תרגול מס' 3
Polymorphism & Smart Pointers

Container של איבר כללי

נרצה לתמוך במערכת המנהלת פגישות זום (ZoomMeeting).
הנתונים של כל פגישה הם:

1. שם הפגישה
2. תאריך
3. זמן התחלה
4. זמן סיום
5. מזהה ייחודי
6. מס' המשתתפים
7. רשימת משתתפים (אחת) כאשר כל משתמש יכול להיות משני סוגים (Host) או (Regular) .
    a. נתוני Host הם:
        i. שם
        ii. האם המיקרופון פעיל
        iii. האם המצלמה פעילה
        iv. האם שיתוף המסך אקטיבי

    b. נתוני Regular הם:
        i. שם
        ii. האם המיקרופון פעיל
        iii. האם המצלמה פעילה
        iv. האם צופה בשיתוף המסך

כעת, המשימה היא לתמוך באופרטורים לקריאה (<<) וכתיבה (>>) של כל נתוני הפגישה.

מבנה הנתונים הגנרי עבור רשימת המשתתפים יהיה מסוג וקטור.

ברשימת המשתתפים כל אובייקט יוחזק ע"י מצביע חכם למחלקת Participant.

ראו את דיאגרמת המחלקות הבאה:

Design

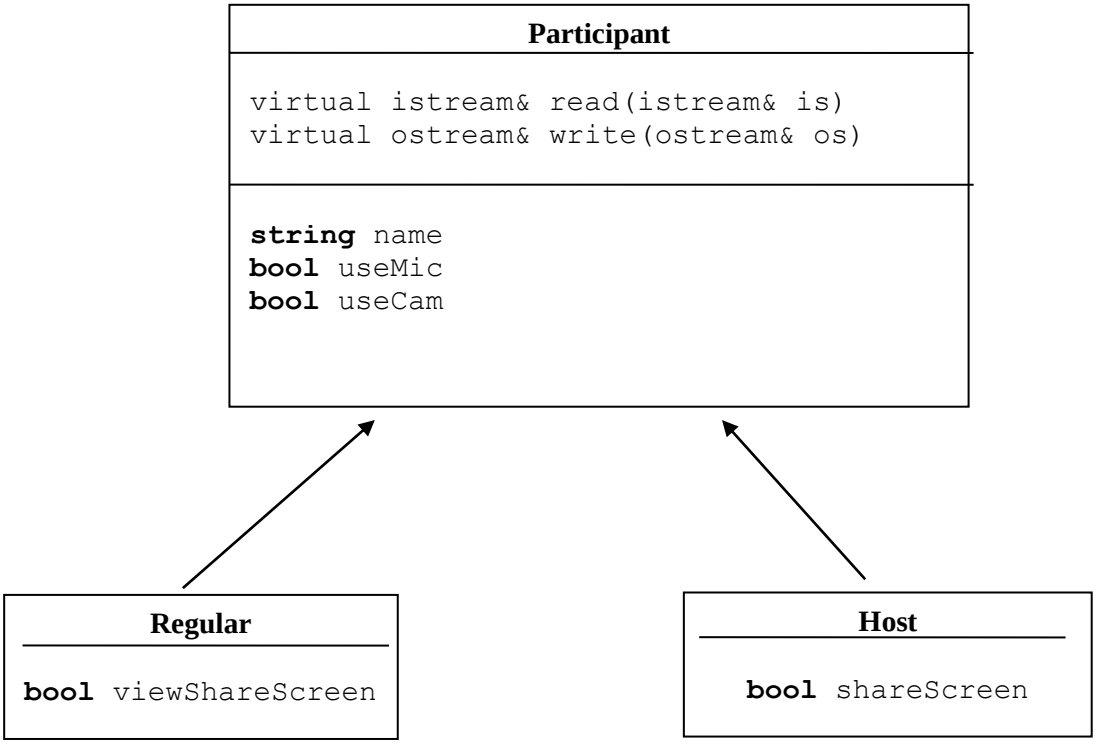
בשלב ראשון, נשים לב שרשימת כל המשתתפים צריכה לשבת ב – container אחד שיכיל בפועל שני טיפוסים. מכאן שה – container יחזיק טיפוס כללי אחד שנקרא Participant שיכול להיות Host או Regular. הטיפוס הזה יהיה base class.

הערה
נשים לב שלאופרטורים << ו - >> לא יכולה להיות התנהגות פולימורפית, משום שהם אינם members של המחלקות שלנו. במקום זה, נספק מתודות שנקראות read ו – write שיעשו קריאה/כתיבה כדרוש, ואופרטורים << ו - >> ידאגו להזעיק אותם בהתאמה. כלומר, הם רק עוטפים את המתודות read/write.

בקריאה או כתיבה של ה – container הזה, יש לתמוך בקריאה/כתיבה של משתתף בודד. Participant יכיל את המתודות הווירטואליות הבאות

read/write – קריאה או כתיבה בפורמט טקסטואלי של מוצר בודד (אופרטורים << , >> ייעזרו במתודות הללו).

בפרט המחלקות הנגזרות יכילו את המתודות הללו.
המתודות הללו יהיו ווירטואליות כדי שהקריאה והכתיבה יתבצעו אוטומטית עבור האובייקט הנכון (Host \ Regular) .
(נשים לב שבפגישה, המתודות הללו לא צריכות להיות ווירטואליות. מדוע?)



ZoomMeeting

```
void read(istream& is)
void write(ostream& os)
void removeParticipant(int idx)
```

```
string name, date, start, end, id
int numOfParticipants

vector<smartPointers<Participant>
Participants;
```