מבחן אמריקאי מבני נתונים

Table of Contents

- שאלה 1
- שאלה 2
- שאלה 3
- שאלה 4
- שאלה 5
- שאלה 6
- שאלה 7
- שאלה 8
- שאלה 9
- שאלה 10
- תשובות

מבחן אמריקאי לדוגמה במבני נתונים (מבוא בסיסי)

להלן מבחן אמריקאי בעל עשר שאלות, כאשר לכל שאלה חמש אפשרויות בחירה (א'–ה'). השאלות עוסקות במבני נתונים בסיסיים, בפעולות נפוצות הנעשות עליהם ובהגדרות תיאורטיות מרכזיות. המבחן נועד להיות קל יחסית, אך מפורט, כדי לסייע בתרגול והבנה. בסוף המבחן תמצא/י את רשימת התשובות הנכונות. מומלץ לקרוא בעיון את כל האפשרויות לפני בחירת התשובה הנכונה. בהצלחה!

שאלה 1

מהו מבנה נתונים (Data Structure) באופן כללי?

א. תיאור אלגוריתמי המתאר כיצד לשלב מספר קבצי תוכנה לפרויקט אחד.

ב. שיטה המאפשרת העלאת ביצועי מעבד בעת ריצת תוכניות מחשב מסוימות.

ג. הדרך שבה נתונים מאורגנים, מאוחסנים ומעובדים כך שניתן יהיה לבצע עליהם פעולות יעילות.

ד. אוסף של אובייקטים השוכנים ביחד במחלקה אחת בשפת תכנות מונחית עצמים.

ה. פורמט קובץ ספציפי המשמש לשמירת נתונים טבלאיים (כמו קובצי CSV).

שאלה 2

מהי הפעולה הנפוצה ביותר בעת שימוש במחסנית (Stack)?

א. איחוד (merge) בין שני מבני נתונים שונים.
ב. הוספה והסרה של איברים רק בתחילת הרשימה המקושרת.
ג. גישה אקראית לאיברים באמצע, בתחילת או בסוף, לפי הצורך.
ד. הוספה והסרה של איברים מהקצה העליון (top) בלבד, בשיטת "האחרון שנכנס הוא הראשון שיוצא" (LIFO).
ה. הוספה של איברים לאמצע המחסנית כדי להאיץ את החיפוש.

שאלה 3

מה היתרון המרכזי בשימוש במערך (Array) לעומת רשימה מקושרת (Linked List)?

א. גמישות מוחלטת בגודל, המאפשרת להוסיף איברים חדשים בלי הגבלה.
ב. גישה ישירה (Random Access) לנתונים על סמך אינדקס, המאפשרת גישה מהירה לאיבר מסוים.
ג. יכולת חיפוש מהיר באמצעות חיפוש בינארי ללא צורך במיון המערך.
ד. תמיד תופס פחות זיכרון מרשימה מקושרת, ללא יוצא מן הכלל.
ה. מתאים רק לאחסון מספרים שלמים ולא מתאים לסוגי נתונים אחרים.

שאלה 4

מהו תור (Queue) וכיצד הוא פועל?

א. זהו מבנה נתונים הפועל בשיטת LIFO (האחרון נכנס – ראשון יוצא), כמו מחסנית.
ב. זהו מבנה נתונים הפועל בשיטת FIFO (הראשון נכנס – ראשון יוצא), בו מוסיפים איברים מאחור ומסירים מהחזית.
ג. זהו מבנה נתונים המשלב רשימה מקושרת דו-כיוונית עם מערך דו-ממדי.
ד. זהו מבנה נתונים המשמש רק עבור בעיות מיון (Sorting) מורכבות.
ה. זהו מבנה נתונים שתמיד מאוחסן בזיכרון בצורה רציפה ואינו מאפשר הוספה של איברים חדשים.

שאלה 5

בעת מימוש רשימה מקושרת בודדת (Singly Linked List), מה מקובל להחזיק בכל איבר (Node)?

א. מפתח (Key) וערך (Value) ללא הפניית (Pointer) המשך.
ב. ערך (Data) וכן מצביע (Pointer) לאיבר הבא ברשימה (next).
ג. שני מצביעים: לאיבר הקודם ולאיבר הבא, כדי לאפשר ניווט דו-כיווני.
ד. סט מצביעים לכל האיברים ברשימה, לצורך חיפוש מהיר בכל שלב.
ה. מערך של נתונים פנימיים שהופך את הרשימה למערך סטטי באורך קבוע.

שאלה 6

מדוע עצים (Trees) נחשבים למבנה נתונים שימושי ונפוץ?

א. עצים תמיד דורשים פחות זיכרון ממערך, בלי תלות במספר האיברים.
ב. עצים מאפשרים שמירה היררכית של נתונים (כמו מבנה תיקיות במחשב), ומסייעים בחיפוש וארגון מהיר.
ג. עצים מתאימים רק ליצירת תרשימי זרימה (Flow Charts) ואין להם שימוש אחר.
ד. עצים אינם ניתנים למימוש בשפות תכנות מונחות עצמים בגלל ריבוי הורשות.
ה. עצים יכולים להחליף מחסנית ותור בכל היישומים האפשריים ללא יוצא מן הכלל.

שאלה 7

מה ההבדל העיקרי בין עץ חיפוש בינארי (Binary Search Tree – BST) לעץ בינארי סתמי (Binary Tree)?

א. בעץ חיפוש בינארי כל צומת (Node) יכול להכיל רק מספרים שלמים.
ב. בעץ חיפוש בינארי, תכונת הסדר קובעת שכל איבר בצד שמאל קטן מהצומת, וכל איבר בצד ימין גדול ממנו.
ג. עץ בינארי סתמי צריך להכיל לפחות שלושה צמתים.
ד. בעץ בינארי סתמי אף פעם אין יותר משני בנים (Children), אבל בעץ חיפוש בינארי עשויים להיות יותר משני בנים לכל צומת.
ה. בעץ חיפוש בינארי אין הגבלה על מספר הצמתים העליונים (Roots).

שאלה 8

בעת מימוש טבלת פיזור (Hash Table), מהו 'ערך הגיבוב' (Hash Value)?

א. זהו מספר הנוצרים באופן אקראי כשלוחצים על מקש Enter במקלדת.
ב. זהו מספר המחושב על בסיס מפתח מסוים (Key) ומשמש לקבוע מיקום אפשרי של האיבר במערך הפנימי של הטבלה.
ג. זהו מזהה ייחודי לכל אובייקט בשפת התכנות, שאינו קשור לשימוש בפונקציית גיבוב.
ד. זהו מספר המייצג תמיד את כמות ההתנגשויות (Collisions) שיש בטבלה.
ה. ערך הגיבוב הוא תמיד 0 או 1, כמו בביט בודד, ולכן קל מאוד לאחסן את המידע.

שאלה 9

מהי בדרך כלל המורכבות בזמן (Time Complexity) של חיפוש איבר בטבלת פיזור (Hash Table) בנסיבות אידיאליות (כלומר, עם פונקציית גיבוב טובה וחלוקה שווה יחסית של נתונים)?

א. $O(1)$ – חיפוש בזמן קבוע בממוצע.
ב. $O(n)$ – חיפוש דורש מעבר על כל האיברים במערך.
ג. $O(\log n)$ – חיפוש בינארי הוא הדרך העיקרית למציאת האיבר.
ד. $O(n^2)$ – נדרשים שני לולאות מקוננות לבדיקת כל איבר כנגד כל איבר אחר.
ה. $O(1)$ במקרה הגרוע ביותר, אך $O(\infty)$ במקרה הממוצע.

שאלה 10

איזו אפשרות מתארת בצורה נכונה את מבנה הנתונים 'ערימה' (Heap)?

א. מבנה נתונים המבוסס על עיקרון FIFO, אך עם עדיפויות שונות לכל איבר.
ב. מבנה נתונים בו האיבר הגדול ביותר או הקטן ביותר (תלוי בסוג הערימה) נשמר תמיד בשורש (Root), ומאפשר גישה מהירה לאיבר זה.
ג. מחסנית (Stack) מיוחדת שבה האיברים ממוינים תמיד לפי סדר עולה.
ד. מבנה נתונים בו כל איבר מצביע על שני איברים נוספים, אחד גדול ממנו ואחד קטן ממנו.
ה. מבנה נתונים שנועד לאחסן ישויות גאומטריות דוגמת משולשים ומרובעים בלבד.

תשובות

1. **ג**

מבנה נתונים הוא אוסף של נתונים המאורגנים ומאוחסנים באופן המאפשר גישה ופעולות יעילות (הוספה, חיפוש, מחיקה וכדומה). התשובות האחרות עוסקות בנושאים שונים שאינם רלוונטיים להגדרת מבנה נתונים.

2. **ד**

מחסנית (Stack) פועלת בשיטת LIFO. אפשר להכניס איבר חדש רק בקצה העליון ולהסיר תמיד מהקצה העליון.

3. **ב**

במערכים ניתן לגשת לאיבר ישירות לפי האינדקס שלו, דבר שמאפשר גישה מהירה. יתר התשובות אינן נכונות או מדויקות בקשר ליתרון מרכזי זה.

4. **ב**

תור (Queue) פועל בשיטת FIFO. מוסיפים איברים בחלקו האחורי (Rear) ומסירים מחזית התור (Front).

5. **ב**

רשימה מקושרת בודדת מורכבת מצומת המכיל ערך ומצביע לצומת הבא (next). רשימה דו-כיוונית הייתה מכילה שני מצביעים (לצומת הבא והקודם).

6. **ב**

עצים מאפשרים ארגון היררכי של מידע, ומקלים על חיפוש וסידור של נתונים. לדוגמה, מבנה תיקיות או אחסון מיון בספריות מסודרות.

7. **ב**

בעץ חיפוש בינארי מתקיימת תכונת הסדר – בצד שמאל יש ערכים קטנים מהצומת, ובצד ימין ערכים גדולים ממנו.

8. **ב**

ערך הגיבוב (Hash Value) מחושב לפי פונקציית גיבוב ומשמש לזיהוי מיקום אפשרי של המפתח (Key) במבנה טבלת הפיזור.

9. **א**

במקרה האידיאלי, עם חלוקה שווה ופונקציית גיבוב טובה, מציאת איבר בטבלת פיזור נעשית בממוצע בזמן קבוע ((O(1).

10. **ב**

בערימה (Heap) נשמר האיבר המינימלי או המקסימלי בשורש (תלוי אם זו Min-Heap או Max-Heap), ומכאן ניתן לגשת אליו בזמן מהיר.

בהצלחה בלימודים ובתרגול!

ניתן לעבור על השאלות שוב ולחשוב כיצד כל מבנה נתונים פועל, מהם החסרונות והיתרונות שלו, ומהן הפעולות העיקריות שניתן לבצע עליו. הבנה של עקרונות אלה חיונית ללימוד מעמיק יותר של אלגוריתמים ומבני נתונים מתקדמים.