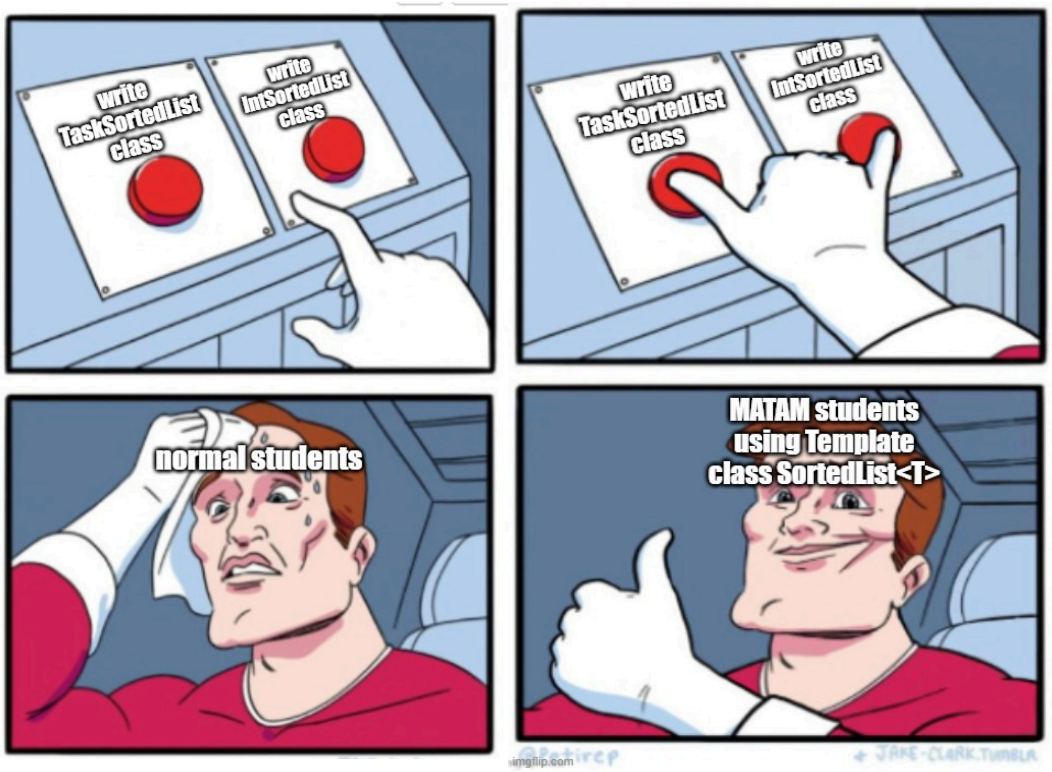

- 3.1

1) במבנה sortedlist הגנרי, הדרישות של הרשימה מהטיפוס T הן:
   1. <operator - אופרטור גדול מ- - על מנת שנוכל למיין את הרשימה, צריך להיות לטיפוס T אופרטור גדול מ- בכדי שנוכל לקבוע יחס סדר ולדעת על פי מה למיין את הרשימה הממויינת.
   2. copy c'tor - שנוכל לשכפל את הערכים המתקבלים לרשימה.
   3. d'tor - בכדי שבעת מחיקת הרשימה נוכל למחוק את איברי הרשימה מסוג T ללא דליפת זיכרון.

2) במידה והיה מתקיים את הנשאל בשאלה, היה ניתן לגשת לאיברים פרטיים ולשנות את T הישר מהאיטרטור, דבר אשר היה גורם להתנהגות בלתי רצויה, גישה לא נכונה למשתנים ואף פגיעה במיון הרשימה עצמה!

המשך בעמוד הבא

3) ניעזר בפונקציית למדה כך:

The basic syntax for a lambda expression is:

```
[] (<parameters>) -> <return-type> { <code> }
```

תחילה נוודא שהמספר שקיבלנו הוא לא 0, אם הוא 0 אז נזרוק שגיאה.
אחרת:
עבור רשימה בשם someList:

```
<typename N>
    try{
      SortedList<int> newlist = someList.filter( [N divider](int number){return(
static_cast<N>(number) % divider == static_cast<N>(0);});
       someList = newlist;
    }
    catch (std::bad_alloc& e){
      throw;
    }
```

כלומר,
נגדיר פונקציית למבדא שמקבלת בcapture list שלה את המספר המחלק, divider, בפרמטרים היא מקבלת את המספר השלם (טיפוס T עבור sortedList), ומחזירה האם שארית החלוקה של המספרים שווה ל-0. נשים לב שעשינו cast למספרים, ומכיוון שמדובר ב-int , אשר נחשב "הטיפוס הקטן", הcast יהיה תקין.

```
.filter( [N divider](int number){return( static_cast<N>(number) % divider ==
static_cast<N>(0);});
```