

### תרגיל בית מס' 3

#### Polymorphism, Templates

(1) יש לכתוב את הממשקים `Comparable` ו `Printable` כאשר `Comparable` מכיל את אופרטורי היחס (`Relational Operators`: `==`, `>`, `<=`, `!=`, `>`, `<`) והממשק `Printable` מכיל את האופרטורים של קלט/פלט.

(2) יש לכתוב מחלקה `Date` המממשת את הממשקים הנ"ל.

(3) יש לכתוב מחלקה מתבוננת `Interval` שהפרמטר לתבנית היא מחלקה המממשת את האופרטורים הנ"ל. כל מופע של המחלקה הזאת הוא אינטרוול פתוח (כלומר, שלא כולל את הקצוות שלו) של ערכים מסוג הפרמטר. לדוגמה:

- המופעים של המחלקה

`Interval<int>`

מייצגים קבוצות של מספרים שלמים כגון (7,18) שהיא קבוצת כל המספרים השלמים שהם גדולים מ 7 וקטנים מ-18. שימו לב שהקבוצה (7,7) היא ריקה.

- המופעים של המחלקה

`Interval<Date>`

מייצגים קבוצות של תאריכים הנמצאים בין שני תאריכים נתונים.

יש לממש את הפונקציות הבאות:

- **isEmpty**: מחזירה ערך אמת אם"מ המופע מייצג קבוצה ריקה<sup>2</sup>.
- **isBefore**: מחזירה ערך אמת אם"מ המופע מייצג קבוצה שכל איבריה קטנים מאברי קבוצה נתונה.
- **isAfter**: מחזירה ערך אמת אם"מ המופע מייצג קבוצה שכל איבריה גדולים מאברי קבוצה נתונה.
- **intersects**: מחזירה ערך אמת אם"מ המופע מייצג קבוצה הנחתכת (חיתוך לא ריק) אם קבוצה נתונה.
- **contains**: מחזירה ערך אמת אם"מ המופע מייצג קבוצה המכילה איבר נתון.
- **אופרטור &&**: מחזיר חיתוך של אינטרוולים

<sup>1</sup> ממשק הוא מושג המתאר מחלקה המכילה רק פונקציות וירטואליות טהורות.

<sup>2</sup> למרות שהקבוצה (7,8) ריקה. יש להתייחס אליה כקבוצה לא ריקה.

- **האופרטור ||** : מחזיר איחוד של אינטרוולים. איחוד של שני אינטרוולים זרים הוא לא אינטרוול. מקרה כזה יחשב לשגיאה.

### דרישות נוספות:

- בכל מצב של שגיאה, יש להדליק דגל בתוך האובייקט ולשמור את סיבת השגיאה בתוך האובייקט. הפונקציות isValid ו getProblem מחזירות את הדגל ואת הסיבה לשגיאה.
- המנעו **מסירחון בקוד** בכלל, ומשכפול קוד ומכתיבת קוד מיותר בפרט.
- יש להשתמש ב const ו references בכל מקום שראוי להשתמש בהם.
- תכנית בדיקה, קלט/פלט לדוגמה נמצאים בסוף התרגיל. בכל פרט שלא הובהר בתאור התרגיל או שיש סתירה בין התאור הנ"ל לדוגמה, הדוגמה קובעת.

## הוראות הגשה

- יש להגיש קובץ ZIP בשם  
1\_HW3\_123456789\_987654321.zip  
(יש להחליף את המספרים עם מספרי ת.ז. של המגישים).
- על הקובץ להכיל את כל קבצי ה h ו cpp שכתבתם בלי התכנית הראשית.
- שימו לב: הקובץ לא צריך להכיל את התיקיה שבה הקבצים נמצאים, אלא רק את הקבצים עצמם.
- רק אחד מהשותפים לתרגיל יגיש אותו ב MOODLE במקרה של הגשה כפולה אחת ההגשות תבדק באופן שרירותי.
- אין להגיש קובץ המכיל את שמות המגישים
- יש לבדוק את התרגיל לפני ההגשה באתר [/http://pyhw.hit.ac.il/homework](http://pyhw.hit.ac.il/homework)
- לאופן השימוש באתר ניתן לצפות בסרטון הזה.
- **תרגיל שלא נבדק באתר, עשוי להיכשל כשלון מוחלט בזמן הבדיקה בפועל ולקבל ציון 0.**  
**לעיתים קרובות הכישלון בגלל פרט טכני קטן (כמו שם הקובץ, תוכן הקובץ וכו') שיכולתם לתקנו בקלות אילו בדקתם אותו באתר לפני ההגשה. הגשת תרגיל ללא בדיקה באתר, על אחראיותכם בלבד.**
- **במקרים של העתקה שני התרגילים יקבלו ציון 0. תזכרו: דיון בין קבוצות שונות על התרגיל ופתרונות אפשריים הוא מותר ואף רצוי, כל עוד לא מסתכלים בקוד של קבוצה אחרת.**

**בהצלחה !!**

## תכנית ראשית

```
#include <iostream>
#include <iomanip>
#include "Date.h"
#include "IComparable.h"
#include "IPrintable.h"
#include "Interval.h"

using namespace std;

void testDate() {
    Date independence(14,5,1948);
    Date otherDate = independence;

    cout << "Independence:" << independence << ", Other: " << otherDate << endl;
    otherDate.setMonth(2);
    cout << "Other date: " << otherDate << endl;
    otherDate.setDay(29);
    cout << "Other date: " << otherDate << endl;
    otherDate.setYear(1947);
    cout << "Other date: " << otherDate << endl;

    otherDate = Date(24,1,1959);
    cout << "Other date: " << otherDate << endl;

    cout << "Comparing using polymorphism" << endl;
    IComparable<Date> *indP = dynamic_cast< IComparable<Date> *> (&independence);
    cout << "Is independence <= otherDate ? " << (*indP <= otherDate) << endl;

    IComparable<Date> *otherP = dynamic_cast< IComparable<Date> *> (&otherDate);
    cout << "Is other date <= independence ? " << (*otherP <= independence) << endl;
}

void testDateInput() {
    Date otherDate(1,1,1);
    do {
        cin >> otherDate;
        cout << otherDate << endl;
    } while (otherDate != Date(1,1,1));
}

void testIntervalInt() {
    Interval<int> interval = Interval<int>(2,1);
    cout<< interval << endl;

    Interval<int> interval13(1,3);
    Interval<int> interval24(2,4);
    Interval<int> interval45(4,5);
    cout << interval13 << endl;
    cout << interval24 << endl;
    cout << interval45 << endl;

    cout << "Does " << interval13 << " contain " << 2 << "? " << interval13.contains(2) << endl;
    cout << "Does " << interval24 << " contain " << 2 << "? " << interval24.contains(2) << endl;
    cout << "Does " << interval45 << " contain " << 2 << "? " << interval45.contains(2) << endl;

    cout << "Is " << interval13 << " before " << interval24 << "? "
        << interval13.isBefore(interval24) << endl;
```

```

cout << "Is " << interval13 << " before " << interval45 << "? "
    << interval13.isBefore(interval45) << endl;
cout << "Is " << interval24 << " before " << interval45 << "? "
    << interval24.isBefore(interval45) << endl;

cout << "Is " << interval13 << " after " << interval24 << "? "
    << interval13.isAfter(interval24) << endl;
cout << "Is " << interval13 << " after " << interval45 << "? "
    << interval13.isAfter(interval45) << endl;
cout << "Is " << interval24 << " after " << interval45 << "? "
    << interval24.isAfter(interval45) << endl;

cout << "Does " << interval13 << " intersect " << interval24 << "? "
    << interval13.intersects(interval24) << endl;
cout << "Does " << interval13 << " intersect " << interval45 << "? "
    << interval13.intersects(interval45) << endl;
cout << "Does " << interval24 << " intersect " << interval45 << "? "
    << interval24.intersects(interval45) << endl;

cout << "Does " << interval24 << " intersect " << interval13 << "? "
    << interval24.intersects(interval13) << endl;
cout << "Does " << interval45 << " intersect " << interval13 << "? "
    << interval45.intersects(interval13) << endl;
cout << "Does " << interval45 << " intersect " << interval24 << "? "
    << interval45.intersects(interval24) << endl;

cout << "interval13 && interval24 = " << (interval13 && interval24) << endl;
cout << "interval13 && interval45 = " << (interval13 && interval45) << endl;
cout << "interval24 && interval45 = " << (interval24 && interval45) << endl;
cout << "interval24 && interval13 = " << (interval24 && interval13) << endl;
cout << "interval45 && interval13 = " << (interval45 && interval13) << endl;
cout << "interval45 && interval24 = " << (interval45 && interval24) << endl;

cout << "interval13 || interval24 = " << (interval13 || interval24) << endl;
cout << "interval24 || interval13 = " << (interval24 || interval13) << endl;
cout << "interval13 || interval45 = " << (interval13 || interval45) << endl;
cout << "interval24 || interval45 = " << (interval24 || interval45) << endl;
cout << "interval45 || interval13 = " << (interval45 || interval13) << endl;
cout << "interval45 || interval24 = " << (interval45 || interval24) << endl;

}

int main() {
    cout << boolalpha << setfill('0');
    testDate();
    cout << endl << endl;
    testIntervalInt();
    cout << endl << endl;
    testDateInput();

    return 0;
}

```

24/1/2020  
30/1/2020  
29/2/2020  
31/3/2020  
-1/1/2019  
32/1/2020  
1/-2/2020  
32/3/2020  
31/4/2020  
15/15/2020  
29/2/2021  
1/1/1  
2/2/2

## פלט

Independence: 14/05/1948, Other: 14/05/1948

Other date: 14/02/1948

Other date: 29/02/1948

Other date: Not a leap year

Other date: 24/01/1959

Comparing using polymorphism

Is independence <= otherDate ? true

Is other date <= independence ? false

Invalid interval

(1, 3)

(2, 4)

(4, 5)

Does (1, 3) contain 2? true

Does (2, 4) contain 2? false

Does (4, 5) contain 2? false

Is (1, 3) before (2, 4)? false

Is (1, 3) before (4, 5)? true

Is (2, 4) before (4, 5)? true

Is (1, 3) after (2, 4)? false

Is (1, 3) after (4, 5)? false

Is (2, 4) after (4, 5)? false  
Does (1, 3) intersect (2, 4)? true  
Does (1, 3) intersect (4, 5)? false  
Does (2, 4) intersect (4, 5)? false  
Does (2, 4) intersect (1, 3)? true  
Does (4, 5) intersect (1, 3)? false  
Does (4, 5) intersect (2, 4)? false  
interval13 && interval24 = (2, 3)  
interval13 && interval45 = EMPTY  
interval24 && interval45 = EMPTY  
interval24 && interval13 = (2, 3)  
interval45 && interval13 = EMPTY  
interval45 && interval24 = EMPTY  
interval13 || interval24 = (1, 4)  
interval24 || interval13 = (1, 4)  
interval13 || interval45 = Invalid interval  
interval24 || interval45 = Invalid interval  
interval45 || interval13 = Invalid interval  
interval45 || interval24 = Invalid interval

31/12/2010

01/01/2020

24/01/2020

30/01/2020

29/02/2020

31/03/2020

Illegal day for month

Illegal day for month

Illegal month

Illegal day for month

Illegal day for month

Illegal month

Not a leap year

01/01/1