הישארות הרשימה ממויינת לכל אורך הריצה של התוכנית כפי שהיינו רוצים.

3. בשפת C++ ישנן שתי דרכים להעביר פרדיקט בעזרת template:
   1. באמצעות מצביע לפונקציה(כפי שלמדנו בשפת C) – בדרך זו אנו נעביר לפונקציה בתור הפרדיקט את המצביע לפונקציה בצירוף * ואז כל מקום בו רשום predicate למעשה "יוחלף" בשם של הפונקציה. אנו נפעיל את הפונקציה באמצעות כתיבת השם שלה ולתוך הסוגריים נכניס את הארגומנטים שהפונקציה צריכה לקבל והקריאה תתבצע כרגיל.
   2. באמצעות function object כפי שלמדנו בהרצאה – בדרך זו אנו נעביר לפונקציה אובייקט שהוא function object בתור הפרדיקט.
   משום שהמימוש הוא באמצעות template הקומפיילר "יחליף" את כל ההופעות של predicate בהופעות של הfunction object ששלחנו.
   ל-function object שנעביר צריך להיות העמסה של אופרטור ().
   כאשר נפעיל את האופרטור () ונכניס את הארגומנט הנדרש לתוכם למעשה נקרא לפונקציה שממומשת באמצעות האופרטור () של הfunction object(ובתוכה יש תנאי כלשהו).

אין צורך בשני מימושים שונים בשל העובדה שמימשנו את הפונקציה באמצעות template(גנרי). הקומפיילר(אם יידרש) ייצר לנו את שני המימושים השונים באמצעות אותה הפונקציה רק ע"י החלפה של הpredicate בטיפוס המתאים.