GuyAI
מבית GOOL בגרויות

סוג הבחינה: בחינת גמר ממלכתית
מועד הבחינה: קיץ 2022 מועד א'
סמל השאלון: 97105
קובץ: תשובות סופיות

מבנה נתונים ותכנות מונחה עצמים

הנדסת תוכנה · הנדסאים וטכנאים

הוראות

א. משך הבחינה: ארבע שעות וחצי.

ב. מבנה השאלון ומפתח ההערכה: **בשאלון זה שני מבחנים, עליכם לענות על מבחן אחד בלבד בהתאם למוסד הלימודים:** מבחן ב- Java (עמוד 2) מבחן ב- C# (עמוד 15) בכל מבחן 11 שאלות. **חלק א' – 45 נקודות** שאלות 1-4: יש לענות על **שלוש** שאלות בלבד. ערך כל שאלה 15 נקודות. **חלק ב' – 30 נקודות** שאלות 5-8: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 15 נקודות. **חלק ג' – 25 נקודות** שאלות 9-11: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 12 נקודות. נקודה אחת תינתן על הערכה. **בסך הכול: 100 נקודות.**

ג. חומר עזר מותר בשימוש: 1. מחשבון (אין להשתמש במחשב כף יד או במחשבון עם תקשורת חיצונית). 2. קלסר **אחד בלבד** עם **חומר ההרצאות**. אין להוציא דפים מהקלסר. **אין לצרף ספרים או חוברות עם פתרונות.**

ד. הוראות מיוחדות: 1. יש לקרוא בעיון את ההנחיות בדף השער ואת כל שאלות הבחינה, ולוודא שהן מובנות. 2. את התשובות יש לכתוב בצורה מסודרת, בכתב יד ברור ונקי (גם בכך תלויה הערכת הבחינה). 3. יש להשאיר את העמוד הראשון במחברת הבחינה ריק. בסיום המבחן יש לרשום בעמוד זה את מספרי התשובות לבדיקה. התשובות ייבדקו לפי סדר כתיבתן בעמוד זה. לא ייבדקו תשובות עודפות. 4. יש לכתוב את התשובות במחברת הבחינה **בעט בלבד**, בכתב יד ברור. 5. יש להתחיל כל תשובה בעמוד חדש ולציין את מספר השאלה ואת הסעיף. אין צורך להעתיק את השאלה עצמה. 6. טיוטה יש לכתוב במחברת הבחינה בלבד. יש לרשום את המילה "טיוטה" בראש העמוד ולהעביר עליו קו כדי שלא ייבדק. 7. יש להציג פתרון מלא ומנומק, כולל חישובים לפי הצורך. הצגת תשובה סופית ללא שלבי הפתרון לא תזכה בניקוד. 8. יש להסביר בפירוט כל תוכנית שנכתבה, **תוכנית ללא הסבר מפורט לא תזכה בניקוד.** 9. אם לדעתכם חסר בשאלה נתון, יש לציין זאת ולהוסיף נתון מתאים שיאפשר לכם להמשיך בפתרון השאלה, נמקו את בחירתכם. בשאלון זה 27 עמודי בחינה ו- 4 עמודי נספחים.

השאלות בשאלון זה מנוסחות בלשון רבים, אף על פי כן על כל תלמידה וכל תלמיד להשיב עליהן באופן אישי.

בהצלחה!

תשובות סופיות

שאלה 1

א. בונים שרשרת חדשה: לכל זוג חוליות סמוכות p ו-p.getNext() מוסיפים לסוף השרשרת החדשה חוליה עם |p.getValue() - p.getNext().getValue()|; מחזירים את החוליה הראשונה שלה (null אם בשרשרת המקורית פחות משתי חוליות).

ב. מדפיסים את השרשרת, ובלולאה - כל עוד יש בה יותר מחוליה אחת - מחליפים אותה בשרשרת ההפרשים שלה (סעיף א) ומדפיסים; מחזירים את ערך החוליה היחידה שנשארה (בדוגמה: 1).

שאלה 2

א. מוציאים את האיברים מהראש כלפי מטה ושומרים במחסנית עזר; צוברים את סכום האיברים שהוצאו, וכל איבר (חוץ מהראשון שבראש) חייב להיות גדול מהסכום הזה - אחרת false. בסוף מחזירים את כל האיברים למחסנית המקורית.

ב. $O(n)$ - כל אחד מ-n האיברים מוצא מהמחסנית פעם אחת ומוחזר אליה פעם אחת, וכל פעולה של מחסנית היא $O(1)$.

שאלה 3

array[0] = 0/10, array[0] = 2/12, array[1] = 0.1, array[1] = -3.4, array[2] = 0, array[2] = 4, array[3] = 0.1, array[3] = -5.6 (כל אחד בשורה נפרדת).

שאלה 4

א. כן. ב-B יש העמסה: f(B b) היא פעולה חדשה עם פרמטר מטיפוס אחר מ-f(A a) שירשה מ-A. ב-C יש דריסה: f(A a) של C היא באותה חתימה בדיוק כמו f(A a) של A ומחליפה אותה.

ב. Java: 1 - שגיאת זמן ריצה (ClassCastException); 2 - F in A, false; 3 - F in A, false; 4 - true; 5 - false; 6 - שגיאת קומפילציה. C#: 1 - שגיאת זמן ריצה (InvalidCastException); 2 - F in A, False; 3 - F in A, False; 4 - F in C, True; 5 - F in C, False; 6 - שגיאת קומפילציה.

שאלה 5

א. תכונה אחת: מחסנית ליצנים `Stack<Clown> clowns` (ראש המחסנית = הליצן העליון בפירמידה); הבנאי יוצר מחסנית ריקה.

ב. מוציאים את הליצנים מלמעלה למטה למחסנית עזר; אם ליצן כבד מהליצן שמתחתיו (`c.getWeight() > clowns.top().getWeight()`) - הפירמידה אינה יציבה. בסוף מחזירים את הליצנים ומחזירים את התוצאה (פירמידה ריקה - יציבה).

ג. בודקים ש-isStable() מחזירה true; מעלים את הליצן לראש (push) ובודקים שוב - אם יציבה מחזירים true, אחרת מורידים אותו (pop) ומחזירים false. אם הפירמידה לא יציבה מלכתחילה - false בלי להוסיף.

שאלה 6

א. Point, Shape – בלי פעולות חובה; Circle extends Shape implements IFillable – fill(String); Square extends Shape implements IFillable, ITurnable – fill(String), left(int), right(int); Rectangle extends Square – בלי פעולות חובה (יורשת אותן מ-Square).

ב. Point(int x, int y); Shape(Point point); Square(Point point, double length1); Circle(Point point, double radius).

ג. (1) תקין, אין פלט. (2) שגיאת קומפילציה. (3) שגיאת קומפילציה. (4) שגיאת קומפילציה. (5) שגיאת קומפילציה. (6) שגיאת קומפילציה (בשורה של left). (7) שגיאת קומפילציה (בשורה של left).

שאלה 7

א. false – הערך 10 מופיע בעץ (בבן הימני של השורש).

ב. למשל x = 5 (כל מספר שאינו מופיע בעץ) – what2(bt, 5) מחזירה true.

ג. מחזירה true אם הערך x אינו מופיע באף צומת של העץ bt (ובעץ ריק), ו-false אם x מופיע בעץ.

ד. יודפסו (כל מספר בשורה): 1, 3, 2, 9, 7, 7, 10.

ה. מדפיסה, בסדר תחילי (preorder), את ערכי הצמתים שערכם אינו מופיע באף צומת בתת-העצים שלהם (באף צאצא שלהם).

שאלה 8

א. מוציאים את כל האיברים, צוברים את סכומם ומעבירים אותם לתור עזר; אחר כך מחזירים אותם לתור המקורי באותו סדר ומחזירים את הסכום.

ב. מוציאים את כל האיברים לתור עזר תוך שמירת האיבר האחרון שהוצא – הוא האיבר שבסוף התור; מחזירים את האיברים לתור המקורי ומחזירים את הערך.

ג. עוברים על השרשרת; לכל תור: אם head() אי-זוגי מכניסים לתור התוצאה את sumQueue שלו, אחרת את lastValue שלו; מחזירים את תור התוצאה (בדוגמה: 7, 3, 18).

ד. $O(n \cdot m)$, כאשר n מספר החוליות בשרשרת ו-m מספר האיברים בתור הגדול ביותר; ובמדויק $O(n + N)$, כאשר N סך כל האיברים בכל התורים – כל איבר מוצא ומוכנס פעמיים בלבד.

שאלה 9

א. Appartment היא מחלקת הבסיס; GardenApp ו-Luxary יורשות ממנה; Penthouse יורשת מ-Luxary.

ב. Appartment: owner, floor, numApp, area; GardenApp extends Appartment: gardenArea; Luxary extends Appartment: terraceArea; Penthouse extends Luxary: numTerr, seaView.

ג. COST_APP במחלקה Appartment (כל הדירות משתמשות בו); COST_GARDEN במחלקה GardenApp; COST_TERRACE במחלקה Luxary (ו-Penthouse מקבלת אותו בירושה).

ד. getPrice בכל ארבע המחלקות: Appartment – area·1000; GardenApp – super + gardenArea·50; Luxary – super + terraceArea·300; Penthouse – super של Luxary, ואם אין נוף לים כפול 0.95.

ה. עוברים על המערך ומדפיסים כל דירה שבעליה "FREE" (פנויה) ו-getPrice() שלה קטן או שווה לסכום.

שאלה 10

א. SpeedCamera עם התכונות: code, road, maxSpeed (int), cars – שרשרת חוליות `Node<Integer>` של מספרי הרכבים שעברו את המהירות, ו-numCars – מונה העבירות (תכונה נוספת).

ב. הבנאי מציב את שלושת הפרמטרים בתכונות, ומאתחל cars = null (שרשרת ריקה) ו-numCars = 0.

ג. אם speed > maxSpeed – יוצרים חוליה חדשה עם מספר הרכב בתחילת השרשרת cars ומגדילים את numCars.

ד. SpeedCameraNetwork עם מערך `SpeedCamera[] cameras` בגודל 100 ומונה count של המצלמות שהוצבו בפועל.

ה. אם count < 100 – מציבים את sc בתא cameras[count] ומגדילים את count.

ו. במחלקה SpeedCameraNetwork: עוברים על המצלמות ומדפיסים את מספר הכביש של כל מצלמה שמונה העבירות שלה (getNumCars) גדול מ-200.

ז. ב-SpeedCamera פעולת עזר isCaught(carNum) שמחפשת את המספר בשרשרת cars; ב-SpeedCameraNetwork הפעולה checkCar עוברת על כל המצלמות, מדפיסה את קוד כל מצלמה שבה הרכב נמצא, ומחזירה true אם נמצא לפחות פעם אחת, אחרת false.

שאלה 11

(1) **לא נכונה. דוגמה סותרת: שורש 13, בן שמאלי 10, ולצומת 10 בן ימני 13.**

(2) **לא נכונה. אותה דוגמה סותרת: שורש 13, בן שמאלי 10, ולצומת 10 בן ימני 13 – 13 בתת-העץ השמאלי.**

(3) **לא נכונה. 13 יכול להיות בן ימני של צומת שערכו 13 או פחות – למשל שורש 13 ובנו הימני 13.**

(4) **נכונה. לכל צומת x: בן שמאלי $\leq x$ > בן ימני, ולכן הבן השמאלי קטן ממש מהבן הימני.**

(5) **נכונה. בעץ שרוך שמאלי כל צומת גדול ממש מבנו השמאלי, וסריקה תחילית עוברת מהשורש כלפי מטה – ערכים יורדים ממש.**

(6) **לא נכונה. דוגמה סותרת: שורש 13, בן שמאלי 10, ולצומת 10 בן ימני 20 – what מחזירה true, והסריקה התוכית מדפיסה 10, 20, 13.**

כל מבחני מה"ט עם פתרונות מלאים: www.guyai.co.il/mahat