



שאלה 1

IsAscending — האם ספרות המספר ממוינות בסדר עולה?

דרישות

- הפעולה `IsAscending(int num)` תחזיר `true` אם ספרותיו של `num` ממוינות בסדר עולה, אחרת `false`.
- אין להשתמש בהמרה למחרוזת.
- נדרש מימוש רקורסיבי (מותר להשתמש בפעולת עזר רקורסיבית עם פרמטר נוסף).
- אם הקלט שלילי, נבחן את הספרות של הערך המוחלט.

הסבר מלא

נפרק את המספר באמצעות פעולות חילוק ושליפה של ספרת היחידות. נשווה בין הספרה הימנית ביותר לספרה שלפניה, ונמשיך רקורסיבית על יתר הספרות.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

- אם $|n| < 10 \leftarrow$ החזירו `true`.
- הוציאו שתי ספרות ימניות: $d_1 = n \bmod 10, d_2 = \lfloor n/10 \rfloor \bmod 10$.
- אם $d_2 > d_1 \leftarrow$ החזירו `false`.
- המשיכו רקורסיבית עם $\lfloor n/10 \rfloor$.

קוד (C#)

```
public static bool IsAscending(int num)
{
    num = Math.Abs(num);
    if (num < 10) return true;
    int d1 = num % 10;
    int d2 = (num / 10) % 10;
    if (d2 > d1) return false;
    return IsAscending(num / 10);
}
```

שאלות הבנה וחשיבה

1. מדוע די לבדוק זוגות סמוכים מימין לשמאל?
2. מה קורה אם יש ספרות שוות? נמקו מדוע הן מותרות/אסורות לפי ההגדרה.

דוגמאות קלט/פלט ותנאי קצה

- $\text{IsAscending}(5) \rightarrow \text{true}$
- $\text{IsAscending}(1234559) \rightarrow \text{true}$
- $\text{IsAscending}(132) \rightarrow \text{false}$
- $\text{IsAscending}(-11239) \rightarrow \text{true}$ (בודקים ערך מוחלט)

שאלה 2

IsEven — האם מספר זוגי? (ללא שימוש ב- %)

דרישות

- הפעולה $\text{IsEven}(\text{int } n)$ תחזיר true אם n זוגי, אחרת false.
- אסור להשתמש באופרטור % (מודולו).
- הפיתרון יהיה רקורסיבי.
- תמיכה במספרים שליליים.

הסבר מלא

$$\text{isEven}(n) = \begin{cases} \text{true}, & n = 0 \\ \text{isEven}(|n| - 2), & |n| > 0 \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n = 0$ אז $\text{true} \leftarrow$
2. אם $n = \pm 1$ אז $\text{false} \leftarrow$
3. אם $n > 0$ קראו רקורסיבית ל- $\text{IsEven}(n - 2)$.
4. אם $n < 0$ קראו רקורסיבית ל- $\text{IsEven}(n + 2)$.

קוד (C#)

```
public static bool IsEven(int n)
{
    if (n == 0) return true;
    if (n == 1 || n == -1) return false;
```

```

if (n > 0)
    return IsEven(n - 2);
else
    return IsEven(n + 2);
}

```

שאלות הבנה וחשיבה

1. מה יקרה עבור ערכים גדולים מאוד ברקורסיה??

דוגמאות ותנאי קצה

- $\text{IsEven}(1) \rightarrow \text{false}$, $\text{IsEven}(0) \rightarrow \text{true}$
- $\text{IsEven}(99999) \rightarrow \text{false}$, $\text{IsEven}(-14) \rightarrow \text{true}$

שאלה 3

IsPrime — האם מספר ראשוני?

דרישות

- הפעולה $\text{IsPrime}(\text{int num})$ תחזיר true אם num ראשוני, אחרת false.
- אין לבדוק מחלקים מעל \sqrt{n} .
- מימוש רקורסיבי בעזרת פעולת עזר הבודקת מחלק הולך וגדל.

הסבר מלא

נגדיר פעולת עזר $\text{prime}(n, d)$ שבודקת שאין מחלקים בטווח d, \dots, \sqrt{n} .

$$\text{prime}(n, d) = \begin{cases} \text{false}, & n \leq 1 \\ \text{true}, & d > \sqrt{n} \\ \text{false}, & n \bmod d = 0 \\ \text{prime}(n, d + 1), & \text{אחרת} \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n \leq 1 \leftarrow \text{false}$.
2. בדקו מחלקים מ-2 ועד \sqrt{n} .
3. אם נמצא d כך ש- $n \bmod d = 0 \leftarrow \text{false}$, אחרת true.

```
public static bool IsPrime(int n)
{
    if (n <= 1) return false;
    return IsPrimeHelper(n, 2);
}
private static bool IsPrimeHelper(int n, int d)
{
    if ((long)d * d > n) return true;
    if (n % d == 0) return false;
    return IsPrimeHelper(n, d + 1);
}
```

שאלות הבנה וחשיבה

1. למה מספיק לבדוק עד \sqrt{n} ?
2. כיצד ניתן להאיץ: דילוג על מספרים זוגיים, טבלאות ראשוניים וכד'?

דוגמאות ותנאי קצה

- $\text{IsPrime}(2) \rightarrow \text{true}$, $\text{IsPrime}(1) \rightarrow \text{false}$
- $\text{IsPrime}(97) \rightarrow \text{true}$, $\text{IsPrime}(49) \rightarrow \text{false}$

שאלה 4

IsPalindrome — האם מספר פלינדרום (ללא המרה למחרוזת)?

דרישות

- הפעולה $\text{IsPalindrome}(\text{int num})$ תחזיר true אם המספר פלינדרום, אחרת false .
- אין להמיר למחרוזת.
- מימוש רקורסיבי ע"י השוואת ספרה ראשונה לאחרונה וצמצום פנימה.

הסבר מלא

האלגוריתם משווה בכל שלב בין הספרה הראשונה לספרה האחרונה במספר. אם הן שונות \leftarrow המספר אינו פלינדרום. אם הן שוות \leftarrow קוראים רקורסיבית לתת-המספר שבאמצע (לאחר הסרת שתי הספרות הקיצוניות). כאשר נותר ספרה אחת או שהמספר התרוקן לגמרי — זהו פלינדרום.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם (השוואת קצוות)

1. חשב את עשר בחזקת (מספר ספרות - 1) כדי להגיע לספרה הראשונה.
2. השווה בין הספרה הראשונה לספרה האחרונה.
3. אם שונות ← החזר false.
4. אם שוות ← הסר את שתי הספרות ובדוק את המספר שנותר.
5. אם נותרת ספרה אחת או אפס ספרות ← החזר true.

קוד (C#)

```
public static bool IsPalindrome(int num)
{
    int n = Math.Abs(num);
    return IsPalindromeHelper(n, Digits(n));
}

private static bool IsPalindromeHelper(int n, int digits)
{
    if (n < 10) return true; // ספרה אחת
    int power = (int)Math.Pow(10, digits - 1);
    int first = n / power;
    int last = n % 10;
    if (first != last) return false;
    int middle = (n % power) / 10; // הסרת ספרה ראשונה ואחרונה
    return IsPalindromeHelper(middle, digits - 2);
}

private static int Digits(int n)
{
    if (n == 0) return 1;
    return (int)Math.Floor(Math.Log10(n)) + 1;
}
```

שאלות הבנה וחשיבה

1. מדוע אנו מקטינים את מספר הספרות ב-2 בכל קריאה רקורסיבית?
2. כיצד מטפלים במקרה של מספר בעל ספרה אחת בלבד?

דוגמאות ותנאי קצה

- `IsPalindrome(0) → true`
- `IsPalindrome(1231) → false`, `IsPalindrome(1221) → true`

שאלה 5

HasDigit — האם מספר מכיל ספרה נתונה?

דרישות

- הפעולה HasDigit(int num, int digit) תחזיר true אם הספרה digit מופיעה ב־num, אחרת false.
- ספרה חוקית: $0 \leq d \leq 9$.
- מימוש רקורסיבי, ללא המרה למחרוזת.

הסבר מלא

נשלוף בכל צעד את ספרת היחידות ונשווה ל־digit; אם לא נמצאה, נמשיך עם $n/10$.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

- אם $n = 0 \leftarrow$ בדקו האם $d = 0$.
- בדקו את ספרת היחידות: אם היא שווה ל־ $d \leftarrow$ true.
- אחרת המשיכו רקורסיבית עם $n/10$.

קוד (C#)

```
public static bool HasDigit(int num, int digit)
{
    num = Math.Abs(num);

    if (digit < 0 || digit > 9) return false;

    if (num == 0) return digit == 0;
    if (num % 10 == digit) return true;
    return HasDigit(num / 10, digit);
}
```

שאלות הבנה וחשיבה

1. כיצד משנים את הפעולה כדי להחזיר את מספר ההופעות של הספרה?

- $\text{HasDigit}(0,0) \rightarrow \text{true}$
- $\text{HasDigit}(12345,7) \rightarrow \text{false}$
- $\text{HasDigit}(-70127,7) \rightarrow \text{true}$

שאלה 6

GCD — המחלק המשותף הגדול ביותר (אוקלידס)

דרישות

- הפעולה $\text{GCD}(\text{int } a, \text{int } b)$ תחזיר את $\text{gcd}(a, b)$ באמצעות אלגוריתם אוקלידס.
- מספרים שליליים יטופלו כערך מוחלט.
- מימוש רקורסיבי.

הסבר מלא

אלגוריתם אוקלידס מבוסס על רעיון פשוט: אם שני מספרים מתחלקים באותו מספר, גם השארית בחלוקת הגדול בקטן מתחלקת בו. לכן, במקום לחשב ישירות את $\text{gcd}(a, b)$, אנו מחשבים את $\text{gcd}(b, a \bmod b)$ עד שנגיע למקרה בסיס.

מקרה הבסיס מתרחש כאשר אחד המספרים מתאפס. במקרה כזה, gcd הוא הערך המוחלט של המספר השני: $\text{gcd}(a, 0) = |a|$.

$$\text{gcd}(a, b) = \begin{cases} |a|, & b = 0 \\ \text{gcd}(b, a \bmod b), & b \neq 0 \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $b = 0 \leftarrow \text{החזירו } |a|$.
2. אחרת \leftarrow קראו ל- $\text{gcd}(b, a \bmod b)$.

קוד (C#)

```
public static int GCD(int a, int b)
{
    a = Math.Abs(a); b = Math.Abs(b);
    if (b == 0) return a;
```

```
return GCD(b, a % b);
```

```
}
```

דוגמאות ותנאי קצה

- $\text{GCD}(18, 0) \rightarrow 18, \text{GCD}(48, 18) \rightarrow 6$
- $\text{GCD}(0, 0) \rightarrow 0$ (הגדרה מקובלת)

שאלה 7

Factorial — עצרת $n!$

דרישות

- הפעולה `Factorial(int n)` תחזיר $n!$ עבור $n \geq 0$.
- מימוש רקורסיבי; $0! = 1$.

הסבר מלא

$$n! = \begin{cases} 1, & n = 0 \\ n \cdot (n-1)!, & n \geq 1 \end{cases}$$

מקרה הבסיס: $0! = 1$. צעד רקורסיבי: בכל קריאה מקטינים את n ב-1 ומכפילים בתוצאה החוזרת.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n < 0 \leftarrow$ החזירו -1.
2. אם $n = 0 \leftarrow$ החזירו 1.
3. אחרת \leftarrow החזירו $n \cdot \text{Factorial}(n - 1)$.

קוד (C#)

```
public static long Factorial(int n)
{
    if (n < 0) return -1;
    if (n == 0) return 1;
    return n * Factorial(n - 1);
}
```


שאלות הבנה וחשיבה

1. כיצד ניתן לכתוב גרסה איטרטיבית ל-`Factorial`?

דוגמאות ותנאי קצה

• $\text{Factorial}(5) \rightarrow 120$, $\text{Factorial}(0) \rightarrow 1$

שאלה 8

SumDigits — סכום ספרות של מספר

דרישות

- הפעולה `SumDigits(int num)` תחזיר את סכום כל הספרות של `num`.
- פתרון רקורסיבי; קלט שלילי יטופל בערך מוחלט.

הסבר מלא

$$\text{sum}(n) = \begin{cases} 0, & n = 0 \\ (n \bmod 10) + \text{sum}(n/10), & n \neq 0 \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n = 0 \leftarrow$ החזירו 0.
2. אחרת \leftarrow חשבו את ספרת היחידות $(n \bmod 10)$ והוסיפו לסכום של $n/10$.

קוד (C#)

```
public static int SumDigits(int num) {  
    num = Math.Abs(num);  
    if (num == 0) return 0;  
    return (num % 10) + SumDigits(num / 10);  
}
```

שאלות הבנה וחשיבה

1. כיצד משנים לפעולה הסופרת ספרות (מספר הספרות)?

דוגמאות ותנאי קצה

- $\text{SumDigits}(0) \rightarrow 0$
- $\text{SumDigits}(123) \rightarrow 6$
- $\text{SumDigits}(-9070) \rightarrow 16$

שאלה 9

Power — חישוב base^{exp} ללא `Math.Pow`

דרישות

- הפעולה `Power(int baseNum, int exp)` תחזיר $\text{baseNum}^{\text{exp}}$ עבור $\text{exp} \geq 0$.
- מימוש רקורסיבי בסיסי.

הסבר מלא

$$\text{pow}(a, e) = \begin{cases} 1, & e = 0 \\ a \cdot \text{pow}(a, e - 1), & e > 0 \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם (בסיסי)

- אם $e = 0 \leftarrow$ החזירו 1.
- אחרת \leftarrow החזירו $a \cdot \text{pow}(a, e - 1)$.

קוד (C#)

```
public static long Power(int baseNum, int exp)
{
    if (exp < 0) return -1;
    if (exp == 0) return 1;
    return baseNum * Power(baseNum, exp - 1);
}
```

דוגמאות ותנאי קצה

- $\text{Power}(2, 10) \rightarrow 1024$, $\text{Power}(2, 0) \rightarrow 1$
- $\text{Power}(5, 1) \rightarrow 5$

שאלה 10

Fibonacci — חישוב האיבר ה- n בסדרת פיבונאצ'י

דרישות

- הפעולה `Fibonacci(int n)` תחזיר את $F(n)$ לפי ההגדרה: $F(1) = 1$, $F(0) = 0$.
- $F(n) = F(n - 1) + F(n - 2)$ עבור $n \geq 2$.

- מימוש רקורסיבי בסיסי.

הסבר מלא

$$F(n) = \begin{cases} 0, & n = 0 \\ 1, & n = 1 \\ F(n-1) + F(n-2), & n \geq 2 \end{cases}$$

מקרה הבסיס: הערכים הראשונים מוגדרים ישירות $F(1) = 1, F(0) = 0$. צעד רקורסיבי: לכל $n \geq 2$, הפונקציה מחושבת כסכום שני איברים קודמים.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n < 0 \leftarrow$ החזירו -1 (סדרת פיבונאצ'י מוגדרת רק עבור $n \geq 0$).
2. אם $n = 0$ או $n = 1 \leftarrow$ החזירו n .
3. אחרת \leftarrow החזירו $F(n-1) + F(n-2)$.

קוד (C#)

```
public static long Fibonacci(int n)
{
    if (n < 0) return -1;
    if (n <= 1) return n;
    return Fibonacci(n - 1) + Fibonacci(n - 2);
}
```

דוגמאות ותנאי קצה

- $Fibonacci(10) \rightarrow 55, Fibonacci(1) \rightarrow 1, Fibonacci(0) \rightarrow 0$

שאלה 11

IsPalindromeString — האם מחרוזת פלינדרומית?

דרישות

- הפעולה `IsPalindromeString(string str)` תחזיר `true` אם `str` פלינדרום, אחרת `false`.
- מימוש רקורסיבי; אין להשתמש במבני עזר מיותרים.

הסבר מלא

נשווה בין התווים הקיצוניים ← נתקדם פנימה: אם הראשון שונה מהאחרון — המחזורת אינה פלינדרום. אחרת נקרא רקורסיבית על תת-המחזורת הפנימית.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אתחלו מצביעים: $j = |s| - 1, i = 0$.
2. אם $i \geq j$ (המחזורת פלינדרום). $\text{true} \leftarrow i \geq j$
3. אם $s[i] \neq s[j]$ (המחזורת אינה פלינדרום). $\text{false} \leftarrow s[i] \neq s[j]$
4. אחרת ← קראו רקורסיבית עם $i + 1, j - 1$.

קוד (C#)

```
public static bool IsPalindromeString(string str)
{
    return IsPalindromeString(str, 0, str.Length - 1);
}
public static bool IsPalindromeString(string s, int i, int j)
{
    if (i >= j) return true;
    if (s[i] != s[j]) return false;
    return IsPalindromeString(s, i + 1, j - 1);
}
```

דוגמאות ותנאי קצה

- "" → true
- "abca" → false, "aba" → true

שאלה 12

ReverseString — היפוך מחזורת (רקורסיבי)

דרישות

- הפעולה ReverseString(string str) תחזיר את המחזורת ההפוכה.
- מימוש רקורסיבי; מותר להשתמש בפעולת עזר.

הסבר מלא

בכל צעד רקורסיבי נחזיר את התו הראשון לסוף ההיפוך של שאר המחרוזת. מקרה בסיס: אם המחרוזת ריקה ← נחזיר מחרוזת ריקה.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $|s| \leq 1$ ← החזירו את s .
2. אחרת ← החזירו $s[0] + \text{rev}(s[1..])$.

קוד (C#)

```
public static string ReverseString(string str)
{
    if (str.Length <= 1) return str;
    return ReverseString(str.Substring(1)) + str[0];
}
```

דוגמאות ותנאי קצה

• $\text{ReverseString}("") \rightarrow ""$, $\text{ReverseString}("ab") \rightarrow "ba"$

שאלה 13

RemoveChar — הסרת כל המופעים של תו נתון

דרישות

- הפעולה $\text{RemoveChar}(\text{string str}, \text{char ch})$ תחזיר מחרוזת חדשה ללא התו ch .
- פתרון רקורסיבי; אין להשתמש בלולאות.

הסבר מלא

בכל צעד רקורסיבי נבדוק את התו הראשון במחרוזת: אם הוא שווה ל- ch נדלג עליו; אחרת נצרף אותו לתוצאה של קריאה רקורסיבית על שאר המחרוזת.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם המחרוזת ריקה ← החזירו מחרוזת ריקה.
2. השוו את התו הראשון ל- ch :

3. אם שווה ← המשיכו רקורסיבית על `s[1..]` בלי לצרף.
4. אם שונה ← צרפו את התו הראשון לתוצאה של הקריאה הרקורסיבית על `s[1..]`.
5. צריך לממש את הפעולה `SubString` שמחזירה מחרוזת חדשה ללא התו הראשון.

קוד (C#)

```
public static string RemoveChar(string str, char ch)
{
    if (str.Length == 0) return "";
    char head = str[0];
    string tail = Substring(str);
    if (head == ch) return RemoveChar(tail, ch);
    return head + RemoveChar(tail, ch);
}
```

שאלות הבנה וחשיבה

1. כיצד תשנו את האלגוריתם שיסנן קבוצת תווים (למשל כל התנועות)?

דוגמאות ותנאי קצה

- `RemoveChar("", 'x') → ""`
- `RemoveChar("banana", 'a') → "bnn"`

שאלה 14

CountOccurrences — ספירת הופעות של תו

דרישות

- הפעולה `CountOccurrences(string str, char ch)` תחזיר את מספר ההופעות של `ch` ב-`str`.
- מימוש רקורסיבי; ללא לולאות.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

אנו רוצים לספור כמה פעמים התו `ch` מופיע במחרוזת `str`. לשם כך נבדוק בכל צעד רקורסיבי את האיבר הראשון במחרוזת: אם המחרוזת ריקה – נחזיר 0. אם התו הראשון שווה ל-`ch` – נספור אותו כ-1 ונמשיך לבדוק את שאר המחרוזת. אם התו הראשון שונה – פשוט נמשיך לבדוק את שאר המחרוזת בלי להוסיף לספירה. בסוף, הסכימה של כל הצעדים תחזיר את מספר ההופעות הכולל.

צריך לממש את הפעולה SubString שמחזירה מחרוזת חדשה ללא התו הראשון.

קוד (C#)

```
public static int CountOccurrences(string str, char ch)
{
    if (str.Length == 0) return 0;

    if (str[0] == ch)
        return 1 + CountOccurrences(Substring(str), ch);
    else
        return CountOccurrences(Substring(str), ch);
}
```

דוגמאות ותנאי קצה

- CountOccurrences("", 'a') → 0
- CountOccurrences("aaa", 'a') → 3

שאלה 15

CountVowels — ספירת תנועות באנגלית (a,e,i,o,u)

דרישות

- הפעולה CountVowels(string str) תחזיר את מספר התנועות.
- פתרון רקורסיבי; התייחסות לאותיות קטנות/גדולות.

הסבר מלא

בכל צעד נבדוק אם התו הראשון שייך לקבוצת התנועות (a, e, i, o, u באותיות קטנות או גדולות), ונוסיף 1 אם כן. נמשיך רקורסיבית על שאר המחרוזת.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם המחרוזת ריקה ← החזירו 0.
2. אחרת:
3. אם התו הראשון שייך לקבוצת התנועות {a, e, i, o, u, A, E, I, O, U} ← הוסיפו 1.
4. המשיכו רקורסיבית על s[1..].

```
public static int CountVowels(string str)
{
    if (str.Length == 0) return 0;
    bool isVowel = "aeiouAEIOU".IndexOf(str[0]) >= 0;
    if (isVowel)
        return 1 + CountVowels(str.Substring(1));
    else
        return CountVowels(str.Substring(1));
}
```

דוגמאות ותנאי קצה

- $\text{CountVowels}("") \rightarrow 0$
- $\text{CountVowels}(\text{"Hello"}) \rightarrow 2$

שאלה 16

IsSorted — האם מערך של מספרים ממויין בסדר עולה?

דרישות

- הפעולה $\text{IsSorted}(\text{int}[] \text{ arr})$ תחזיר true אם arr ממויין בסדר עולה.
- פתרון רקורסיבי; ללא לולאות.

הסבר מלא

בכל צעד נשווה איברים סמוכים \leftarrow נתקדם באינדקס: אם נמצא זוג יורד — המערך אינו ממויין. אם עברנו את סוף המערך בלי למצוא בעיה — המערך ממויין.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $|A| \leq 1 \leftarrow \text{true}$ (מערך ריק או בעל איבר אחד תמיד ממויין).
2. בדקו אם $A[i] \leq A[i + 1]$:
3. אם כן \leftarrow המשיכו רקורסיבית עם $i + 1$.
4. אם לא $\leftarrow \text{false}$ (המערך אינו ממויין).


```
public static bool IsSorted(int[] arr)
{
    return IsSorted(arr, 0);
}
public static bool IsSorted(int[] a, int i)
{
    if (i >= a.Length - 1) return true;
    if (a[i] > a[i + 1]) return false;
    return IsSorted(a, i + 1);
}
```

שאלות הבנה וחשיבה

1. כיצד תתמכו בבדיקה למערך בסדר יורד?
2. מה קורה עם null או מערך ריק? פרטו התנהגות רצויה.

דוגמאות ותנאי קצה

- `IsSorted(new[] { }) → true`
- `IsSorted(new[] { 1, 2, 2, 3 }) → true`, `IsSorted(new[] { 3, 2, 1 }) → false`

שאלה 17

IsPalindrome — תוכנית אינטראקטיבית לבדיקת מספר פלינדרום

דרישות

- כתבו פעולה `IsPalindrome(int num)`, וצרו תוכנית קונסול שקוראת מספר מהמשתמש ומדפיסה הודעה מתאימה.
- אין להמיר למחרוזת; מותר להשתמש בפעולת עזר להיפוך מספר.

הסבר מלא

נשתמש בהשוואה ל־Reverse ונעטוף בתוכנית קונסול.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. קריאת מספר שלם מהמשתמש והפיכת סדר התווים בעזרת פעולת עזר `Reverse`.
2. בדיקה באמצעות `IsPalindrome`.

קוד (C#)

```
static bool IsPalindrome(int num)
{
    int n = Math.Abs(num);
    return n == Reverse(n, 0);
}
static int Reverse(int n, int acc)
{
    if (n == 0) return acc;
    return Reverse(n / 10, acc * 10 + (n % 10));
}
static void Main()
{
    Console.Write("הכנס מספר ");
    if (int.TryParse(Console.ReadLine(), out int x))
    {
        Console.WriteLine(IsPalindrome(x) ? "פלינדרום" : "לא פלינדרום");
    }
    else
    {
        Console.WriteLine("קלט לא חוקי");
    }
}
```

שאלות הבנה וחשיבה

1. כיצד תטפלו במספרים גדולים מ-`int`?
2. כיצד לשלב בדיקות יחידה לפעולה ולטפל בקצוות (0, שלילי, מסתיים באפסים)?

שאלה 18

DecimalToBinary — המרת מספר שלם לבינארי (מחרוזת)

דרישות

- הפעולה `DecimalToBinary(int n)` תחזיר מחרוזת המייצגת את n בבסיס 2.
- מימוש רקורסיבי.
- אין צורך לטפל בקלט שלילי אם לא נדרש במפורש.

הסבר מלא

$$\text{bin}(n) = \begin{cases} //0//, & n = 0 \\ \text{bin}(n/2) + (n \bmod 2), & n > 0 \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n = 0 \leftarrow$ החזר "0".
2. קרא לפעולת עזר $\text{ToBin}(n)$.
3. ToBin : אם $n = 0 \leftarrow$ החזר מחרוזת ריקה.
4. אחרת \leftarrow קרא רקורסיבית ל- $\text{ToBin}(n/2)$ והוסף בסוף את הספרה $n \bmod 2$.
5. התוצאה המצטברת מהקריאות תחזיר את הייצוג הבינארי המלא.

קוד (C#)

```
public static string DecimalToBinary(int n)
{
    if (n == 0) return "0";
    return ToBin(n);
}
public static string ToBin(int n)
{
    if (n == 0) return "";
    return ToBin(n / 2) + (n % 2).ToString();
}
```

דוגמאות ותנאי קצה

- $\text{DecimalToBinary}(0) \leftarrow "0"$
- $\text{DecimalToBinary}(13) \leftarrow "1101"$

שאלה 19

RecursiveSearch — חיפוש אינדקס של איבר במערך

דרישות

- הפעולה $\text{RecursiveSearch}(\text{int}[] \text{ arr}, \text{int} \text{ target})$ תחזיר את האינדקס של target אם קיים, אחרת -1.
- מימוש רקורסיבי; ללא לולאות.

בכל צעד נבדוק את האיבר הראשון שעדיין לא נבדק (באינדקס i): אם הוא שווה ל- x ← נחזיר את i . אם לא, נתקדם רקורסיבית ל- $i + 1$. אם הגענו לסוף המערך ללא מציאה ← נחזיר -1.

$$\text{find}(A, i, x) = \begin{cases} -1, & i = |A| \\ i, & A[i] = x \\ \text{find}(A, i + 1, x), & \text{אחרת} \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. התחל ב-`RecursiveSearch(arr, 0, target)`.
2. אם i שווה לאורך המערך ← החזר -1 (המספר לא נמצא).
3. בדוק את האיבר $a[i]$.
4. אם $a[i] == x$ ← החזר את i .
5. אחרת ← קרא רקורסיבית ל-`RecursiveSearch(a, i+1, x)`.

קוד (C#)

```
public static int RecursiveSearch(int[] arr, int target)
{
    return RecursiveSearch(arr, 0, target);
}

public static int RecursiveSearch(int[] a, int i, int x)
{
    if (i == a.Length) return -1;
    if (a[i] == x) return i;
    return RecursiveSearch(a, i + 1, x);
}
```

תנאי קצה

- מערך ריק.
- מספר מופעים — מוחזר האינדקס הראשון בלבד.
- ערכים שליליים או גדולים מאוד — נתמכים כמו כל ערך אחר.

FindPath — האם קיים מסלול במבוך דו־ממדי?

דרישות

- הפרמטר: `grid: int[][]` שבו $0 =$ תא חופשי, $1 =$ קיר.
- הפעולה `FindPath(int[][] grid)` תחזיר `bool` אם קיים מסלול מהפינה השמאלית העליונה $(0, 0)$ לפינה הימנית התחתונה $(r - 1, c - 1)$.
- מותר לזוז למעלה/מטה/שמאלה/ימינה בלבד; אין לצאת מגבולות המטריצה ואין לדרוך שוב על תא.

הסבר מלא

נשתמש ב־DFS רקורסיבי עם מערך ביקור. בכל צעד נבדוק חוקיות תא ונגסה את ארבעת הכיוונים.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. **בדיקת קלט:** אם המטריצה ריקה או שיש שורה ריקה \leftarrow החזירו `false`.
2. **אתחול:** הגדירו `rows = grid.Length, cols = grid[0].Length` ואת `vis[rows, cols]` מאותחל ל־`false`.
3. **קריאה רקורסיבית ראשונה:** קרא ל־`Dfs(0, 0)` והחזירו את תוצאתה.
4. **ב־`Dfs(r, c)`:**
 5. אם r, c מחוץ לתחום \leftarrow החזירו `false`.
 6. אם `grid[r][c] == 1` (קיר) או `vis[r, c] == true` (כבר ביקרנו) \leftarrow החזירו `false`.
 7. אם $r = rows - 1$ ו־ $c = cols - 1$ (יעד) \leftarrow החזירו `true`.
 8. סמן ביקור: `vis[r, c] = true`.
 9. נסו ארבעה כיוונים: `Dfs(r+1, c)` או `Dfs(r-1, c)` או `Dfs(r, c+1)` או `Dfs(r, c-1)`.
 10. אם אחת הקריאות החזירה `true` \leftarrow החזירו `true`; אחרת \leftarrow החזרו `false`.

קוד (C#)

```
public static bool FindPath(int[][] grid)
{
    if (grid == null || grid.Length == 0 ||
        grid[0] == null || grid[0].Length == 0)
        return false;

    int rows = grid.Length, cols = grid[0].Length;
```

```

// Start or end blocked → no path
if (grid[0][0] == 1 || grid[rows - 1][cols - 1] == 1)
    return false;

bool[,] vis = new bool[rows, cols];
return Dfs(grid, vis, rows, cols, 0, 0);
}

private static bool Dfs(int[][] grid, bool[,] vis,
                        int rows, int cols, int r, int c)
{
    // Out of bounds
    if (r < 0 || c < 0 || r >= rows || c >= cols)
        return false;

    // Wall or already visited
    if (grid[r][c] == 1 || vis[r, c])
        return false;

    // Reached target
    if (r == rows - 1 && c == cols - 1)
        return true;

    vis[r, c] = true;

    // Explore 4 directions
    return Dfs(grid, vis, rows, cols, r + 1, c) ||
        Dfs(grid, vis, rows, cols, r - 1, c) ||
        Dfs(grid, vis, rows, cols, r, c + 1) ||
        Dfs(grid, vis, rows, cols, r, c - 1);
}

```

דוגמאות

$\text{true} \leftarrow \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$ •

$\text{false} \leftarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ •

SubsetSumExists — האם קיימת תת-קבוצה שסכומה שווה ליעד?

דרישות

- ממשו פעולה `SubsetSumExists(int[] arr, int target)` המחזירה `bool`.
- לכל איבר במערך יש שתי אפשרויות: לכלול את האיבר בתת-הקבוצה/לא לכלול את האיבר בתת-הקבוצה. השתמשו ברקורסיה לחקירת שתי האפשרויות.

הסבר מלא

נגדיר פעולה רקורסיבית עם אינדקס: $f(i, t) = f(i + 1, t) \vee f(i + 1, t - a_i)$.

כלומר, בכל שלב מחליטים אם להתקדם בלי לבחור את a_i , או לבחור אותו ולהקטין את היעד t בהתאם. מקרי בסיס: $t = 0 \leftarrow$ אמת (נמצא תת-קבוצה מתאימה), או $i = |A| \leftarrow$ שקר (הגענו לסוף בלי הצלחה).

דוגמאות

- קלט: `arr = {3, 2, 5}, target = 5` פלט: `true` (כי קיימת תת-קבוצה {5} או {3,2} שסכומה 5).
- קלט: `arr = {1, 4, 6}, target = 8` פלט: `false` (אין תת-קבוצה שסכומה בדיוק 8).
- קלט: `arr = {2, 4, 8}, target = 6` פלט: `true` (כי קיימת תת-קבוצה {2,4} שסכומה 6).

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. קלט: מערך `arr` ויעד t . הגדרו פעולה עזר `Dfs(i, t)` הבודקת את הטווח $1 \dots |A|$.
2. מקרה בסיס \leftarrow אם $t = 0 \leftarrow$ החזירו `true` (נמצא סכום מדויק).
3. מקרה בסיס \leftarrow אם $i = |A|$ או $t < 0 \leftarrow$ החזירו `false` (אין פתרון במסלול זה).
4. מסלול "ללא בחירה" \leftarrow חשבו $Dfs(i + 1, t) \leftarrow$ ממשיכים בלי לכלול את a_i .
5. מסלול "עם בחירה" \leftarrow חשבו $Dfs(i + 1, t - a_i) \leftarrow$ ממשיכים לאחר הכללת a_i .
6. בדיקה \leftarrow אם אחת משתי הקריאות מהצעדים 4–5 החזירה `true` \leftarrow החזירו `true`; אחרת \leftarrow החזירו `false`.
7. קריאה ראשונית \leftarrow `SubsetSumExists(arr, target)` קוראת ל-`Dfs(0, target)`.
8. סיבוכיות \leftarrow זמן אקספוננציאלי $O(2^n)$, זיכרון עומק מחסנית $O(n)$. (להאצה: זיכרון ביניים לפי זוג $((i, t))$).

```

public static bool SubsetSumExists(int[] arr, int target)
{
    return Dfs(arr, 0, target);
}

private static bool Dfs(int[] arr, int i, int t)
{
    if (t == 0) return true;
    if (i == arr.Length || t < 0) return false;
    return Dfs(arr, i + 1, t) || Dfs(arr, i + 1, t - arr[i]);
}

```

שאלות חשיבה

1. מה קורה אם המערך כולל מספרים שליליים — כיצד זה משפיע על האלגוריתם?

שאלה 22

CountElements — ספירת אלמנטים מאינדקס התחלתי

דרישות

- הפעולה `CountElements(int[] arr, int startIndex)` תחזיר את מספר האלמנטים במערך החל מהאינדקס הנתון.
- המימוש יהיה רקורסיבי וללא לולאות.

הסבר מלא

$$C(A, i) = \begin{cases} 0, & i \geq |A| \\ 1 + C(A, i + 1), & \text{תרחא} \end{cases}$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

- קלט: מערך `arr` ואינדקס התחלתי `i = startIndex`.
- מקרה בסיס ← אם $i \geq |A|$ ← החזירו 0 (אין עוד אלמנטים לספור).
- צעד רקורסיבי ← החזירו $1 + C(A, i + 1)$ (סופרים את התא הנוכחי ומתקדמים לאינדקס הבא).
- קריאה ראשונית ← `CountElements(arr, startIndex)`.


```
public static int CountElements(int[] arr, int startIndex)
{
    if (startIndex >= arr.Length) return 0;
    return 1 + CountElements(arr, startIndex + 1);
}
```

דוגמאות ותנאי קצה

- $\text{CountElements}(\text{new int[]} \{\}, 0) \rightarrow 0$
- $\text{CountElements}(\text{new int[]} \{5, 7, 9\}, 1) \rightarrow 2$

שאלה 23

CountVowels — תוכנית אינטראקטיבית לספירת תנועות באנגלית

דרישות

- ממשו פעולה רקורסיבית בשם `CountVowels(string str)` שמחזירה את מספר התנועות במחרוזת באנגלית.
- לאחר מכן, כתבו תוכנית קונסול אינטראקטיבית שקולטת מחרוזת מהמשתמש/ת, קוראת ל-`CountVowels`, ומדפיסה את מספר התנועות.
- התנועות הן `a, e, i, o, u` בשתי הרישיות (קטנה/גדולה).

הסבר מלא

ברקורסיה נבדוק את התו הראשון, ונוסיף 1 אם הוא תנועה; נתקדם על שאר המחרוזת עד שהמחרוזת ריקה.

$$V(s) = (s = \epsilon ? 0 : [s[0] \in \{a, e, i, o, u, A, E, I, O, U\}] + V(s[1..]))$$

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם המחרוזת ריקה \leftarrow החזירו 0.
2. בדקו אם $s[0]$ שייכת לקבוצת התנועות.
3. החזירו (0 או 1) + קריאה רקורסיבית ל-`CountVowels` על $s[1..]$.

```
public static int CountVowels(string str) {
    if (str is null) throw new ArgumentNullException(nameof(str));
    if (str.Length == 0) return 0;
    bool isVowel = "aeiouAEIOU".IndexOf(str[0]) >= 0;
    return (isVowel ? 1 : 0) + CountVowels(str.Substring(1));
}

static void Main() {
    Console.Write("Enter an English string: ");
    string s = Console.ReadLine() ?? "";
    Console.WriteLine(CountVowels(s));
}
```

שאלות הבנה וחשיבה

1. כיצד תתמכו בקבוצת תנועות שונה (לשפות אחרות)?
2. איך ניתן לצמצם יצירת מחרוזות ביניים (ללא Substring)?

דוגמאות ותנאי קצה

- קלט: "Hello" → פלט: 2
- קלט: מחרוזת ריקה → פלט: 0

שאלה 24

PrintReverse — הדפסת 1..n בסדר עולה (רקורסיבי, void)

דרישות

- הפעולה PrintReverse(int n) מקבלת מספר שלם חיובי אחד.
- עליה להדפיס את המספרים מ-1 עד n בסדר עולה, כאשר כל מספר מופרד ברווח.
- הפעולה חייבת להיות רקורסיבית ואינה מחזירה ערך (void).
- אם $n \leq 0$ — אין להדפיס דבר.

הסבר מלא

נשתמש בהדפסה לאחר הקריאה הרקורסיבית: תחילה נדפיס את התחום 1..n-1, ואז נדפיס את n. למרות שם הפעולה, סדר ההדפסה עולה.

הנחיות ליישום (אלגוריתם + קוד)

אלגוריתם

1. אם $n \leq 0$ ← עצירה.
2. קראו רקורסיבית ל- $\text{PrintReverse}(n-1)$.
3. הדפיסו את n ואחריו רווח.

קוד (C#)

```
public static void PrintReverse(int n) {  
    if (n <= 0) return;  
    PrintReverse(n - 1);  
    Console.Write(n + " ");  
}
```

דוגמא

• $\text{PrintReverse}(5) \rightarrow$ הדפסה: 5 4 3 2 1