

# מבוא למדעי המחשב - סמסטר א' תשפ"ג

## עבודת בית מספר 1

### צוות העבודה:

- מרצה אחראית: מיכל שמש.
- מתרגלים אחראים: דור אמזלג, חנה יאיר.

**מועד פרסום:** 30.10.22 בשעה 14:00.

**מועד אחרון להגשה:** 13.11.22 בשעה 23:59.

### הוראות מקדימות:

#### הגשת עבודות בית

1. קראו את העבודה מתחילתה ועד סופה לפני שאתם מתחילים לפתור אותה. ודאו שאתם מבינים את כל המשימות. רמת הקושי של המשימות אינה אחידה: הפתרון של חלק מהמשימות קל יותר, ואחרות מצריכות חקירה מתמטית - שאותה תוכלו לבצע בעזרת מקורות דרך רשת האינטרנט. בתשובות שבהן אתם מסתמכים על עובדות מתמטיות שלא הוצגו בשיעורים, יש להוסיף כהערה במקום המתאים בקוד את ציטוט העובדה המתמטית ואת המקור (כגון ספר או אתר).
2. לעבודה זו מצורף מסמך הדרכה לבדיקה עצמית: **SelfTestingGuidelines.pdf**. קראו אותו בעיון.
3. עבודה זו תוגש ביחידים במערכת המודל. ניתן לצפות בסרטון הדרכה על הגשת העבודה במערכת ה-vpl תחת "קישורים שימושיים" באתר הקורס.
4. במערכת מופיעים קבצי Java עם שמות כגון Task<n>.java, כאשר <n> מציין את מספר המשימה המתאימה לקובץ (לדוגמא, קובץ Java בשם Task2.java מתאים למשימה מספר 2) וקובץ ה-IntegrityStatement.java (הצהרה על יושר אקדמי). אלו הם קבצי השלד אותם עליכם לערוך ולהגיש. עליכם לערוך את הקבצים האלו בהתאם למפורט בתרגיל ולהגישם כפתרון. **אין לשנות את שמות קבצי השלד.**
5. **המלצה על דרך העבודה** - אנו ממליצים לפתוח פרויקט ב-eclipse בשם Assignment1. כשתעבדו, תערכו (לאחר שהורדתם את קבצי השלד וחילצתם אותם לתוך הפרויקט) בתוך הפרויקט את הקבצים: IntegrityStatement, Task1, Task2, Task3a, Task3b, Task4a, Task4b, Task4c, Task4d, Task4e, Task4f בהתאם להוראות המשימה והגישו אותם לפי ההנחיות.

#### הנחיות נוספות

1. בכל קובץ מופיעה שורה המגדירה משתנה אשר יהווה את הפלט של התוכנית בינתן פלט כלשהו. לדוגמה:

```
int ans = 0
```

2. כמובן כי ייתכן וטיפוס המשתנה יהיה שונה כתלות בשאלה. בנוסף ייתכן ויהיו שני פלטים לתוכנית. במקרה כזה יוגדרו שני משתנים כמתואר בהמשך.
3. משתנה זה יהווה את פלט התוכנית, ובהוראות עבודה זו נתייחס אליו בשם זה.
4. עליכם להציב במשתנה זה את פלט התוכנית כך שבסוף ריצת התוכנית שלכם, המשתנה יכיל את ערך הפתרון.
5. שימו לב שבכל הקבצים המסופקים לכם עליכם לכתוב את הפתרון שלכם אך ורק בין שתי השורות המוגדרות ע"י ההערות:

```
// -----write your code BELOW this line only! -----
```

```
// -----write your code ABOVE this line only! -----
```

6. אין לשנות את השורות המסופקות לכם בקבצי השלד, למעט אתחול שונה (במידה ואתם חשים שיש צורך לכך) למשתנה המהווה את פלט התוכנית. אין לשנות שמות משתנים.
7. אין להדפיס למסך דברים נוספים חוץ משורת ההדפסה המסופקת לכם בקובץ. הדפסות נוספות יגררו הורדה בציון. כמו כן, אין לערוך את שורת ההדפסה.
8. עבודות שלא יעברו קומפילציה במערכת או שבהן לא נחתמה הצהרה על יושר אקדמי (משימה 0) יקבלו את ה**ציון 0** ללא אפשרות לערער על כך. אחריותכם לוודא שהעבודה שאתם מגישים עוברת תהליך קומפילציה במערכת (ולא רק ב-eclipse). להזכירכם, תוכלו לבדוק זאת ע"י לחיצה על כפתור



**Evaluate-ה**

9. עבודות הבית נבדקות גם באופן ידני וגם באופן אוטומטי. לכן, יש להקפיד על ההוראות ולבצע אותן במדויק.
10. סגנון כתיבת הקוד ייבדק באופן ידני. יש להקפיד על כתיבת קוד יעיל, ברור, על מתן שמות משמעותיים למשתנים, על הזחות (אינדנטציה), ועל הוספת הערות בקוד המסבירות את תפקידם של מקטעי הקוד השונים. אין צורך למלא את הקוד בהערות מיותרות, אך חשוב לכתוב הערות בנקודות קריטיות, המסבירות קטעים חשובים בקוד. הערות יש לרשום אך ורק באנגלית. כתיבת קוד אשר אינה עומדת בדרישות אלו תגרור הפחתה בציון העבודה.

### עזרה והנחיה

1. לכל עבודת בית בקורס יש צוות שאחראי לה. ניתן לפנות לצוות בשעות הקבלה. פירוט שמות האחראים לעבודה מופיע במסמך זה וכן באתר הקורס, כמו גם פירוט שעות הקבלה.
2. בתגבור שיתקיים בשבוע השני של הסמסטר, נפתור באופן מודרך את משימות 1, 2, 4 כמו כן, אתם יכולים להיעזר בפורום ולפנות בשאלות לחבריכם לכיתה. צוות הקורס עובר על השאלות ונותן מענה במקרה הצורך. שימו לב, **אין לפרסם פתרונות בפורום**.
3. בכל בעיה אישית הקשורה בעבודה (מילואים, אשפוז וכו'), אנא פנו אלינו דרך מערכת הפניות, כפי שמוסבר באתר הקורס.
4. אנחנו ממליצים בחום להעלות פתרון למערכת המודל לאחר כל סעיף שפתרתם. הבדיקה תתבצע על הגרסה האחרונה שהועלתה (בלבד!).

### הערות ספציפיות לעבודת בית זו

1. בעבודה זו 4 משימות ו-10 תתי-משימות וסך הנקודות המקסימלי הוא 100. שימו לב שמספר הנקודות לכל תת-משימה אחיד (10 נקודות למשימה) ואינו מצביע על קושי המשימה.
2. בעבודה זו מותר להשתמש בידע שנלמד עד הרצאה 3 (כולל), וכן עד תרגול 2 (כולל). לא ניתן להשתמש במערכים, מחרוזות, פונקציות, או כל צורת קוד אחרת אשר לא נלמדה בכיתה.
3. בעבודה זו, בתוכניות אותן אתם מגישים, כל המשתנים עבור מספרים שלמים צריכים להיות מטיפוס **int**.
4. בכל המשימות ניתן להניח כי הקלט תקין.
5. בפירוט המשימות שזורים מספר קטעי "העשרה". הם נועדו לתת מוטיבציה לשימושי המשימות שתתכנתו בעבודה זו, ולהרחיב את הידע התיאורטי לגביהם, למי שמעוניין בכך. רובכם תיתקלו במונחים המופיעים בהם בהמשך לימודיכם. סטודנטים החשים שקטעים אלה מעמיסים עליהם יותר מדי מידע בשלב זה, יכולים לדלג עליהם בבטחה. הם אינם הכרחיים לטובת ביצוע העבודה, וניתן להשלימה בצורה מלאה גם ללא העמקה בהם כלל. עם זאת, אנו בטוחים שסטודנטים סקרנים ימצאו בהם עניין רב, ומעודדים אתכם להתעניין ולהפליג בקריאה ככל שתמצאו.

יושר אקדמי

הימנעו מהעתקות! ההגשה היא ביחידים. אם מוגשות שתי עבודות עם קוד זהה או אפילו דומה - זוהי העתקה, אשר תדווח לאלתר לוועדת משמעת. אם טרם עיינתם [בסילבוס הקורס](#) אנא עשו זאת כעת.

**חובה לחתום על הצהרת יושר אקדמי בהתאם להנחיות במשימה 0 !**

## משימות:

יש להגיש את כל השאלות עד התאריך 13.11.22 תחת **עבודת בית 1 - VPL**. עקבו אחרי הוראות ההגשה בסוף העבודה.

### משימה 0 - הצהרה

פתחו את הקובץ `IntegrityStatement.java` וכתבו בו את שמכם ומספר תעודת הזהות שלכם במקום המסומן. משמעות פעולה זו היא שאתם מסכימים וחותמים על הכתוב בהצהרה הבאה:

I, <Israel Israeli> (<123456789>), assert that the work I submitted is entirely my own.

I have not received any part from any other student in the class, nor did I give parts of it for use to others.

I realize that if my work is found to contain code that is not originally my own, a formal case will be opened against me with the BGU disciplinary committee.

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

**שימו לב! עבודות בהן לא תמולא ההצהרה, יקבלו ציון 0.**

### הקדמה: חילוק שלמים ושארית חלוקה

לכל שני מספרים שלמים  $a, b$  כך ש-  $b \neq 0$ , החלק השלם במנה  $\frac{a}{b}$  הוא מספר שלם  $q$  כך ש-  $a = q \cdot b + r$  ו-  $r < b$  שלם. המספר  $r$  נקרא שארית החלוקה של  $a$  ב-  $b$  ומסומנת  $r = a \% b$ .

למשל בעבור  $a = 13$  ו-  $b = 3$  החלק השלם במנה  $\frac{13}{3}$  הוא  $q = 4$  ושארית החלוקה  $r = 1$  כיוון שמתקיים  $13 = 4 \cdot 3 + 1$ .

### משימה 1 - משימת חימום

פתחו את הקובץ `Task1.java` וכתבו בו תכנית אשר קולטת מהמשתמש ארבעה מספרים שלמים  $a, b, q, r$  ומציבה במשתנה `boolean ans` את הערך `true` אם  $a = q \cdot b + r$ ,  $b \neq 0$ ,  $r < b$  ו- `false` אחרת. המספרים ייקלטו בסדר הבא (משמאל לימין):  $a, b, q, r$ . בשאלה זו ניתן להניח כי הקלטים:  $a, b, q, r$  הם מספרים שלמים וכי  $b \geq 0$ . שימו לב, לא ניתן להניח דברים נוספים על הקלט.

דוגמאות:

אם הקלט הוא  $a = 10, b = 4, q = 2, r = 1$  אזי התוכנית תציב במשתנה `ans` את הערך `false`

אם הקלט הוא  $a = 10, b = 0, q = 2, r = 2$  אזי התוכנית תציב במשתנה `ans` את הערך `false`

אם הקלט הוא  $a = 9, b = 3, q = 3, r = 0$  אזי התוכנית תציב במשתנה ans את הערך: true

אם הקלט הוא  $a = 5, b = 7, q = 0, r = 5$  אזי התוכנית תציב במשתנה ans את הערך: true

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

בסוף ריצת התוכנית על המשתנה ans שסיפקנו לכם להכיל את הפתרון.

## משימה 2 – עוד משימת חימום

פתחו את הקובץ Task2.java וכתבו בו תכנית אשר קולטת מהמשתמש שני מספרים שלמים  $a, b$  כך ש  $a \leq b$  ומציבה במשתנה **int ans** מספר שלם  $n$  בתחום  $[a, b]$  אותו היא מגרילה באקראי ובהתפלגות אחידה\*. במילים: המספר  $n$  שיוגרל צריך לקיים  $a \leq n \leq b$ .

\* הדרכה: יש להשתמש בפקודה `Math.random()` המגרילה באקראי ובהתפלגות אחידה מספר  $x$  בתחום החצי פתוח  $[0, 1)$ . במילים:  $x$  שמוחזר ע"י הפקודה מקיים  $0 \leq x < 1$ . המספרים ייקלטו בסדר הבא (משמאל לימין):  $a, b$ . ניתן להניח כי הקלט תקין, כלומר כי  $a, b$  הם מספרים שלמים וכן כי  $a \leq b$ .

### דוגמאות:

אם הקלט הוא  $a = 2, b = 24$  אזי ערך אפשרי שיוצב במשתנה ans יכול להיות: 17

אם הקלט הוא  $a = -4, b = 5$  אזי ערך אפשרי שישתמש במשתנה ans הוא: -4

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

הערה: אתם לא חייבים ליצור משתנה נוסף  $n$ . ניתן להציב את התוצאה ישירות ב-ans. בסוף ריצת התוכנית על המשתנה ans שסיפקנו לכם להכיל את הפתרון.

**משימה 3: חזקות של 2 ושארית חלוקה****משימה 3א:**

פתחו את הקובץ Task3a וכתבו בו תכנית אשר קולטת מהמשתמש מספר שלם אי-שלילי  $n$  ומציבה במשתנה `int ans` את הערך  $2^n$ . זיכרו כי יש להשתמש במשתנים מטיפוס `int` בלבד. על התוכנית לחשב נכונה את החזקות של 2 עבור כל ערך של  $n$  בין 0 ל-30 כולל.

דוגמאות:

אם הקלט הוא  $n = 0$  אז במשתנה `ans` יוצב הערך:

1

אם הקלט הוא  $n = 1$  אז במשתנה `ans` יוצב הערך:

2

אם הקלט הוא  $n = 10$  אז במשתנה `ans` יוצב הערך:

1024

אם הקלט הוא  $n = 31$  אז במשתנה `ans` יוצב הערך:

-2147483648

נסו להבין מדוע.

**שימו לב:** בחלק זה אין להשתמש בספרייה `Math`. עליכם לחשב את  $2^n$  ע"י שימוש בלולאה.

ניתן להניח כי הקלט תקין, כלומר כי  $n$  הוא שלם אי-שלילי בין 0 ל-30 כולל.

בסוף ריצת התוכנית על המשתנה `ans` שסיפקנו לכם להכיל את הפתרון.

**משימה 3ב:**

`int MV = Integer.MAX_VALUE;` מייצג את הערך המקבל מהפקודה

פתחו את הקובץ Task3b.java וכתבו בו תכנית אשר קולטת מהמשתמש שני מספרים שלמים  $n, k$  ומציבה במשתנה

`int ans` את הערך של  $2^n \% k$ , כלומר את שארית החלוקה של  $2^n$  ב- $k$ .

המספרים ייקלטו בסדר הבא (משמאל לימין):  $n, k$ .

דוגמאות:

אם ערכי הקלט הם  $n = 10, k = 54$  אזי הערך שיוצב יהיה:

52

כיוון ש-  $2^{10} = 54 * 18 + 52$

אם בקלט שני הערכים הם  $n = 35, k = 151$  אזי הערך שיוצב יהיה:

32

כיוון ש-  $2^{35} = 151 * 227,547,936 + 32$

יש להניח כי המספרים  $n, k$  הם שלמים אי-שליליים וכי  $1 < k < \sqrt{MV}$  (לידיעתכם,  $\sqrt{MV} \approx 46,000$ ).

על התוכנית לחשב נכונה את  $2^n \% k$  לכל ערך כנ"ל של  $n$  ו- $k$  (בפרט עבור  $n \geq 31$ ).

הדרכת חובה: על מנת לפתור נכונה תרגיל זה גם עבור ערכים גדולים של  $n$ , יש להשתמש בעובדה הבאה:

$$(a \cdot b) \% k = ((a \% k) \cdot (b \% k)) \% k$$

לדוגמה:

$$(6 \cdot 7) \% 5 = 2 = ((6 \% 5) \cdot (7 \% 5)) \% 5$$

בסוף ריצת התוכנית על המשתנה `ans` שסיפקנו לכם להכיל את הפתרון.

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

**משימה 4: במשימה זו נדון בבעיית בדיקת ראשוניות של מספר****להזכירכם:**

- $MV$  מייצג את הערך המתקבל מהפקודה `int MV = Integer.MAX_VALUE;`
- $\sqrt{MV} \approx 46,000$

**!! העשרה:**

בדיקת ראשוניות של מספר, ובפרט מציאת מספר ראשוני בתחום מסוים, הינה בעיה שימושית ביותר. למשל, לטובת בניית סכמות הצפנה. המפורסמת שבהן הינה סכמת ההצפנה *RSA*, המבוססת על קושי בעיית הפירוק לגורמים ראשוניים של מספר גדול. כדי לייצר את המפתח לסכמה, נדרשים שני מספרים ראשוניים מאוד מאוד גדולים. את הבסיס לשיטות למציאת מספרים ראשוניים בכל טווח שנרצה תראו במשימה הקרובה, ואף תממשו אותו בעצמכם!

**משימה 4א: אלגוריתם נאיבי לבדיקת ראשוניות של מספר**

תזכורת:

מספר ראשוני (prime)  $p$  הוא מספר שלם גדול מ-1 אשר מתחלק ללא שארית רק ב-1 ובעצמו. לדוגמה: 2,3,5,7,... מספר פריק (composite) הוא מספר שלם אשר קיים לו מחלק שלם גדול מ-1 השונה מ-1 ומעצמו. לדוגמה: 4,6,8,9,... הנחיה: בכל חלקי המשימה הבאים אין להשתמש בפונקציה *Math.pow* (למעט עבור בדיקות נכונות אשר תכתבו בעצמכם ואינן כלולות בקוד המוגש).

פתחו את הקובץ Task4a.java וכתבו בו תכנית אשר קולטת מהמשתמש מספר שלם  $n$  כך ש-  $1 < n \leq MV$  ומציבה במשתנה **boolean ans** את הערך `true` אם  $n$  ראשוני ו-`false` אחרת. הדרכת חובה: יש לבדוק בלולאה האם קיים ל-  $n$  מחלק שאינו טריוויאלי.

דוגמאות:אם הקלט הוא  $n = 10$  אזי הערך במשתנה `ans` יהיה:`false`אם הקלט הוא  $n = 11$  אזי הערך במשתנה `ans` יהיה:`true`

ניתן להניח כי הקלט תקין, כלומר כי  $n$  הוא שלם  $1 < n \leq MV$ . בסוף ריצת התוכנית על המשתנה `ans` שסיפקנו לכם להכיל את הפתרון.

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

**משימה 4ב: אלגוריתם נאיבי לבדיקת מספר ראשוניים הקטנים או שווים ל- $n$** 

לכל מספר שלם  $n > 1$  נסמן ב-  $\pi(n)$  את מספר המספרים הראשוניים אשר קטנים או שווים ל-  $n$ . לדוגמה:  $\pi(2) = 1$ ,  $\pi(5) = 3$ ,  $\pi(20) = 8$

פתחו את הקובץ Task4b.java וכתבו בו תכנית אשר קולטת מהמשתמש מספר שלם  $n$  ומחשבת את  $\pi(n)$ . על הפתרון להיות מוכלל במשתנה `int ans` שסיפקנו לכם. ניתן להניח כי הקלט תקין, כלומר כי  $n$  הוא מספר שלם ו-  $n \leq MV$ .

דוגמאות:

אם הקלט הוא  $n = -10$  אז הערך שיוכל ב `ans` יהיה:

0

אם הקלט הוא  $n = 0$  אז הערך שיוכל ב `ans` יהיה:

0

אם הקלט הוא  $n = 2$  אז הערך שיוכל ב `ans` יהיה:

1

אם הקלט הוא  $n = 5$  אז הערך שיוכל ב `ans` יהיה:

3

אם הקלט הוא  $n = 20$  אז הערך שיוכל ב `ans` יהיה:

8

סיימתם חלק זה? כל הכבוד! העלו את הגרסה האחרונה של עבודתכם למערכת המודל.

**לאן אנחנו הולכים:**

סעיף 4ב נותן את התחושה שיש "לא מעט" מספרים ראשוניים. סעיף 4א נותן כלי (נאיבי ולא יעיל) לבדיקה האם מספר הוא ראשוני או לא. משני הסעיפים הנ"ל ניתן לחשוב על הרעיון העקרוני הבא כדי למצוא מספר ראשוני בתחום מסוים: נגריל אקראית מספר בתחום המבוקש. נבדוק אם הוא ראשוני. אם כן, מצאנו. אם לא, נגריל שוב ושוב, עד שנמצא ראשוני. מכיוון שיש "לא מעט" מספרים ראשוניים, כנראה תוך מספר סביר של הגרלות נמצא מספר ראשוני. הבעיה היא שהבדיקה שראינו בסעיף 4א, דורשת בדיקה שיטתית מספר מספר, מה שיהיה לא ישים עבור מספרים מאוד מאוד גדולים. לכן, אנחנו זקוקים לדרך אחרת ויעילה יותר לבדיקה האם מספר הוא ראשוני. דרך כזו נממש בסעיפים הבאים.

**!! העשרה:**

ה"תחושה" שקיבלנו מסעיף 4ב על קיום "לא מעט" מספרים ראשוניים, היא למעשה טעימה ממשפט מוכח לגבי צפיפותם של המספרים הראשוניים בתוך המספרים הטבעיים. משפט זה נקרא משפט המספרים הראשוניים (*Prime Number Theorem*). סטודנטים המעוניינים בכך מוזמנים לחפש את המשפט ברשת ולקרוא עליו.



**משימה 4ג: הכרות עם המשפט הקטן של פרמה (Fermat's little theorem)**

המשפט הקטן של פרמה (באחת מגרסאותיו) קובע את העובדה הבאה:  
 לכל מספר ראשוני  $p$ , ולכל מספר טבעי  $a$  כך ש- $0 < a < p$ , מתקיים  $a^{p-1} \% p = 1$ .  
 במילים: שארית החלוקה של  $a^{p-1}$  ב- $p$  היא 1.  
 משפט זה מוכח מתמטית, וניתן להסתמך על נכונותו בהמשך העבודה.

מהמשפט נובעת בדיקת הראשוניות הבאה:  
 נניח שאנו רוצים לבדוק האם מספר טבעי מסוים  $n$  הוא ראשוני או לא. אם נמצא  $0 < a < n$ , כך ש- $a^{n-1} \% n \neq 1$ ,  
 נדע **בוודאות** ש- $n$  איננו ראשוני. ( $a$  שכזה נקרא "עד", מכיוון שהוא מעיד על פריקותו של  $n$ ).  
 שימו לב, שאם מצאנו  $0 < a < n$ , כך ש- $a^{n-1} \% n = 1$ , זה לא יגיד לנו כלום **בוודאות** על  $n$ .

במשימה זו נדגום את נכונות המשפט, וננסה להבין מה ניתן או לא ניתן להגיד על  $a^{n-1} \% n$  כאשר  $n$  פריק.

פתחו את הקובץ Task4c.java וכתבו בו תכנית אשר קולטת מהמשתמש שני מספרים שלמים  $n, a$ , ומציבה במשתנה **boolean ans** את הערך `true` אם  $a^{n-1} \% n = 1$  ו-`false` אחרת.  
 המספרים ייקלטו בסדר הבא (משמאל לימין):  $n, a$ .  
 יש להניח כי המספרים  $n, a$  הם שלמים אי-שליליים וכי  $1 < a < n < \sqrt{MV}$ .

דוגמאות:

אם ערכי הקלט הם  $n = 5, a = 3$  אז הערך במשתנה `ans` יהיה:  
`true`, מכיוון ש- $3^4 \% 5 = 1$ .

אם ערכי הקלט הם  $n = 6, a = 2$  אז הערך במשתנה `ans` יהיה:  
`false`, מכיוון ש- $2^5 \% 6 = 2 \neq 1$ .

אם ערכי הקלט הם  $n = 31, a = 4$  אז הערך במשתנה `ans` יהיה:  
`true`, מכיוון ש- $4^{30} \% 31 = 1$ . (למעשה, מכיוון ש-31 הוא ראשוני, מנכונות המשפט הקטן של פרמה אנחנו מצפים לקבל `true` עבור כל ערך של  $a$ )

אם ערכי הקלט הם  $n = 57, a = 3$  אז הערך במשתנה `ans` יהיה:  
`false`, מכיוון ש- $3^{56} \% 57 \neq 1$ .  
 (חשבו: מה תוצאה זו אומרת לנו לגבי פריקותו/ראשוניותו של המספר 57?)

אם ערכי הקלט הם  $n = 57, a = 20$  אז הערך במשתנה `ans` יהיה:  
`true`, מכיוון ש- $20^{56} \% 57 = 1$ .  
 (בדוגמה זו,  $a = 20$  מכונה "שקרן", מכיוון שהוא מקיים את התנאי הנבדק, על-אף שהמספר 57 איננו ראשוני)

אם ערכי הקלט הם  $n = 99, a = 10$  אז הערך במשתנה `ans` יהיה:  
`true`, מכיוון ש- $10^{98} \% 99 = 1$ .  
 (חשבו: האם ומה תוצאה זו אומרת לנו לגבי פריקותו/ראשוניותו של המספר 99?)

**משימה 4: כמה חזרות צריך כדי למצוא "עד"?**

במשימה הקודמת הבנו שאם נמצא עבור מספר טבעי  $n$ , "עד" (כלומר, מספר  $a$  כך ש  $a \neq 1 \pmod{n}$ ), נדע להגיד בוודאות ש- $n$  פריק. בנוסף הבנו, שאם  $n$  ראשוני לא נוכל למצוא עבורו אף "עד".  
 על-פניו, אם נרצה לבדוק האם מספר טבעי הוא ראשוני, נוכל לרוץ על כל המספרים הקטנים ממנו ולבדוק אם הם מקיימים עבורו את תנאי ה"עדות". אם הוא פריק, בטוח נמצא "עד" כלשהו. אם הוא ראשוני, בטוח לא נמצא אף "עד".  
 אבל מעבר שכזה על כל המספרים לא נותן לנו שום יתרון על-פני האלגוריתם הנאיבי שמימשנו בסעיף 4א. עדיין, עבור מספרים גדולים מאוד (מאוד) זו תהיה שיטה לא ישימה.

במשימה זו, נבין שאנו לא באמת צריכים לבדוק את כל האפשרויות, ושמספיקות לנו בדיקות מדגמיות בודדות, כדי להכריע האם מספר טבעי הוא ראשוני או לא, ברמת סמך גבוהה מאוד.

פתחו את הקובץ Task4d.java וכתבו בו תכנית אשר קולטת מהמשתמש מספר שלם פריק  $n$ , מגרילה מספר שלם  $a$  בתחום  $[1, n-1]$  שוב ושוב, עד אשר מתקיים  $a \neq 1 \pmod{n}$ , ומציבה במשתנה `int ans` את מספר ההגרלות שבוצעו. במילים: המשתנה `ans` מציין כמה פעמים היינו צריכים להגריל מספר באקראי כדי למצוא "עד" לפריקותו של  $n$ . יש להניח כי הקלט  $n$  הוא שלם אי-שלילי פריק וכי  $1 < n < \sqrt{MV}$ .

**דוגמאות:**

באופן כללי, לרוב לא נוכל לחזות בוודאות מה הערך שנקבל, מכיוון שהוא תלוי בתוצאות ההגרלות האקראיות, אך נוכל לחזות מה סביר שיהיה הפלט. בנוסף, מכיוון שכל הרצה יכולה לספק תוצאה שונה, מומלץ להריץ את התכנית מספר פעמים (לפחות 10) על אותו קלט ולראות מה נקבל.

אם הקלט הוא  $n = 33$  אז סביר שברוב ההרצות הערך במשתנה `ans` יהיה:

1

בחלק קטן של ההרצות אולי נקבל 2 או אף יותר מכך, אך ההסתברות לכך נמוכה. זאת מכיוון שקיימים מעט מאוד "שקרנים" עבור המספר 33. זה המצב עבור רוב מוחלט של המספרים הפריקים.

אם הקלט הוא  $n = 91$  אז סביר שברוב ההרצות הערך במשתנה `ans` יהיה:

בין 1 ל-3

בחלק קטן של ההרצות אולי נקבל אף יותר מכך, אך ההסתברות לכך נמוכה. 91 הוא מספר פריק עבורו קיימים יותר "שקרנים" באופן יחסי. מספרים כאלה נדירים. וגם עבורו, תיווכחו לגלות שמספיק מספר קטן מאוד של הגרלות כדי למצוא "עד" לפריקותו.

שימו לב! אם תריצו את התכנית עם קלט ראשוני (בניגוד להנחיות השאלה!), התכנית תכנס ללופ אינסופי, ותצטרכו ללחוץ על כפתור ה-stop האדום באקליפס כדי לעצור את ריצתה. זאת מכיוון שעבור מספר ראשוני לא נוכל למצוא אף "עד"...

(זהו בדיוק המשפט הקטן של פרמה).

**!! העשרה:**

נסו להריץ את התכנית על הקלטים 6601 ו-8911 (לפחות 10 פעמים על כל קלט).  
 מה קיבלתם?  
 סביר שעבור הקלטים האלה תקבלו יותר פעמים תוצאות קצת יותר גדולות מ-1 או 2.  
 הסיבה לכך היא שקלטים אלה הם דוגמאות למספרים (נדירים מאוד) הנקראים **מספרי קרמייקל**.  
 מספרי קרמייקל הם מספרים **פריקים** עבורם קיימים הרבה "שקרנים".  
 למעשה, עבור מספר קרמייקל  $n$ , כל  $a$  חזר ל- $n$  (כלומר,  $\gcd(n, a) = 1$ ), מקיים  $a^{n-1} \% n = 1$ .  
 סטודנטים המעוניינים בכך מוזמנים לחפש את המונח מספר קרמייקל (*Carmichael number*) ברשת ולקרוא עליו.

**משימה 4: בדיקת ראשוניות של מספר**

בהינתן מספר  $n$  נרצה לבדוק האם הוא ראשוני, בדרך יעילה יותר מהאלגוריתם הנאיבי אותו מימשתם בסעיף 4א.  
 נעשה זאת באמצעות המשפט הקטן של פרמה והרעיונות שגילינו בסעיפים הקודמים.

פתחו את הקובץ Task4e.java וכתבו בו תכנית אשר קולטת מהמשתמש מספר שלם  $n$  ומציבה במשתנה **boolean ans** את הערך `true` אם  $n$  ראשוני ו-`false` אחרת.

יש להניח כי המספר  $n$  הוא שלם אי-שלילי וכי  $1 < n < \sqrt{MV}$ .

**חובה** לממש את בדיקת הראשוניות **על-פי האלגוריתם הבא בדיוק**:

1. קלוט מהמשתמש מספר טבעי  $1 < n < \sqrt{MV}$ .

2. בצע לכל היותר 5 פעמים:

א. הגרל מספר שלם  $a$  בתחום  $[1, n - 1]$ .

ב. אם  $a^{n-1} \% n \neq 1$ ,

i. הכרז כי  $n$  פריק (הצב במשתנה `ans` את הערך `false`).

3. אם כל 5 ה- $a$  שהוגרלו קיימו  $a^{n-1} \% n = 1$ , הכרז כי  $n$  ראשוני (הצב במשתנה `ans` את הערך `true`).

**דוגמאות:**

אם הקלט הוא  $n = 31$  אז הערך במשתנה `ans` יהיה **בוודאות**:

`true`, מכיוון ש-31 הוא ראשוני, ואין סיכוי שבאחת ההגרלות נמצא "עד" לפריקותו (כי לא קיים כזה).

אם הקלט הוא  $n = 33$  אז הערך במשתנה `ans` יהיה **בהסתברות גבוהה מאוד**:

`false`, מכיוון ש-33 הוא פריק, והסיכוי שבחמש הגרלות לא נמצא "עד" לפריקותו הוא נמוך מאוד (כפי שראינו בסעיף 4ד).

אם הקלט הוא  $n = 99$  אז הערך במשתנה `ans` יהיה **בהסתברות גבוהה מאוד**:

`false`, מכיוון ש-99 הוא פריק, והסיכוי שבחמש הגרלות לא נמצא "עד" לפריקותו הוא נמוך מאוד.

אם הקלט הוא  $n = 9547$  אז הערך במשתנה `ans` יהיה **בוודאות**:

`true`, מכיוון ש-9547 הוא ראשוני, ואין סיכוי שבאחת ההגרלות נמצא "עד" לפריקותו (כי לא קיים כזה).

**!! העשרה:**

זהו **אלגוריתם אקראי**, דבר הגורר אפשרות שתוחזר תשובה שגויה:

1. אם הכרזנו כי  $n$  פריק, אזי אנו יודעים בוודאות כי הוא פריק (מכיוון שמצאנו עבורו "עד").
2. אם הכרזנו כי  $n$  ראשוני, קיימים שני מצבים אפשריים:
  - $n$  אכן ראשוני. אז החזרנו תשובה נכונה.
  - $n$  פריק. אז טעינו. זה עלול לקרות אם במשך 5 הגרלות ברצף הגרלנו "שקרנים" - מספרים המקיימים את תנאי פרמה עבור  $n$ , על אף ש- $n$  פריק. כפי שראינו בסעיף 4ד, הסיכוי לכך הוא נמוך מאוד (מאוד מאוד).

כמו-כן, ככל שנגדיל את מספר החזרות, אפילו במעט, מ-5 ל-10 או ל-20 חזרות, נקטין את ההסתברות לשגיאה באופן משמעותי מאוד.

אלגוריתמים אקראיים הם לרוב יעילים הרבה יותר מאלגוריתמים דטרמיניסטיים, כפי שנוכחנו לדעת במקרה זה. בשימוש באלגוריתם אקראי אנחנו "משלמים" בכך שקיימת הסתברות לשגיאה, אך באמצעות ניתוח הסתברותי מסודר, וחזרה על התהליך "מספיק" פעמים (בהתאם לניתוח ולסף השגיאה הרצוי), אנו יכולים להוריד את ההסתברות לשגיאה לערך זניח.

סטודנטים המעוניינים בכך מוזמנים לחפש את המונחים אלגוריתם אקראי (Randomized algorithm) ואלגוריתם דטרמיניסטי (Deterministic algorithm) ברשת ולקרוא עליהם.

**משימה 14: מציאת מספר ראשוני בתחום מסוים**

כעת אנחנו מוכנים למשימה הסופית לעבודה זו!!

פתחו את הקובץ Task4f.java וכתבו בו תכנית אשר קולטת מהמשתמש שני מספרים שלמים  $x, y$  ומציבה במשתנה **int ans** מספר ראשוני כלשהו בתחום  $[x, y]$ .

המספרים ייקלטו בסדר הבא (משמאל לימין):  $x, y$ .

יש להניח כי המספרים  $x, y$  הם שלמים אי-שליליים, כי  $1 < x < y < \sqrt{MV}$ , וכי קיים ראשוני בתחום  $[x, y]$ .

את משימה זו עליכם לממש באופן הבא:

הגרילו באקראי מספר בתחום  $[x, y]$ , בדקו אם הוא ראשוני על פי האלגוריתם שמימשתם במשימה 4ה. אם כן, מצאתם ראשוני. אם לא, חזרו על התהליך שוב ושוב, עד אשר תמצאו ראשוני.

**דוגמאות:**





אם ערכי הקלט הם  $y = 10, x = 2$  אז הערך במשתנה ans יהיה אחד מבין:

2 או 3 או 5 או 7

אם ערכי הקלט הם  $y = 2022, x = 2000$  אז הערך במשתנה ans יהיה אחד מבין:

2003 או 2011 או 2017

הוראות הגשה:

1. גשו ל- [עבודת בית 1 – VPL](#) באתר הקורס.
2. גשו ללשונית Edit.
3. לחצו על הכפתור ה- .
4. יפתחו לכם עוד אופציות, בין היתר אופציה של  upload לחצו על הכפתור ובחרו את הקבצים שערכתם בפרויקט Assignment1. **ודאו כי לא חסרים קבצים וכי הקבצים שהעליתם הם הקבצים המעודכנים ביותר.**
5. שמרו את השינויים (יש ללחוץ על כפתור השמירה) .
6. לחצו על Evaluate .
7. אתם אמורים לקבל פידבק עבור הצלחתכם בבדיקות החלקיות שרצות בזמן הגשה זו (בדיקות נוספות יתבצעו בתום תאריך ההגשה).
8. אנו חוזרים ואומרים, זו אחריותכם לבדוק שהקבצים שהגשתם עוברים תהליך קומפילציה במערכת. עבודות שלא יתקמפלו יקבלו את הציון 0.

**!! העשרה:**

סטודנטים המתעניינים בקריאה נוספת מוזמנים לקרוא על המושגים הבאים (בהם תיתקלו במהלך לימודיכם):

1. מספר ראשוני – *Prime Number*
2. פירוק מספר לגורמים ראשוניים – *Finding Factors of a Number*
3. חשבון מודולרי (חשבון קונגרוואנציות) – *Modular Arithmetic*
4. אלגוריתם דטרמיניסטי – *Deterministic Algorithm*
5. אלגוריתם אקראי – *Randomized Algorithm*
6. משפט המספרים הראשוניים – *Prime Number Theorem*
7. המשפט הקטן של פרמה – *Fermat's Little Theorem*
8. מבחן הראשוניות של פרמה – *Fermat primality test*
9. מספר קרמייקל – *Carmichael number*

# בהצלחה !