בונוס 5 – Smart Pointers

מה זה RAII – Resource Acquisition Is Initialization ?

עקרון תכנותי אשר מטרתו לנהל משאבי מערכת, כגון: file descriptors, מצביעים, Mutex, sockets ועוד. הוא מסתמך על כך שיוצרים אובייקט אשר מוקדש לאותו משאב מערכת שברצוננו לנהל. בצורה זו מקבילים את אורך החיים (lifetime) של המשאב לאורך החיים של האובייקט.

RAII מבטיח את קיומו של המשאב לכל פונקציה אשר יכולה לגשת לאובייקט, בנוסף לכך כאשר פג אורך החיים של האובייקט גם המשאב משוחרר (מחיקת מבציע, שחרור lock, סגירת קובץ).'

RAII מאפשר מניעת זליגות זיכרון, ומבטיח exception safe code.

הרעיון הכללי: יוצרים מחלקה חדשה, אשר בconstructor תאתחל את המשאב על פי הפרמטרים שהועברו לבנאי, ותשמור את המשאב במשתנה private. ב- Destructor הפונקציה תשחרר את המשאב שהוקצה.

לפירוט נוסף, מוזמנים להיעזר ב: https://en.cppreference.com/w/cpp/language/raii, או בכל מקור מידע אחר, **ולשתף ביניכם (sharing is caring).**

איך RAII נכנס בתרגיל ?

הדרך הנכונה לעבוד עם קבצים ב cpp היא ע"י streams שממומשים כחלק מהספרייה הסטנדרטית ב C++. האובייקט std::iostream("file_path") יודע לקבל נתיב לקובץ, לפתוח אותו ולשמור מצביע ל file descriptor של הקובץ. ניתן לקרוא ולכתוב לקובץ זה, ע"י מתודות של האובייקט, אולם אין צורך לסגור את הקובץ, כיוון שהאובייקט iostream, מימש ב destructor שלו סגירה של הקובץ בכל מקרה.

בתרגיל זה, נרצה שתממשו עבודה עם קבצים ללא שימוש במנגנון stream המובנה של CPP. אתם נדרשים לממש מחלקה File שעונה על הממשק הבא:

```cpp
#pragma once

#include <cstdio>
#include <string>

// The class responsible for holding FILE pointer - RAII implementation of FILE resource
class File {
public:

    /* File C'tor, open file path (name) with specified mode*/
    File(const std::string& name, const char* mode);

    // Implicit conversion function to FILE*
    operator FILE*();

    // D'tor for File object, closes opened file
    ~File();

private:
    FILE* file_fd;
};
```

מוזמנים לכתוב תוכנית main קצרה אשר תבדוק שימוש באובייקט File.

מה המשימה שלכם (grep)?

פקודה grep בלינוקס: מאפשרת לחפש מחרוזות בתוך מספר קבצים ויודעת להדפיס את השורה בה הופיעה המחרוזת ואת הקובץ בו היא מצאה אותו.

מטרתנו תרגיל זה היא לממש גרסה מעודנת של הפקודה grep ב-cpp תוך שימוש בכל העקרונות תכנות שנלמדו עד כה בקורס, ובפרט ב RAII לצורך עבודה עם קבצים.

מצורפים מספר ספרים פופולריים בקרב בני נוער אי שם באמריקה הדרום מזרחית. אותם בני נוער מעוניינים למצוא את כל השורות בהם מופיעות מחרוזות שונות לבחירתם, **עליכם מוטלת המשימה לבנות תכנית בעלת המפרט הבא:**

התכנית תיקרא lightweight_grep, ותקבל מספר משתנים.

- המשתנה הראשון יהיה המחרוזת אותה נרצה לחפש. במידה והמחרוזת מכילה רווחים היא תוקף בגרשיים לדוגמא "hello world", במידה ופרמטר לתוכנית מוקף בגרשיים, הוא יופיע באותו שדה במערך argv, ללא הגרשיים.
  דוגמא לתוכנית אשר מדפיסה את הפרמטר הראשון שניתן לה, ודוגמת הרצה שלה:

```
#include <iostream>

int main(int argc, char* argv[])
{
    std::cout << "Hello World!\n";
    std::cout << argv[1] << std::endl;
}
```

```
C:\Users\□□□\source\repos\lightweight_grep\Debug>lightweight_grep.exe "magsimim program"
Hello World!
magsimim program
```

- שאר המשתנים הם path לקבצים בהם יש לחפש את המחרוזת המופיעה בפרמטר הראשון. כאמור במידה והpath מכיל רווחים הוא יוקף בגרשיים.

Usage:

lightweight_grep <string to search> <first file path> <second file path> ...

אנא וודאו את הקלט מהמשתמשים והדפיסו הודעות שגיאה אינדיקטיביות. זכרו שהמשתמשים של התוכנית אינם מכירים את הקוד של התוכנית, ולכן הפלט והודעות השגיאה צריכות להיות כמה שיותר מובנות ותמציתיות.

<u>איך קוראים שורה מקובץ?</u> השתמשו בפונקציה fgets, הניחו כי אורך שורה מקסימלי הוא 1024 בתים (כי מי יכול לקרוא שורות יותר ארוכות מ- 1 KB).

https://en.cppreference.com/w/cpp/string/basic_string/getline

<u>עבודה עם קבצים:</u> כאמור בחלק הקודם, מימשתם אובייקט File שמאפשר פתיחת וסגירת קבצים בצורה נוחה ונכונה תוך שימוש בכללי RAII, אנא השתמשו בו בחלק זה על מנת לפתוח קבצים ולקרוא מהם.

מצורפת תכנית לדוגמא, דוגמא להרצה (כמובן שאין צורך לממש את ה- ascii art):

```
C:\Users\□□□\Documents\Mordehai>"C:\Users\□□□\source\repos\lightweight_grep\Debug\lightweight_grep.exe" "apple" alice_in_wonderland.txt the_little_prince.txt

=====================================================================
  ____                  _    _     _   _
 / ___|             | |   (_)   | |  | |
| |   __ _ __ __ _ __  | |    _  __ _| |__ | |_
| | |_ | '__/ _ \ '_ \ | |   | |/ _` | '_ \| __|
| |__| | | | __/ |_) | | |___| | (_| | | | | |_
 \____|_|  \___| .__/ |_____|_|\__, |_| |_|\__|
                | |             __/ |
                |_|            |___/


 ___        __  __             _    _        _
|  _ \      |  \/  |           | |  (_)      (_)
| |_) |_   _ | \  / | __ _ __ _ __| |_  __ _ __ _ __ __ __ __ __ __ __ __
|  _ <| | | | |\/| |/ _` |/ _` / __| '_ \| | '_ ` _ \| | '_ ` _ \ \ \ \ \ \ \ \ \ \
| |_) | |_| | | |  | | (_| | (_| \__ \ | | | | | | | | | | | | | | | | | | |
|____/ \__, | |_|  |_|\__,_|\__, |___/_| |_|_|_| |_| |_|_|_| |_| |_|
       __/ |                __/ |
      |___/               |___/

=====================================================================


alice_in_wonderland.txt
Found 3 instances of pattern apple in file alice_in_wonderland.txt
        -> of mixed flavour of cherry-tart, custard, pine-apple, roast
        -> I'm here!  Digging for apples, yer honour!'
        ->   `Digging for apples, indeed!' said the Rabbit angrily.  `Here!
the_little_prince.txt
Found 1 instances of pattern apple in file the_little_prince.txt
        -> Γ□£I am right here,Γ□¥ the voice said, Γ□£under the apple tree.Γ□¥ Γ□£

C:\Users\□□□\Documents\Mordehai>_
```

בנוסף מצורפים 2 הטקסטים מהדוגמא, נסו לשחזר את הדוגמא והגיעו לפלט דומה, מוזמנים לחשוב על חיפושים מעניינים נוספים ולשתף אותנו.

בהצלחה 😊