חלק יבש:

א. על מנת לאפשר איטרציה על תור קבוע (const) הממשק מאפשר לנו שימוש ב-ConstIterator בנוסף לאיטרטור הרגיל של התור. מדוע לא ניתן להסתפק בלהגדיר את פעולות האיטרטור הרגיל כ-const?

כיוון שכאשר נרצה לעבוד על תור שאינו קבוע, נרצה לאפשר מודיפיקציה של התור באמצעות האופרטור *() (שבמקרה של איטרטור רגיל יחזיר לנו &T) , בעוד שעבור תור קבוע לא נוכל לאפשר זאת (ולכן האופרטור *() יחזיר לנו const &T).

לצורך העניין גם לא נוכל להגדיר שתי גרסאות למתודה – אחת שמחזירה const ואחת שלא, כיוון שלקומפיילר לא תהיה דרך לדעת לאיזו אחת מן השתיים קראנו. הדרך לגרום לו לקרוא לפונקציה שמחזירה const היא להגדיר אותה לעבוד על אובייקט const, אך במקרה של מתודות על איטרטור, זה ייאלץ אותנו לקבוע את האיטרטור כconst וזה דבר שאף פעם לא נרצה לעשות, כיוון שללא האפשרות לשנות את ערך האינדקס שלו, אין לאיטרטור שימוש.

ב. באילו מהפונקציות בממשק התור קיימות הנחות על הטיפוס הטמפלייטי? עבור כל אחת מהפונקציות הללו פרטו את ההנחות.

<u>ב**בנאי**, ב**אופרטור ההשמה** ב**בנאי ההעתקה**, ב**expand** (הגדלת המערך) וב**shrink** (הקטנת המערך)-</u> אנחנו מאתחלים מערך של אובייקטים של הטיפוס הטמפלייטי, כלומר מניחים שקיים לו **בנאי חסר פרמטרים**.

<u>ב**כל הפוקנציות הנ"ל למעט הבנאי** וכן ב**Destructor**</u> – אנחנו מבצעים מחיקה של מערך של האובייקטים, במהלכה נקרא ההורס של האובייקט. כלומר אנחנו מניחים שקיים לטיפוס **Destructor** (למעשה כל טיפוס זקוק לd'tor). כמו כן, אנחנו מבצעים השמה של אברי מערך אחד לאברי מערך אחר (רצים בלולאה), כלומר מניחים שקיים לטיפוס **אופרטור השמה**.

<u>בפונקציית **PushBack**</u> – אנחנו יוצרים עותק של הרפרנס שהתקבל מהמשתמש ובכך מניחים שקיים לטיפוס **בנאי העתקה**. אנחנו גם כן מבצעים העתקה של אובייקט למערך, כלומר מניחים שקיים לטיפוס **אופרטור השמה**.

ג. סטודנט בקורס מבוא לתכנות מערכות שכח מהאזהרות שקיבל בתרגול ומימש את המחלקה Queue בקובץ cpp במקום בקובץ h . מהי השגיאה שיקבל כאשר ינסה לקמפל את התרגיל ובאיזה משלבי הקומפילציה היא מתרחשת?

ב++c הtemplate רץ בזמן קומפילציה ולכן כבר בשלב יצירת הobject file (.o) שבו הקומפיילר נתקל לראשונה בדרישה לייצר אובייקט חדש מtype מסויים לפי התבנית, הוא חייב לקבל את כל המידע ולא יכול לחכות לשלב הlinker שבו יקושר המידע מקובץ הcpp. כלומר התלות הזו בין הקבצים השונים היא שתגרום **בזמן הקומפילציה** לשגיאות שמעידות שהקומפיילר לא מזהה את הטיפוס (undefined reference, was not declared in this scope).

ד . סטודנטית בקורס מבוא לתכנות מערכות סיימה לפתור את תרגיל בית 3, והחליטה להשתמש במימוש התור מהתרגיל לפרוייקט צד שהיא מפתחת בשעות הפנאי. במימוש פרוייקט הצד הסטודנטית נדרשה לסנן תור של מספרים שלמים, כך שישארו בתור רק מספרים המתחלקים במספר כלשהו שאינו ידוע בזמן קומפילציה אלא רק בזמן ריצה. הסבירו כיצד ניתן לממש את הפונקציונליות הדרושה בעזרת הפונקציה filter ?

הסטודנטית תייצר על פי הtemplate של Condition שהוא חלק מהinterface של Queue מחלקה בשם DividedBy שהבנאי שלה מקבל int כלשהו והיא בעלת מתודה מסוג אופרטור**()** שבודקת האם שארית החלוקה של int אחר המתקבל אצלה כארגומנט באותו מספר היא 0.
בזמן ריצה, תתקבל ההכרזה על אובייקט חדש (Function Object או Functor) מהמחלקה DividedBy שתעביר לבנאי שלו את המספר שרוצים לבדוק את החלוקה בו. את אותו אובייקט ניתן כעת להעביר כארגומנט בקריאה לfilter יחד עם התור שברצונה לערוך, ולבצע השמה של התור החדש המוחזר מfilter לתור המקורי.