נושאים מתקדמים בתכנות מונחה עצמים: תרגיל חזרה בשפת C++

תרגיל 1

נתון טיפוס נתונים "מספר רציונלי" המוגדר באמצעות הפרוטוטייפ **rational.h** .

א. הוסיפו למחלקה את שני המרכיבים הבאים **וממשו אותם**:

1. אופרטור המרה למחרוזת מטיפוס `string` (פורמט כרצונכם)
2. בנאי המרה מטיפוס `string` (הניחו פורמט זהה לזה שבחרתם עבור אופרטור ההמרה)

ב. הניחו כי הערך `NaN` אינו קיים בשפה, והציעו מנגנון המטפל **בשגיאה של חלוקה באפס** (או של אתחול מכנה אפס). עדכנו את פרוטוטייפ המחלקה הנתונה בכדי לשלב את המנגנון המוצע והוסיפו קוד במידת הצורך.

ג. **ממשו פעולת העלאה בחזקה של מספר רציונלי** מן הצורה `**`, כלומר, בהינתן מופע רציונלי `r`, הפעולה `r**d` תעלה את הרציונלי בחזקת `d` ותחזיר מספר רציונלי המהווה את התוצאה (על `d` לייצג מספר ממשי מטיפוס כרצונכם).

תרגיל 2

עליכם לממש תכנית המקבלת כקלט שם של קובץ, קוראת את תוכנו (טקסט רצוף ללא הפרדת שורות) וכותבת לערוץ הפלט הסטנדרטי תוכן זה בהיפוך סדר, תוך השמטת מילים שהופיעו מספר פעמים רצוף. לדוגמא, עבור קובץ קלט הכולל את הטקסט הבא,

```
Richard Feynman Feynman 1942 ; John John Nash 1950 ; Lloyd Shapley 1953 1953
```

בהרצת התכנית שלכם עם שם קובץ הקלט הנכון, ייכתב התוכן הבא לפלט הסטנדרטי:

```
1953 Shapley Lloyd ; 1950 Nash John; 1942 Feynman Richard
```

אילוצים —

- עליכם להשתמש באלגוריתם ההעתקה `unique_copy` לשם השמטת המלים המופיעות מספר פעמים; תיעוד בנספח.

הנחות —

- ניתן להניח כי הקובץ הנתון ייפתח בהצלחה, כלומר, אין צורך לבדוק שגיאה מסוג זה.
- הניחו כי מספר המלים בקובץ כולו אינו עולה על 1024 תווים.

תרגיל 3

עליכם לממש גרסה מיוחדת של אלגוריתם ההעתקה `copy`, המוגדר ע"י הפרוטוטייפ הבא:

```
template<class InIter, class OutIter, class Comparator>
void SortCopy( InIter first, InIter last, OutIter result, Comparator cmp);
```

הפונקציה מעתיקה איברים מתחום המקור (מוגדר באמצעות צמד האיטרטורים: `[first,last)`) אל תחום היעד (מתחיל באיטרטור `result`, ואורכו כאורך התחום המקורי).

סדר האיברים בתחום אליו מעתיקה הפונקציה אינו הסדר המקורי, אלא הסדר הנקבע ע"י הקומפרטור `cmp` .

אילוצים/דרישות:

- הטיפוס `InIter` מקיים את הדרישות מ-Input Iterator ב-STL .
- הטיפוס `OutIter` מקיים את הדרישות מ-Output Iterator ב-STL .
- אין להניח כי האיטרטורים הם Random Access (שימו לב כי הפונקציה `std::sort` מניחה כי האיטרטורים שהיא מקבלת הינם Random Access).

הנחות:

- ניתן להמיר את האלמנט הבודד (value_type) של `InIter` לטיפוס הניתן להשמה בערך אליו מצביע `OutIter`, כלומר, הביטוי `*result=*first` חוקי מבחינת הטיפוסים.
- ניתן להמיר את ה-value_type של `InIter` לטיפוס הארגומנט של `Comparator`.

- `[first,last)` הוא תחום חוקי.
- האיטרטור `result` אינו בתחום `[first,last)`.
- יש מספיק מקום לביצוע ההעתקה, כלומר ניתן לקדם את `result` מספר פעמים כמספר האיברים בתחום, מבלי לקבל איטרטור בלתי-חוקי.

א. ממשו את האלגוריתם כפונקציית תבנית

ב. בידקו את מימוש האלגוריתם באמצעות פונקציית `main` על לפחות שני קומפרטורים

נספח

```
OutputIterator std::unique_copy (InputIterator first,
InputIterator last, OutputIterator result);
```

- Copies the elements in the range `[first,last)` to the range beginning at `result`, except consecutive duplicates (elements that compare equal to the element preceding). Only the first element from every consecutive group of equivalent elements in the range `[first,last)` is copied.
  Parameters:
  `first, last`
  Forward iterators to the initial and final positions in a sequence. The range used is `[first,last)`, which contains all the elements between `first` and `last`, including the element pointed by `first` but not the element pointed by `last`.
  `result`
  Output iterator to the initial position of the range where the resulting range of values is stored.

```
OutputIterator copy (InputIterator first, InputIterator last,
OutputIterator result);
```

- Copies the elements in the range `[first,last)` to the range beginning at `result`.
  The function returns an iterator to the end of the destination range (which points to the element following the last element copied).