שאלה 1)

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?

תשובה 1)

לא ניתן להסתפק בפעולות האיטרטור הרגיל כconst כי אם כן אז לא נוכל לבצע פעולות עליו שצריכים אותם כמו קידום, ולכן הוא ישרת מטרה ממש שונה ממה שאנחנו היינו מתכוונים.

שאלה 2:

באילו מהפונקציות בממשק התור קיימות הנחות על הטיפוס הטמפלייטי? עבור כל אחת מהפונקציות הללו

פרטו את הנחות.

תשובה 2:

עבור **בנאי ואופרטור ההעתקה ואופרטור ההשמה** אנחנו צריכים בנאי דיפולטי ל T בגלל חוסר הפרמטרים ולכן צריכים אותם עבור כל טיפוס T כך שיעבדו בצורה נכונה.

בנוסף, עבור **הדסרקטור** קיימת ההנחה לטיפוס טמפלייטי הורס.

וגם כן, עבור **אופרטור ההשוואה** (=), קיימת ההנחה לטיפוס טמפלייטי אופרטור ההשמה וההורס.

עבור **pushback**, קיימת ההנחה שעבור הטיפוס הטמפלייטי קיים בנאי העתקה.

עבור **popfront**, קיימת ההנחה שעבור הטיפוס הטמפלייטי קיים הורס.

עבור **filter** and **transform**, קיימת ההנחה לטיפוס טמפלייטי אופרטור ההשמה באופרציה, וההשמה בtransform .

שאילה 3:

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?

תשובה 3:

היא מתרחשת בקישור.

שאילה 4:

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

תשובה 4: נגדיר את functor הבאה:

```
class DivideBy{

public:
    DivideBy(int n):n(n){}
    bool operator()(int number) const
    {
        return (number%n==0);
    }
private:
    int n;
};
```

המימוש הוא על ידי שליחת את DivideBy לפונקצית filter עבור הפרמטר השני והתור בפרמטר בראשון, כך יחזור לנו התור החדש שנדרש.