**פתרון מועד X**

**שאלה 1 (40 נקודות):**

לכל פגישת אונליין (OnlineMeeting) נרצה לשמור את פרטי מארגן הפגישה, את נושא הפגישה, ואת רשימת משתתפי הפגישה (מוגבל עד 10 משתתפים) .

להלן תיאור של מחלקות הקיימות במערכת ומחלקות **שעליכם להוסיף או לעדכן**, בהמשך יוצג main לדוגמה ופלט תוצאת הרצתו.

1. נתונה בהמשך באופן **מלא** המחלקה עבור תאריך ושעה (DateTime):
   1. תכונות המחלקה: יום, חודש, שנה, שעה ודקות.
   2. קונסטרקטור המקבל את כל התכונות ומאתחל את האובייקט.
   3. בנאי העתקה (copy c’tor) היודע לייצר אובייקט חדש מנתוני אובייקט אחר.
   4. מתודה toString המחזירה מחרוזת עם נתוני האובייקט כפי שמופיע בפלט בהמשך.
2. נתונה באופן **חלקי** מחלקה עבור משתמש (User):
   1. תכונות המחלקה הן שם המשתמש, סיסמה, מערך הפגישות בהן השתתף (עד מקסימום של 2 פגישות, אבל ערך זה כמובן נבחר אקראית לצורכי בדיקות), וכן לכל משתמש מספר סידורי החל מהמספר 1000. מותר להוסיף תכונות נוספות במידת הצורך.
   2. **(6 נק') כתבו** את קונסטרקטור המחלקה המקבל שם משתמש וסיסמה, ומאתחל את נתוני האובייקט.
   3. נתונה המתודה getName המחזירה את שם המשתמש.
   4. נתונה המתודה clearMeetingsHistory המאפסת את מספר הפגישות בהיסטוריה ל- 0.
   5. **(6 נק') כתבו** את הפעולה isAttendingMeeting המקבלת כפרמטר פגישה ומחזירה true אם הפניה לפגישה זו כבר שמורה בהיסטורית הפגישות הנוכחית של המשתמש, אחרת תחזיר false.
   6. **(6 נק') כתבו** את המתודה addMeeting המקבלת פגישה ומנסה להוסיף אותה לאוסף הפגישות של המשתמש: יש לוודא שהפגישה טרם נוספה למערך הפגישות, שיש מקום במערך הפגישות, וכמובן לא לשכוח לעדכן את המשתמש גם בנתוני הפגישה (הרחיבו קריאה בהמשך). המתודה תחזיר true במידה והצליחה להוסיף את הפגישה, אחרת תחזיר false.
   7. נתונה המתודה toString המחזירה מחרוזת עם נתוני המשתמש.
3. נתונה באופן **חלקי** מחלקה עבור OnlineMeeting:
   1. תכונות המחלקה: המשתמש שהוא מארגן הפגישה, נושא הפגישה, מערך משתתפי הפגישה (פרט למארגן), מתי תהיה הפגישה ואורכה. עבור כל פגישה ניתן לרשום מקסימום 10 משתתפים בנוסף למארגן.
   2. נתונות מתודות get המחזירות תכונות של המחלקה וכן קונסטרקטור המאתחל את האובייקט.
   3. נתונה המתודה isUserAttending המקבלת User ומחזירה true אם הוא כבר אחד ממשתפי הפגישה, אחרת מחזירה false.
   4. **(6 נק')** **כתבו** את המתודה addAttendee המקבלת משתמש ומנסה להוסיף אותו לפגישה: יש לוודא שהמשתמש אינו עדיין חלק מהפגישה, שיש מקום במערך המשתתפים, ובמידת הצלחה להוסיף את הפגישה לאוסף הפגישות של המשתמש. המתודה תחזיר את אחד מערכי ה- enum שמוגדר בראש המחלקה eAddAttendeeStatus.
   5. **(6 נק')** **כתבו** את המתודה toString המחזירה מחרוזת עם נתוני הפגישה.
4. **(10 נק') הוסיפו** בקובץ ה- Program את המתודה getLongMeetings המקבלת מערך פגישות וערך המייצג מספר דקות מינימאלי לפגישה, ומחזירה מערך עם הפניות לכל הפגישות שאורכן לפחות כמו הזמן שהתקבל כפרמטר השני. שימו לב שבמערך המוחזר לא יהיו מקומות ריקים.

להלן main לדוגמה:

**public** **class** Program {

**public** **static** OnlineMeeting[] getLongMeetings

(OnlineMeeting[] allMeeting, **int** longerThan) {

// count long meeting

**int** count = 0;

**for** (**int** i=0 ; i < allMeeting.length ; i++) {

**if** (allMeeting[i].getDuration() >= longerThan)

count++;

}

OnlineMeeting[] res = **new** OnlineMeeting[count];

**for** (**int** i=0, j=0 ; i < allMeeting.length ; i++) {

**if** (allMeeting[i].getDuration() >= longerThan)

res[j++] = allMeeting[i];

}

**return** res;

}

**public** **static** **void** main(String[] args) {

DateTime d1 = **new** DateTime(31, 12, 2021, 19, 0);

DateTime d2 = **new** DateTime(1, 3, 2022, 20, 0);

User u1 = **new** User("u1", "12345");

User u2 = **new** User("u2", "12345");

User u3 = **new** User("u3", "12345");

User u4 = **new** User("u4", "12345");

User u5 = **new** User("u5", "12345");

OnlineMeeting m1 = **new** OnlineMeeting(u1, "Eating cakes", d1, 90);

m1.addAttendee(u2);

m1.addAttendee(u3);

OnlineMeeting m2 = **new** OnlineMeeting(u5, "Playing Soccer", d2, 120);

m2.addAttendee(u1);

m2.addAttendee(u3);

m2.addAttendee(u4);

OnlineMeeting m3 = **new** OnlineMeeting(u5, "Solving Exams",

**new** DateTime(23, 1, 2022, 10, 0), 180);

m3.addAttendee(u1);

m3.addAttendee(u4);

System.***out***.println(m1);

System.***out***.println(m2);

System.***out***.println(m3);

System.***out***.println(u1);

System.***out***.println(u2);

System.***out***.println(u3);

System.***out***.println(u4);

System.***out***.println(u5);

u3.addMeeting(m3);

System.***out***.println(m3);

OnlineMeeting[] allMeetings = {m1, m2, m3};

OnlineMeeting[] longMeetings = *getLongMeetings*(allMeetings, 100);

System.***out***.println("The long meetings:");

**for** (OnlineMeeting m : longMeetings)

System.***out***.println(m.toString());

}

}

להלן פלט עבור main זה:

The meeting 'Eating cakes' organized by u1 and it is on the 31/12/2021 19:00 for 90 minutes.

There are 2 attendees:

1) u2

2) u3

The meeting 'Playing Soccer' organized by u5 and it is on the 1/3/2022 20:00 for 120 minutes.

There are 3 attendees:

1) u1

2) u3

3) u4

The meeting 'Solving Exams' organized by u5 and it is on the 23/1/2022 10:00 for 180 minutes.

There are 2 attendees:

1) u1

2) u4

User u1 [password: 12345, id: 1000] attended these last meetings lately:

1) Eating cakes organized by u1 at the 31/12/2021 19:00

2) Playing Soccer organized by u5 at the 1/3/2022 20:00

User u2 [password: 12345, id: 1001] attended these last meetings lately:

1) Eating cakes organized by u1 at the 31/12/2021 19:00

User u3 [password: 12345, id: 1002] attended these last meetings lately:

1) Eating cakes organized by u1 at the 31/12/2021 19:00

2) Playing Soccer organized by u5 at the 1/3/2022 20:00

User u4 [password: 12345, id: 1003] attended these last meetings lately:

1) Playing Soccer organized by u5 at the 1/3/2022 20:00

2) Solving Exams organized by u5 at the 23/1/2022 10:00

User u5 [password: 12345, id: 1004] attended these last meetings lately:

1) Playing Soccer organized by u5 at the 1/3/2022 20:00

2) Solving Exams organized by u5 at the 23/1/2022 10:00

The meeting 'Solving Exams' organized by u5 and it is on the 23/1/2022 10:00 for 180 minutes.

There are 2 attendees:

1) u1

2) u4

The long meetings:

The meeting 'Playing Soccer' organized by u5 and it is on the 1/3/2022 20:00 for 120 minutes.

There are 3 attendees:

1) u1

2) u3

3) u4

The meeting 'Solving Exams' organized by u5 and it is on the 23/1/2022 10:00 for 180 minutes.

There are 2 attendees:

1) u1

2) u4

**להלן המחלקה המלאה DateTime:**

**public** **class** DateTime {

**private** **int** day, month, year, hours, minutes;

**public** DateTime(**int** day, **int** month, **int** year, **int** hours, **int** minutes) {

**this**.day = day;

**this**.month = month;

**this**.year = year;

**this**.hours = hours;

**this**.minutes = minutes;

}

**public** DateTime(DateTime other) {

**this**(other.day, other.month, other.year, other.hours, other.minutes);

}

**public** String toString() {

**return** day + "/" + month + "/" + year + " " + (hours<10 ? "0" : "") + hours + ":" + (minutes < 10 ? "0" : "") + minutes;

}

}

**השלימו את המחלקה Userעפ"י הנדרש בהגדרות למעלה:**

**public** **class** User {

**public** **static** **final** **int** ***MAX\_MEETINGS\_IN\_HISTORY*** = 2;

**private** String name;

**private** String password;

**private** **int** id;

**private** **static** **int** *idGenerator* = 1000;

**private** OnlineMeeting[] meetingHistory;

**private** **int** numOfMeetings;

**public** User(String name, String password) {

**this**.name = name;

**this**.password = password;

meetingHistory = **new** OnlineMeeting[***MAX\_MEETINGS\_IN\_HISTORY***];

numOfMeetings = 0;

id = ++*idGenerator*;

}

**public** String getName() {

**return** name;

}

**public** **void** clearMeetingsHistory() {

numOfMeetings = 0;

}

**public** **boolean** isAttendingMeeting(OnlineMeeting meeting) {

**for** (**int** i=0 ; i < numOfMeetings ; i++) {

**if** (meetingHistory[i] == meeting)

**return** **true**;

}

**return** **false**;

}

**public** **boolean** addMeeting(OnlineMeeting newMeeting) {

**if** (numOfMeetings == ***MAX\_MEETINGS\_IN\_HISTORY*** || isAttendingMeeting(newMeeting) )

**return** **false**;

meetingHistory[numOfMeetings++] = newMeeting;

newMeeting.addAttendee(**this**);

**return** **true**;

}

**public** String toString() {

String str = "User " + name + " [password: " + password + ", id: " + id + "] attended these last meetings lately:\n";

**for** (**int** i=0, j=1 ; i < numOfMeetings ; i++) {

str += "\t" + (i+1) + ") " + meetingHistory[i].getSubject() + " organized by " + meetingHistory[i].getOrganizer().getName() + " at the " + meetingHistory[i].getWhen() + "\n";

}

**return** str;

}

}

**השלימו את המחלקה OnlineMeetingעפ"י הנדרש בהגדרות למעלה:**

**public** **class** OnlineMeeting {

**public** **enum** eAddAttendeeStatus {***Success***, ***MeetingIsFull***, ***AttendeeAlreadyRegistered***, ***FailedAddingMeetingToUser***};

**public** **final** **static** **int** ***MAX\_ATTENDEES*** = 10;

**private** User organizer;

**private** String subject;

**private** User[] attendees;

**private** **int** numOfAttendees;

**private** DateTime when;

**private** **int** duration;

**public** OnlineMeeting(User organizer, String subject, DateTime when, **int** duration) {

**this**.subject = subject;

attendees = **new** User[***MAX\_ATTENDEES***];

numOfAttendees = 0;

**this**.when = **new** DateTime(when);

**this**.duration = duration;

**this**.organizer = organizer;

**this**.organizer.addMeeting(**this**);

}

**public** String getSubject() {

**return** subject;

}

**public** DateTime getWhen() {

**return** when;

}

**public** User getOrganizer() {

**return** organizer;

}

**public** **int** getDuration() {

**return** duration;

}

**public** **boolean** isUserAttending(User user) {

**if** (organizer == user)

**return** **true**;

**for** (**int** i=0 ; i < numOfAttendees ; i++) {

**if** (attendees[i] == user)

**return** **true**;

}

**return** **false**;

}

**public** eAddAttendeeStatus addAttendee(User user) {

**if** (isUserAttending(user))

**return** eAddAttendeeStatus.***AttendeeAlreadyRegistered***;

**if** (numOfAttendees == ***MAX\_ATTENDEES***)

**return** eAddAttendeeStatus.***MeetingIsFull***;

**if** (user.addMeeting(**this**)) {

attendees[numOfAttendees++] = user;

**return** eAddAttendeeStatus.***Success***;

}

**else** {

**return** eAddAttendeeStatus.***FailedAddingMeetingToUser***;

}

}

**public** String toString() {

String str = "The meeting '" + subject + "' organized by " +   
 organizer.getName() + " and it is on the " + when.toString() +   
 " for " + duration + " minutes.\n";

str += "There are " + numOfAttendees + " attendee" +  
 (numOfAttendees != 1 ? "s" : "") + ":\n";

**for** (**int** i=0 ; i < numOfAttendees ; i++) {

str += "\t" + (i+1) + ") " + attendees[i].getName() + "\n";

}

**return** str;

}

}

**שאלה 2 (25 נקודות):**

**חלק א' (10 נקודות)**

להלן פונקציה רקורסיבית:

**public** **static** **int** foo(**int**[] arr, **int** one, **int** two) {

**if** (one == two)

**return** arr[one];

**if** (one+1 == two)

**return** arr[one]+arr[two];

**int** temp1 = *foo*(arr, one+1, two-1);

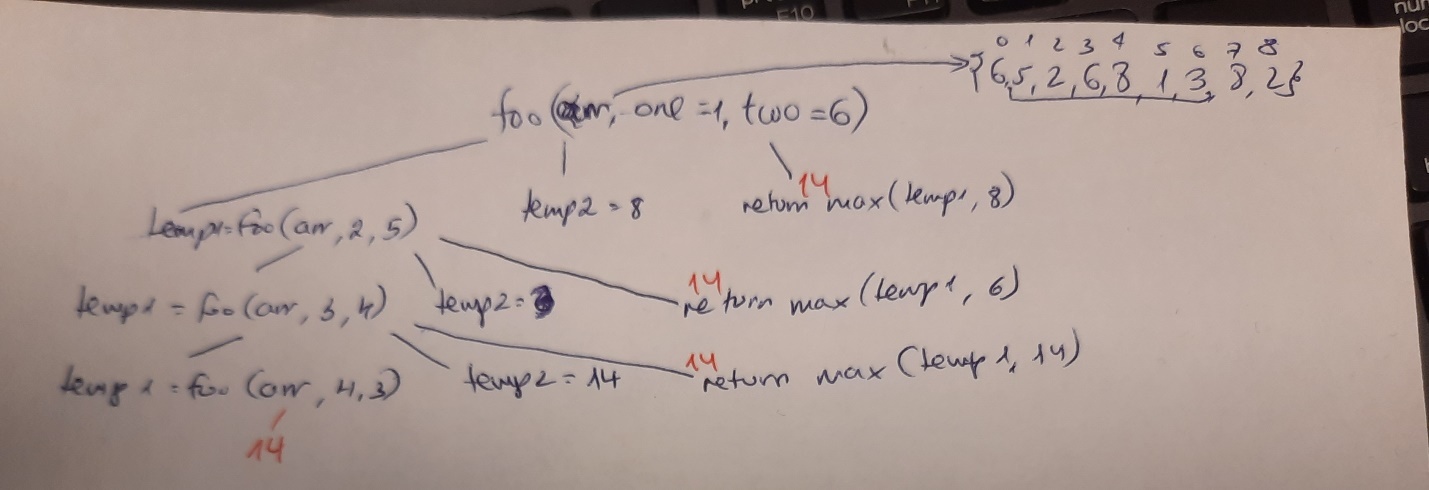
**int** temp2 = arr[one] + arr[two];

**return** Math.*max*(temp2, temp1);

}

1. **(7 נקודות)** ציירו עץ מעקב עבור הפונקציה foo עבור המערך {3,8,26,8,1,,6,5,2} כאשר two=6, one=1 וכתבו מה הפונקציה תחזיר עבור זימון הפונקציה עם ערכים אלו.
2. **(3 נקודות)** כתבו במשפט מה הפונקציה עושה. (שימו לב, אין להסביר מהן הפקודות המבוצעות או לתת דוגמת הרצה - כלומר אין להסביר את ה"איך").

**פתרון:**



הפונקציה מחזירה את ערך הזוג שסכומו מקסימלי. זוג ערכים הוא צמד ערכים במיקום נגדי במערך. לצורך העניין, אם המערך באורך אי-זוגי, האיבר האמצעי נחשב זוג עם ערכו בלבד.

**חלק ב' (15 נקודות)**

כתבו את הפונקציה הרקורסיבית הבאה:

**public** **static** **void** printDigitsAsChars(**int** num)

הפונקציה מקבלת מספר ומדפיסה במקום כל ספרה את התו התואם במיקומו.

כלומר, במקום הספרה 0 יודפס התו A, במקום הספרה 1 יודפס התו B וכו'.

דוגמאות:

זימון הפונקציה עם המספר 123 ידפיס את המחרוזת BCD

זימון הפונקציה עם המחרוזת 9054 ידפיס את המחרוזת JAFE

**פתרון:**

**public** **static** **void** printDigitsAsChars(**int** num) {

**if** (num < 10) {

System.***out***.print((**char**)(num + 'A'));

**return**;

}

*printDigitsAsChars*(num/10);

System.***out***.print((**char**)(num%10 + 'A'));

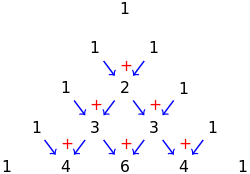
}

**שאלה 3 (35 נקודות):**

**הגדרה:**

**משולש פסקל** הוא סידור של מספרים בצורת משולש, הנבנה באופן הבא: הקודקוד העליון של משולש זה מכיל את המספר 1, וכל מספר במשולש מהווה את סכום שני המספרים שנמצאים מעליו (המספרים שנמצאים על שוקי המשולש הם כולם 1).

דוגמה:



(ההגדרה והתמונה לקוחים מוויקיפדיה).

.

**סעיף א' (15 נק'):**

כתבו את הפונקציה הבאה:

**public** **static** **int**[][] buildPascalValues(**int** rows)

הפונקציה מקבלת מספר המייצג מספר שורות במשולש פסקל, והיא תחזיר מערך דו-מימדי שיש בו rows שורות, ובכל שורה הערכים שצריכים להיות בשורה זו במשולש.

דוגמה:

עבור rows=5, יבנה ויוחזר המערך הדו-מימדי הבא:

[1]

[1, 1]

[1, 2, 1]

[1, 3, 3, 1]

[1, 4, 6, 4, 1]

**סעיף ב' (15 נק'):**

כתבו את הפונקציה הבאה:

**public** **static** **void** printTriangleFromMatrix(**int**[][] values)

הפונקציה מקבלת מערך דו-מימדי המחזיק ערכים של משולש (כלומר בשורה הראשונה ערך אחד, השני שני ערכים, בשלישי שלושה וכו').

הפונקציה תדפיס משולש ממורכז עם הערכים שבמערך הדו-מימדי.

דוגמה:

עבור המערך הדו-מימדי שהוצג בסעיף א', הפונקציה תצייר את המשולש הבא:



**סעיף ג' (5 נק'):**

כתבו את הפונקציה הבאה:

**public** **static** **void** drawPascalTriangle(**int** rows)

הפונקציה מקבלת מספר שורות ומדפיסה משולש פסקל עם מספר שורות זה.

**פתרון:**

**public** **static** **int**[][] buildPascalValues(**int** rows) {

**int**[][] res = **new** **int**[rows][];

**for** (**int** i=0 ; i < res.length ; i++) {

res[i] = **new** **int**[i+1];

res[i][0] = 1;

res[i][res[i].length-1] = 1;

**for** (**int** j=1 ; j < res[i].length-1 ; j++) {

res[i][j] = res[i-1][j-1] + res[i-1][j];

}

}

**return** res;

}

**public** **static** **void** printTriangleFromMatrix(**int**[][] pascal) {

**for** (**int** i=0 ; i < pascal.length ; i++) {

**for** (**int** j=0 ; j < pascal.length-1-i ; j++)

System.***out***.print(" ");

**for** (**int** j=0 ; j < pascal[i].length ; j++)

System.***out***.print(pascal[i][j] + " ");

System.***out***.println();

}

}

**public** **static** **void** drawPascalTriangle(**int** rows) {

**int**[][] mat = *buildPascalValues*(rows);

*printTriangleFromMatrix*(mat);

}