הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

מבוא לתכנות מערכות
תרגיל בית מספר 4

מועד פרסום: 21.6.2026 בשעה 23:00
מועד הגשה: 28.6.2026 בשעה 23:59

מטלת בית: מ C-ל – ++C-מחלקת `DynamicArray`

רקע

בשפת C, ניהול מערכים דינמיים דרש לרוב הגדרת `struct` המכיל מצביע למידע ומשתנה השומר את הגודל, לצד אוסף פונקציות המקבלות מצביעים לאותו `struct` בתרגיל זה, ניקח את הקונספט הזה ונהפוך אותו למחלקה (Class) תקנית ב C++, תוך ניצול היכולות של השפה לניהול זיכרון בטוח, קוד קריא יותר ושימוש נכון בהרשאות גישה.

מטרות התרגיל

- מעבר משימוש במבנים ופונקציות גלובליות לגישה מונחית עצמים.
- הגדרת טיפוסים פנימיים למחלקה
- שימוש בבנאים והורסים לניהול מחזור חיי האובייקט והקצאות דינמיות.
- העמסת פונקציות.(Function Overloading)
- אתחול משתני מחלקה באמצעות רשימות אתחול.(Initialization Lists)
- עבודה עם רפרנסים במקום מצביעים.

תיאור המטלה

עליכם לממש מחלקה בשם `DynamicArray` המייצגת מערך דינמי של מספרים שלמים (`int`). המחלקה תנהל את הקצאת הזיכרון באופן אוטומטי ותספק ממשק נוח לגישה, הוספה ומחיקה של איברים.

1. *מבנה המחלקה ושדות פרטיים*

המחלקה תכיל את השדות הפרטיים הבאים:

- `int* data` – מצביע למערך הדינמי.
- `int size` – מספר האיברים הקיימים במערך כרגע.
- `int capacit` – הקיבולת המקסימלית הנוכחית של המערך.

2. *טיפוס שגיאות פנימי(Nested Enum)*

תחת הרשאת הגישה הציבורית (`public`) של המחלקה, עליכם להגדיר `enum` בשם `Result` אשר ישמש לדיווח על הצלחה או כישלון של פעולות. ה `enum`-יכיל את הערכים הבאים:

- `SUCCESS` – הפעולה הסתיימה בהצלחה.
- `OUT_OF_BOUNDS` – נסיון גישה או פעולה מחוץ לגבולות המערך.
- `NOT_FOUND` – האיבר המבוקש לא נמצא.

הערה: מכיוון שה enum-מוגדר בתוך המחלקה, קוד חיצוני המשתמש בו יצטרך לפנות אליו באמצעות אופרטור סקופ, למשל `DynamicArray::SUCCESS`.

3. *בנאים והורס(Constructors & Destructor)*

עליכם לממש את המתודות הבאות. **חובה** להשתמש *ברשימות אתחול* (Initialization Lists)היכן שניתן עבור אתחול השדות:

- **בנאי ברירת מחדל :**מאתחל מערך ריק לחלוטין (למשל `capacity`, של 2, ו `size`-של 0)
- **בנאי פרמטרי :**מקבל מספר שלם `initialCapacity` הבנאי מקצה זיכרון למערך בקיבולת זו.
- **בנאי העתקה :(Copy Constructor)** מקבל *רפרנס קבוע* לאובייקט אחר מאותו סוג, ומייצר עותק (Deep Copy) של המערך.
- **הורס :(Destructor)** משחרר את הזיכרון שהוקצה דינמית (באמצעות `delete[]` כדי למנוע דליפות זיכרון.

4. *הוספת איברים והעמסת פונקציות*

עליכם לממש פונקציות בשם `append`, תוך שימוש בהעמסה:

- `void append(int value)` מוסיפה מספר שלם יחיד לסוף המערך. אם המערך מלא (`size == capacity`), יש להקצות זיכרון חדש **כפול** בגודלו, להעתיק את הנתונים הישנים, ולשחרר את הזיכרון הישן.
- `void append(const DynamicArray& other)` מקבלת *רפרנס* למערך דינמי אחר, ומוסיפה את כל האיברים שלו לסוף המערך הנוכחי.

5. *מתודות הפעולה מחזירות* `Result`

- `Result insert(int index, int value)` מכניסה איבר במיקום ספציפי ודוחפת את שאר האיברים ימינה. במידת הצורך מכפילה את הקיבולת. מחזירה `OUT_OF_BOUNDS`עבור אינדקס לא חוקי, אחרת `SUCCESS`.
- `Result remove(int index)` מסירה איבר במיקום ספציפי ודוחפת את שאר האיברים שמאלה כדי לסגור את הפער. ה `size`-קטן ב-1. מחזירה `OUT_OF_BOUNDS` עבור אינדקס לא חוקי, אחרת `SUCCESS`.
- `Result find(int value, int& outIndex) const` מחפשת את הערך `value` במערך. אם נמצא, מעדכנת את `outIndex` לאינדקס שלו ומחזירה `SUCCESS`. אם אינו קיים, מחזירה `NOT_FOUND` ללא שינוי ה-`outIndex`.

6. *מתודות עזר נוספות*

- `void print() const` מדפיסה את המערך בפורמט צמוד.`[1, 2, 3]`
- `int& getAt(int index)` מחזירה *רפרנס* לאיבר במיקום מסוים. אם האינדקס מחוץ לגבולות ה-`size`הפונקציה תדפיס הודעת שגיאה ל `cerr`-ותסיים את התוכנית מיד בעזרת `exit(1)`.

הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

תוכנית הבדיקה (main.cpp) והנחיות פלט

יש להשתמש בקובץ ה main.cpp- המסופק. בפרט אין לערוך שינויים כלשהם בקובץ

הנחיות הגשה ובדיקה

1. **ארגון קבצים :**חובה להגיש שני קבצים: קובץ כותרת DynamicArray.h וקובץ מימוש DynamicArray.cpp.
2. **ניהול זיכרון :**חלק מהותי מהציון על התרגיל יסתמך על ניהול זיכרון תקני. קוד ההגשה שלכם יורץ בסביבת בדיקות אוטומטית שכוללת כלי אבחון כגון valgrind / leaks שגיאות גישה לזיכרון או דליפות זיכרון יגררו הפחתה משמעותית בציון.
3. **התאמת פלט :(STDOUT)** סקריפט הבדיקה ישווה את פלט התוכנית שלכם במדויק (byte-by-byte) לפלט המצופה באמצעות פקודות כמו diff ודאו שאין לכם שגיאות כתיב בהדפסות, רווחים עודפים או הדפסות דיבאג מיותרות שהשארתם בטעות. כל חריגה תגרום לכישלון הבדיקה האוטומטית.

בהצלחה!