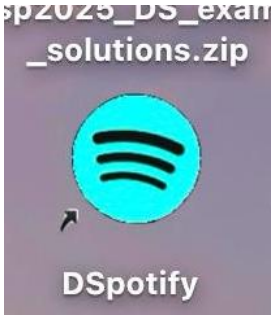
מתרגל ממונה על התרגיל: ניב סולימני, sulimany.niv@campus.technion.ac.il

תאריך ושעת הגשה: 27/06/2025 בשעה 23:59

אופן ההגשה: בזוגות. אין להגיש ביחידים. (אלא באישור מתרגל אחראי של הקורס)

הנחיות כלליות:

- שאלות על התרגיל יש לפרסם באתר הפיאצה של הקורס תחת לשונית ״hw-wet-2״:
  - האתר: https://piazza.com/class/m8krba145sw2zt, נא לקרוא את השאלות של סטודנטים אחרים לפני שמפרסמים שאלה חדשה, למקרה שנשאלה כבר.
- נא לקרוא את המסמך ״נהלי הקורס״ באתר הקורס. בנוסף, נא לקרוא בעיון את כל ההנחיות בסוף מסמך זה.
- בפורום הפיאצה ינוהל FAQ ובמידת הצורך יועלו תיקונים כ**הודעות נעוצות** (Pinned Notes). תיקונים אלו **מחייבים**.
- התרגיל מורכב משני חלקים: יבש ורטוב.
  - לאחר קריאת כלל הדרישות, מומלץ לתכנן תחילה את מבני הנתונים על נייר. דבר זה יכול לחסוך לכם זמן רב.
  - לפני שאתם ניגשים לקודד את פתרונכם, ודאו כי יש לכם פתרון העומד בכל דרישות הסיבוכיות בתרגיל. תרגיל שאינו עומד בדרישות הסיבוכיות יחשב כפסול.
  - את הפתרון שלכם מומלץ לחלק למחלקות שונות שאפשר לממש (ולבדוק!) בהדרגתיות.
  - המלצות לפתרון התרגיל נמצאות באתר הקורס תחת: ״Programming Tips Session״.
    - המלצות לתכנות במסמך זה אינן מחייבות, אך מומלץ להיעזר בהן.
  - חומר התרגיל הינו כל החומר שנלמד בהרצאות ובתרגולים עד מיונים (לא כולל).
- העתקת תרגילי בית רטובים תיבדק באמצעות תוכנת בדיקות אוטומטית, המזהה דמיון בין כל העבודות הקיימות במערכת, גם כאלו משנים קודמות. לא ניתן לערער על החלטת התוכנה. התוכנה אינה מבדילה בין מקור להעתק! אנא הימנעו מהסתכלות בקוד שאינו שלכם.
- בקשות להגשה מאוחרת יש להפנות למתרגל האחראי בלבד בכתובת: jonathan.gal@campus.technion.ac.il



הקדמה:

בעקבות ההצלחה שלכם בתרגיל הבית הקודם, מנהלי **DSpotify** רוצים להמשיך לעבוד על המערכת ולהרחיב את היכולות שלה. כעת נרצה לזהות כל שיר עם ז'אנר ספציפי, ולהיות מסוגלים להוסיף, להוריד, ולקבל מידע לגבי הז'אנרים והשירים השונים.

סימונים לצורכי סיבוכיות:

נסמן ב-n את מספר השירים במערכת.

ב-m את מספר הז'אנרות במערכת.

דרוש מבנה נתונים למימוש הפעולות הבאות:

DSpotify_t()

מאתחלת מבנה נתונים ריק. תחילה אין במערכת שירים או ז'אנרים.

פרמטרים: אין

ערך החזרה: אין

סיבוכיות זמן: $O(1)$ במקרה הגרוע.

---

virtual ~DSpotify_t()

הפעולה משחררת את המבנה (כל הזיכרון אותו הקצאתם חייב להיות משוחרר).

פרמטרים: אין

ערך החזרה: אין

סיבוכיות זמן: $O(n+m)$ במקרה הגרוע.

StatusType addGenre(int genreId)

הפעולה מוסיפה למבנה ז'אנר חדש שאין לו שירים עדיין.

פרמטרים:

genreId הז'אנר החדש שיש להוסיף.

ערך החזרה:

ALLOCATION_ERROR במקרה של בעיה בהקצאה/שחרור זיכרון.

INVALID_INPUT אם genreId <=0.

FAILURE אם קיים כבר ז'אנר עם מזהה genreId.

SUCCESS במקרה של הצלחה.

סיבוכיות זמן: $O(1)$ באופן משוערך.

StatusType addSong(int $songId$, int $genreId$)

שיר בעל מזהה ייחודי $songId$ מתוסף למערכת, השיר מ-ז'אנר שמתאים למזהה genreId .

פרמטרים:

$songId$	מזהה השיר שצריך להוסיף.
$genreId$	ז'אנר השיר.

ערך החזרה:

ALLOCATION_ERROR	במקרה של בעיה בהקצאה/שחרור זיכרון.
INVALID_INPUT	אם $songId$ <=0 או אם genreId<=0.
FAILURE	אם קיים כבר שיר במזהה $songId$ במערכת או שלא קיים ז'אנר עם מזהה genreId.
SUCCESS	במקרה של הצלחה.

סיבוכיות זמן: $O(\log^* n)$ באופן משוערך ביחד עם mergeGenres, getSongGenre ו- getNumberOfGenreChanges במומצע על הקלט.

StatusType mergeGenres(int $genreId1$, int $genreId2$, int $genreId3$)

מאחדים את כל השירים שיש במבנה תחת הז'אנרות עם מזהים genreId1, genreId2. כל השירים שאוחדו יהפכו להיות כעת בז'אנר עם מזהה genreName3. בסוף הפעולה אין שירים בז'אנרות הישנות.

פרמטרים:

$genreId1$	מזהה הז'אנר הראשון.
$genreId2$	מזהה הז'אנר השני.
$genreId3$	מזהה הז'אנר החדש.

ערך החזרה:

ALLOCATION_ERROR	במקרה של בעיה בהקצאה/שחרור זיכרון.
INVALID_INPUT	אם genreId1<=0 או genreId2<=0 או genreId3<=0 או genreId1=genreId2 או genreId2=genreId3 או genreId1=genreId3.
FAILURE	אם הז'אנר בעל מזהה genreId1 או genreId2 לא קיימים במערכת או אם הז'אנר עם מזהה genreId3 קיים במערכת.
SUCCESS	במקרה של הצלחה.

סיבוכיות זמן: $O(\log^* n)$ באופן משוערך ביחד עם addSong, getSongGenre ו-getNumberOfGenreChanges במומצע על הקלט.

---

output_t $< int >$ $getSongGenre$(int songId)

מחזיר את מזהה הז'אנר של השיר עם מזהה songId.

פרמטרים:

songId	מזהה השיר.

ערך החזרה:

ALLOCATION_ERROR	במקרה של בעיה בהקצאה/שחרור זיכרון.
INVALID_INPUT	אם songId <=0.
FAILURE	אם אין שיר עם מזהה songId.
SUCCESS	במקרה של הצלחה, במקרה זה גם יוחזר מזהה הז'אנר של השיר.

סיבוכיות זמן: $O(\log^* n)$ באופן משוערך ביחד עם mergeGenres, addSong ו-getNumberOfGenreChanges במומצע על הקלט.

output_t $< int >$ getNumberOfSongsByGenre(int $genreId$)

מחזיר את מספר השירים מז'אנר עם מזהה genreId.

<u>פרמטרים</u>:

$genreId$	מזהה הז'אנר.

<u>ערך החזרה</u>:

ALLOCATION_ERROR	במקרה של בעיה בהקצאה/שחרור זיכרון.
INVALID_INPUT	אם $genreId \leq 0$.
FAILURE	אם אין ז'אנר עם מזהה genreId
SUCCESS	במקרה של הצלחה. במקרה זה גם יוחזר מספר השירים בז'אנר.

<u>סיבוכיות זמן</u>: $O(1)$ בממוצע על הקלט.

---

output_t $< int > getNumberOfGenreChanges$(int $songId$)

מחזיר כמה ז'אנרות שונות היו לשיר עם מזהה songId. השיר מתחיל עם ז'אנר יחיד, ובכל פעם שהז'אנר שלו מתאחד עם ז'אנר נוסף תחת קריאה ל-mergeGenres(genreId1, genreId2, genreId3) השיר נחשב כעבר שינוי של ז'אנר וכעת הוא עם ז'אנר genreId3.

<u>פרמטרים</u>:

$songId$	מזהה השיר.

<u>ערך החזרה</u>:

ALLOCATION_ERROR	במקרה של בעיה בהקצאה/שחרור זיכרון.
INVALID_INPUT	אם $songId <= 0$.
FAILURE	אם אין שיר במזהה $songId$ במערכת.
SUCCESS	במקרה של הצלחה. במקרה זה גם יוחזר מספר הז'אנרות השונות.

<u>סיבוכיות זמן</u>: $O(\log^* n)$ באופן משוערך ביחד עם mergeGenres, getSongGenre ו-addSong בממוצע על הקלט.

דוגמה הרצה:

addSong(1, 30): SUCCESS
addSong(2, 20): SUCCESS
addSong(3, 10): SUCCESS
addSong(4, 20): SUCCESS
getSongGenre(3): SUCCESS, 10
getSongGenre(2): SUCCESS, 20
getNumberOfSongsByGenre(20): SUCCESS, 2
getNumberOfGenreChanges(2): SUCCESS, 1
mergeGenres(30, 20, 40): SUCCESS
getNumberOfSongsByGenre(40): SUCCESS, 3
getNumberOfSongsByGenre(20): SUCCESS, 0
getSongGenre(2): SUCCESS, 40
getNumberOfGenreChanges(1): SUCCESS, 2

סיבוכיות מקום:

סיבוכיות המקום הדרושה עבור מבנה הנתונים היא $O(n+m)$ במקרה הגרוע. כלומר בכל רגע בזמן הריצה, צריכת המקום של מבנה הנתונים תהיה לינארית במספר השירים ובמספר הז'אנרים השונים.

סיבוכיות המקום הנדרשת עבור כל פעולה (כלומר, זיכרון "העזר" שכל פעולה משתמשת בו) אינה מצוינת לכל פעולה לחוד, אך אסור לעבור את סיבוכיות המקום הדרושה שמוגדרת לכל המבנה.

ערכי החזרה של הפונקציות:

כל אחת מהפונקציות מחזירה ערך מטיפוס StatusType שייקבע לפי הכלל הבא:

- תחילה, יוחזר INVALID_INPUT אם הקלט אינו תקין.
- אם לא הוחזר INVALID_INPUT:
  - בכל שלב בפונקציה, אם קרתה שגיאת הקצאה/שחרור יש להחזיר ALLOCATION_ERROR. מצב זה אינו צפוי אלא באחד משני מקרים (לרוב): באמת השתמשתם בקלט גדול מאוד ולכן המבנה ניצל את כל הזיכרון במערכת, או שיש זליגת זיכרון בקוד.
  - אם קרתה שגיאה אחרת, כפי שמצוין בכל פונקציה, יש להחזיר מיד FAILURE מבלי לשנות את מבנה הנתונים.
- אחרת, יוחזר SUCCESS.

חלק מהפונקציות צריכות להחזיר בנוסף עוד פרמטר (int או bool), לכן הן מחזירות אובייקט מטיפוס output_t<T>.
אובייקט זה מכיל שני שדות: הסטטוס (__status) ושדה נוסף (__ans) מסוג T.
במקרה של הצלחה (SUCCESS), השדה הנוסף יכיל את ערך החזרה, והסטטוס יכיל את SUCCESS. בכל מקרה אחר, הסטטוס יכיל את סוג השגיאה והשדה הנוסף לא מעניין.
שני הטיפוסים (output_t<T>,StatusType) ממומשים כבר בקובץ "wet1util.h" שניתן לכם כחלק מהתרגיל.
קייסם קונסטרקטור של output_t<T> מ-T ומ-StatusType כך שניתן פשוט לכתוב בפונקציות הרלוונטיות:

```
return 7;
return StatusType::FAILURE;
```

הנחיות ודגשים כלליים:

חלק יבש:

- החלק היבש הווה חלק מהציון על התרגיל כפי שמצוין בנהלי הקורס.

- לפני מימוש הפעולות בקוד יש לתכנן היטב את מבני הנתונים והאלגוריתמים ולוודא כי באפשרותכם לממש את הפעולות בדרישות הזמן והזיכרון שלעיל.
- החלק היבש חייב להיות מוקלד.
- הגשת החלק הרטוב מהווה תנאי הכרחי לקבלת ציון על החלק היבש, כלומר, הגשה בה יתקבל אך ורק חלק יבש תגרור ציון 0 על התרגיל כולו.
- יש להכין מסמך הכולל תיאור של מבני הנתונים והאלגוריתמים בהם השתמשתם בצירוף הוכחת סיבוכיות הזמן והמקום שלהם. חלק זה עומד בפני עצמו וצריך להיות מובן לקורא גם לפני העיון בקוד. אין צורך לתאר את הקוד ברמת המשתנים, הפונקציות והמחלקות, אלא ברמה העקרונית. חלק יבש זה <u>לא</u> תיעוד קוד.
- ראשית הציגו את מבני הנתונים בהם השתמשתם. רצוי ומומלץ להיעזר בציור.
- לאחר מכן הסבירו כיצד מימשתם כל אחת מהפעולות הנדרשות. הוכיחו את דרישות סיבוכיות הזמן של כל פעולה תוך כדי התייחסות לשינויים שהפעולות גורמות במבני הנתונים.
- הוכיחו שמבנה הנתונים וכל הפעולות עומדים בדרישת סיבוכיות המקום.
- החסמים הנתונים בתרגיל הם לא בהכרח הדוקים ולכן יכול להיות שקיים פתרון בסיבוכיות טובה יותר. מספיק להוכיח את החסמים הדרושים בתרגיל.
- רמת פירוט: יש להסביר את כל הפרטים שאינם טריוויאליים ושחשובים לצורך מימוש הפעולות ועמידה בדרישות הסיבוכיות. אין לדון בפרטים טריוויאליים (הפעילו את שיקול דעתכם בקשר לזה, ושאלו את האחראי על התרגיל אם אינכם בטוחים). אין לצטט קטעים מהקוד כתחליף להסבר. אין צורך לפרט אלגוריתמים שנלמדו בכתה. כמו כן, אין צורך להוכיח תוצאות ידועות שנלמדו בכתה, אלא מספיק לציין בבירור לאיזו תוצאה אתם מתכוונים. אין (וגם אין צורך) להשתמש בתוצאות של עצי דרגות והלאה.
- **על חלק זה לא לחרוג מ-8 עמודים.**
- והכי חשוב **keep it simple!**

<u>חלק רטוב:</u>

- <u>מומלץ לממש תחילה את מבני הנתונים בצורה הכללית ביותר ורק אז לממש את הפונקציות הנדרשות בתרגיל.</u>
- אנו ממליצים בחום על מימוש **Object Oriented**, ב**++C**, מימוש כזה יאפשר לכם להגיע לפתרון פשוט וקצר יותר לפונקציות אותן עליכם לממש ויאפשר לכם להכליל בקלות את מבני הנתונים שלכם (זכרו שיש תרגיל רטוב נוסף בהמשך הסמסטר).
- פקודת הקימפול שמורצת בgradescope הינה: g++ -std=c++14 -DNDEBUG -Wall -o main.out *.cpp
- חתימות הפונקציות שעליכם לממש ומספר הגדרות נמצאים בקובץ dspotify25b2.h.
- אין לשנות את הקבצים main25b2.cpp ו-wet2util.h אשר סופקו כחלק מהתרגיל, ואין להגיש אותם. ישנה בדיקה אוטומטית שאין בקוד שימוש בSTL, ובדיקה זו נופלת אם מגישים גם את main25b2.cpp.
- את שאר הקבצים ניתן לשנות, ותוכלו להוסיף קבצים נוספים כרצונכם, ולהגיש אותם.
- העיקר הוא שהקוד שאתם מגישים יתקמפל עם הפקודה לעיל, כאשר מוסיפים לו את שני הקבצים main25b2.cpp ו-wet2util.h.
- עליכם לממש בעצמכם את כל מבני הנתונים (למשל **אין להשתמש במבנים של STL** ואין להוריד מבני נתונים מהאינטרנט). **כחלק מתהליך הבדיקה אנו נבצע בדיקה ידנית של הקוד ונוודא שאכן מימשתם את מבני הנתונים שבהם השתמשתם.**
  - בפרט, אסור להשתמש ב-std::iterator, std::vector, std::pair, או כל אלגוריתם של STL, רשימה מלאה של הספריות להן אסור לעשות include נמצאת בקובץ dont_include.txt.
  - **ניתן** להשתמש במצביעים חכמים (Smart pointers כמו shared_ptr), בספריית math או בספריית exception.
  - חשוב לוודא שאתם מקצים/משחררים זיכרון בצורה נכונה (מומלץ לוודא עם valgrind). לא חייבים לעבוד עם מצביעים חכמים, אך אם אתם מחליטים כן לעשות זאת, לוודא שאתם משתמשים בהם נכון. (תזכרו שהם לא פתרון קסם, למשל, כאשר יוצרים מעגל בהצבעות). שימו לב שהעתקת מצביעים חכמים היא איטית מאד ולכן יכולה לגרום ל-timeout. עדיף להעביר shared_ptr כשצריכים by reference.
  - שגיאות של ALLOCATION_ERROR בד"כ מעידות על זליגה בזיכרון.
  - על הקוד להתקמפל ולעבור את כל הבדיקות שמפורסמות לכם ב-gradescope. הטסטים שמורצים באתר מייצגים את הבדיקת אותן נריץ בנתינת הציון, כאשר פרסמנו 5 מתוך 50.
  - אותם טסטים שבgradescope גם מפורסמים כקבצי קלט ופלט, יחד עם סקריפט בשם run_tests.py שנכתב בשביל python 3.6 ומעלה, המאפשר לבדוק את הקוד שלכם. מומלץ לבדוק את התרגיל לוקאלית לפני שמגישים.
  - במידה ויש timeout על אחד מהטסטים זה יחשב ככשלון בטסט. עליכם לכתוב קוד יעיל במידת הסביר. אין לדאוג מכך יותר מידי, הרוב המוחלט של הפתרונות שעומד בסיבוכיות עומד גם בזמני הריצה.
  - **<u>שימו לב:</u>** התוכנית שלכם תיבדק על קלטים רבים ושונים מקבצי הדוגמא הנ"ל. יחד עם זאת הטסטים האלו מייצגים מבחינת אורך ואופן היצירה שלהם את השאר.

אופן ההגשה:

הגשת התרגיל הנה דרך אתר ה-gradescope של הקורס.

חלק הרטוב:

יש להגיש רק את קבצי הקוד שלכם (לרוב קבצי cpp. ,h. בלבד אפשר גם להגיש אותם כקובץ zip)

חלק היבש:

יש להגיש קובץ PDF אשר מכיל את הפתרון היבש. החלק היבש חייב להיות מוקלד.

- **שימו לב כי אתם מגישים את כל שני החלקים הנ"ל, במטלות השונות.**
- לאחר שהגשתם, יש באפשרותכם לשנות את התוכנית ולהגיש שוב. ההגשה האחרונה היא הנחשבת.
- הערכת הציון שמופיעה ב-gradescope אינה ציונכם הסופי על המטלה. הציון הסופי יתפרסם רק לאחר ההגשות המאוחרות של משרתי המילואים.
- במידה ואתם חושבים שישנה תקלה מהותית במערכת הבדיקה ב-gradescope נא להעלות זאת בפורום הפיאצה ונתפל בה בהקדם.

דחיות ואיחורים בהגשה:

- דחיות בתרגיל הבית תינתנה אך ורק לפי תקנון הקורס.
- 5 נקודות יורדו על כל יום איחור בהגשה ללא אישור מראש. באפשרותכם להגיש תרגיל באיחור של עד 5 ימים ללא אישור. תרגיל שיוגש באיחור של יותר מ-5 ימים ללא אישור מראש יקבל 0.
- במקרה של איחור בהגשת התרגיל יש עדיין להגיש את התרגיל אלקטרונית דרך אתר הקורס.
- בקשות להגשה מאוחרת יש להפנות למתרגל האחראי בלבד בכתובת jonathan.gal@campus.technion.ac.il. לאחר קבלת אישור במייל על הבקשה, מספר הימים שאושרו לכם נשמר אצלנו. לכן, אין צורך לצרף להגשת התרגיל אישורים נוספים או את שער ההגשה באיחור.

בהצלחה!