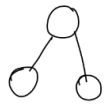
תרגיל בית מספר 1 - יבש
מגישים - מעיין אברמוב 211789342
עומר לוי 315609883

להלן מבנה הנתונים -

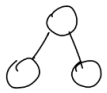
מבנה הנתונים:

Olympics

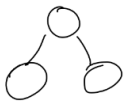
עץ של המדינה



עץ של הקבוצה

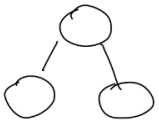


עץ של המשתתפים

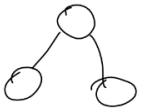


Country

עץ של מאגר המדינה



עץ של נתוני המדינה



Team

עץ כלל המתחרים במדינה לפי id.
עץ כלל המתחרים במדינה בקבוצה.

- עץ SmallbyId ממוין לפי id מכיל את שליש המתחרים עם ה-id הנמוך ביותר.
- עץ SmallbyStrength ממוין לפי המדינה בקבוצה ומכיל את המתחרים בעץ SmallbyId רק על פי המדינה בקבוצה.
- עץ MedbyId ממוין לפי id מכיל את שליש המתחרים עם ה-id הנמוך ביותר.
- עץ MedbyStrength ממוין לפי המדינה בקבוצה ומכיל את המתחרים בעץ SmallbyId רק על פי המדינה בקבוצה.
- עץ BigbyId ממוין לפי id מכיל את שליש המתחרים עם ה-id הנמוך ביותר.
- עץ BigbyStrength ממוין לפי המדינה בקבוצה ומכיל את המתחרים בעץ SmallbyId רק על פי המדינה בקבוצה.

הערה - מיון בקבוצה:
1. לפי strength
2. לפי id

נציין כי יצרנו גם מחלקות עבור המפתחות של Contestant, Team, Country.

AvlTree:

עץ avl מאוזן כמו שהוצג בהרצאות.

המבנה מכיל את המחלקות הבאות -

הגדרה: מיון חדש - מתקיים A > B אם הכוח של מתחרה A גדול מהכוח של מתחרה B. אחרת, אם שני הכוחות של המתחרים שווים אז אם תעודת הזהות של מתחרה A גדולה מתעודת הזהות של מתחרה B (נציין כי בהכרח תעודות הזהות של השחקנים שונות ולכן פעולה ההשוואה הזו מוגדרת היטב).
*כל עוד לא צויין אחרת, המיון בעץ הוא לפי id.

Contestant_Key:
מכיל שלושה שדות: תעודת זהות המתחרה, רמת הכוח שלו והאם נמצא בעץ הממוין לפי id או לפי המיון החדש.

Contestant:
מכיל את המידע הבא על כל מתחרה במערכת : מספר הזהות מתחרה, מספר זהות מדינה, רמת כוח, סוג ספורט, דגל האם נמצא בעץ הממוין לפי id או לפי הסידור החדש ובנוסף, יכיל גם מערך באורך 3 שמטרתו לשמור את תעודות הזהות של המדינות שהוא מתחרה עבורן.

Country_key:

מכיל את תעודת הזהות של כל מדינה.

Country:
מכיל את המידע הבא על כל מדינה: תעודת זהות המדינה, מספר המדליות, פוינטר לעץ עם המתחרים השייכים למדינה ופוינטר לעץ עם הנבחרות השייכות למדינה.

Team_key:

מכיל את תעודת הזהות של הנבחרת.

Team:
מכיל את המידע הבא על כל נבחרת: תעודת זהות הנבחרת, סוג הספורט של הנבחרת, מספר המשתתפים בנבחרת, תעודת זהות של המדינה אליה שייכת הנבחרת, שמונה פוינטרים לעצים:

1. פוינטר לעץ avl של כלל המשתתפים בנבחרת ממויינים לפי id
2. פוינטר לעץ avl של כלל המשתתפים בנבחרת ממויינים לפי המיון החדש.
3. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הכי קטן ממויין לפי id.
4. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הכי קטן ממויין לפי המיון החדש.
5. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הבינוני ממויין לפי id.
6. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הבינוני ממויין לפי המיון החדש.
7. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הגדול ממויין לפי id.
8. פוינטר לעץ avl בגודל של שליש מכמות המשתתפים עם הid הגדול ממויין לפי המיון החדש.
9. המפתח המקסימלי של עץ 3, 5, 7
10. המפתח המינימלי של עץ 3, 5, 7
11. המפתח המקסימלי של עץ 4, 6, 8
12. מערך בגודל 3 של סטטוס הוספה לעצים 3, 5, 7 שתפקידו לתת אינדיקציה האם ניתן להוסיף לעץ מסויים או שנדרש לדלל אותו לפני ההוספה (כלומר לפנות מקום על ידי העברה של המינימום או המקסימום שלו לעצים אחרים).

13. מערך בגודל 3 של סטטוס הסרה לעצים 4, 6, 8 שתפקידו לתת אינדיקציה האם ניתן להסיר מעץ מסוים או שנדרש להגדיל אותו לפי ההוספה (כלומר למלא אותו בתא אחד יותר ע"י הלוואה מהעצים).
14. הכוח האופטימלי של הקבוצה (כלומר הכי חזק שניתן לאחר הסרה של שלושה מתחרים).

Olympicsa1:

מחלקה זו מכילה את המידע על מערכת האולימפיאדה: יש בה את עץ הנבחרות, עץ המשתתפים , עץ של קבוצות.

תיאור הפונקציות :

***בכל מקום שנאמר בו שנעשו פעולות בעץ ב $O(log(x))$ הכוונה למימוש העץ כפי שנלמד בהרצאות ובתרגולים.**

olympics_t():

אתחול כלל השדות ב $O(1)$. מדובר באתחול שלושה עצים ב $O(1)$, היות וזה מספר סופי של פעולות אז סיבוכיות הזמן הינה $O(1)$ במקרה הגרוע.

virtual ~olympics_t():

שחרור של כל הזיכרון אותו הקצנו במעבר בposorder על כל אברי העצים. סה"כ סיבוכיות הסיור היא $O(x)$ כאשר x הוא מספר האיברים בעץ. לכן עבור כלל המבנים בתוכנית סיבוכיות הפונקציה הינה $O(n + m + k)$.

StatusType add_country(int countryId,int medals):

ניצור מדינה חדשה ובשדות שלה יהיו שני עצים ריקים ומספר המשתתפים יאותחל ל0. כל אלו מספר סופי של פעולות ב$O(1)$ ונעדכן לה את השדה מספר המדליות ב $O(1)$. נכניס את האובייקט אל עץ המדינות בסיבוכיות של $O(logk)$ במקרה הגרוע. לכן סה"כ סיבוכיות פונקציות זו הינה ב$O(logk)$ במקרה הגרוע.

StatusType remove_country(int countryId):

נמצא את המדינה בעץ המדינות בסיבוכיות זמן של $O(logk)$ במקרה הגרוע. נבדוק האם מספר המשתתפים וגם הנבחרות בה הוא 0 (בדיקה זו מתבצעת ב $O(1)$). במידה ומס' המשתתפים או מספר הנבחרות הוא 0 נוציא את האיבר מהעץ ב$O(logk)$ במקרה הגרוע. אחרת, לא נמחק את המדינה ונחזיר סטטוס כשלון ב$O(1)$.

בשני המקרים מבוצע מספר סופי של פעולות ב $O(logk)$ וניתן להזניח את הפעולות ב $O(1)$ ולכן הסיבוכיות הינה $O(logk)$ במקרה הגרוע.

StatusType add_team(int teamId,int countryId,Sport sport):

ניצור מדינה חדשה ונאתחל את השדות שלה באופן הבא: את שמונת עצים נאתחל להיות עצים ריקים בסיבוכיות זמן של $O(1)$ כל אחד, את השדות של המפתחות המקסימליים והמנימליים להיות 0. סה"כ יצירת מדינה כללה מספר סופי של פעולות אשר כל אחת מהן היא בסיבוכיות זמן של $O(1)$ במקרה הגרוע ולכן אתחול מדינה הוא ב$O(1)$ במקרה הגרוע. נוסיף את המדינה לעץ המדינות ב $O(logm)$, נעדכן בה את השדה של המדינה ששייכת אליה ב$O(1)$. עתה נוסיף את הנבחרת אל עץ המדינות (חיפוש המדינה ב$O(logk)$ והוספה של הנבחרת ב$O(logm)$.

סה"כ נסכם כי סיבוכיות הפונקציה הינה $O(logk + logm)$.

StatusType remove_team(int teamId):

נבדוק האם מספר המתחרים וגם מספר הנבחרות בה הוא 0. במידה וכן, נסיר את הנבחרת מעץ הנבחרות של התכנית ב$O(logm)$ אחרת, לא נסירה ונחזיר סטטוס כשלון ב$O(1)$ במקרה הגרוע.

סה"כ נסכם כי במקרה הגרוע סיבוכיות הפונקציה הינה $O(logm)$ במקרה הגרוע.

StatusType add_contestant(int contestantId ,int countryId,Sport sport,int strength):

ניצור מתחרה חדש ונאתחל את השדות שלו בערכים הדיפולטיים ב$O(1)$ במקרה הגרוע (כי זה מספר סופי של פעולות). נחפש את המדינה בסיבוכיות זמן של $O(logk)$ במקרה הגרוע. נבדוק האם המתחרה נמצא שם. אם כן, נחזיר שגיאה ב$O(1)$. אחרת, נכניס את המתחרה אל עץ המתחרים בסיבוכיות זמן של $O(logn)$. נעדכן את השדה של המתחרה כי שייך אל המדינה הנתונה ב $O(1)$.

סה"כ סיבוכיות הפונקציה הינה $O(logk + logn)$.

StatusType remove_contestant(int contestantId):

נחפש את המתחרה בעץ המתחרים בסיבוכיות זמן של $O(logn)$. נבדוק בשדה המערך השייך למתחרה האם הוא נמצא בנבחרת כלשהי. נציין כי המערך הוא באורך 3, אשר הוא מספר סופי ולכן הסיור על המערך יתבצע בסיבוכיות זמן של $O(1)$ במקרה הגרוע. במידה והוא שייך לנבחרת כלשהי נחזיר שגיאה ב $O(1)$. אחרת, נסיר אותו מהעץ בסיבוכיות זמן של $O(logn)$ במקרה הגרוע.

StatusType add_contestant_to_team(int teamId,int contestantId):

נחפש את הנבחרת ואת המתחרה בעץ הנבחרות ובעץ המתחרים בסיבוכיות זמן של $O(logn + logm)$. נבדוק אם סוגי הספורט וגם תעודות הזהות של המדינות שלהם זהים ב$O(logk)$ על פי השדות שלהם ב $O(1)$. עתה, נחפש האם המתחרה נמצא בעץ הנבחרת בסיבוכיות זמן של $O(logn)$ (מכיוון שבמקרה הגרוע ביותר כל n המשתתפים נמצאים בנבחרת הזו). אם כן, נחזיר שגיאה. אחרת, נוסיף את המתחרה אל הנבחרת בסיבוכיות זמן של $O(logn)$. (נציין כי טרם ההוספה נתבונן בשדה שהוא מערך של סטטוס ההוספה ונבדוק האם הסטטוס הוא 1. במידה והוא 1 לא ניתן להוסיף, ואז נצטרך לדלל את העץ בסיבוכיות זמן של $O(logn)$ ע"י העברה של המקסימום או המינימום לעץ אחר כתלות במקרה. לאחר הדילול נעדכן את המערך (במקום הi יהיה המקסימום בהפרשים בין כמות התאים בעץ ה i לבין כמות התאים בעצים האחרים. נציין כי זה החישוב הזה הוא מתבצע 3 פעמים, אחת עבור כל עץ. היות שסיבוכיות החישוב של ההפרש היא ב $O(1)$ במקרה הגרוע ומדובר במספר סופי של פעולות, סיבוכיות העדכון של המערך היא ב $O(1)$ במקרה הגרוע.)

כמו כן, נמצא את המדינה בה נמצא המשתתף ונעדכן את שדה המערך של המשתתף בו מופיעות המדינות בהן הוא חבר. סיבוכיות החיפוש בעץ המדינות היא ב $O(logk)$ אבל נבחין שנבחרת שייכת למדינה בעת ההכנסה לתוכנית ולכן נבחרת אחת לפחות משוייכת לכל מדינה. מכאן נסיק כי

$k \leq m$

ולכן סיבוכיות החיפוש של מדינה היא ב$O(logm)$ במקרה הגרוע.

נציין כי לאחר כל הוספה לאחד משלושת העצים הממויינים לפי id נדרש להוסיף את אותו התא גם אל העץ המתאים לו משלושת העצים הממויינים לפי הstrength. כמו כן, נזכור לעדכן גם את השדה "הכוח האופטימלי של קבוצה" ע"י הסרה של שלושה תאים מהעצים בכל אחת מהקומבינציות האפשריות, חישוב המקסימום בכל אחד מהעצים והחזרה של התאים הללו חזרה אל העצים כדי לשמר את המידע של המבנה. סה"כ מדובר במספר סופי של פעולות בסיבוכיות זמן של $O(logn)$ במקרה הגרוע.

נסכם כי סיבוכיות הפונקציה היא $O(logn + logm)$ במקרה הגרוע.

StatusType remove_contestant_from_team(int teamId,int contestantId):

נחפש בעץ הנבחרות ובעץ המתחרים האם המדינה והנבחרת קיימות ובאמת המתחרה נמצא בא. כל זאת בסיבוכיות של $O(logn + logm)$ במקרה הגרוע. במידה ולא, נחזיר שגיאה בסיבוכיות זמן של $O(1)$ במקרה הגרוע. אחרת, נוציא את המתחרה מהעץ של המתחרים הכללי של התוכנית ומהעץ של הנבחרות אליהן הוא שייך. עתה נוציא מהעץ הכללי של הנבחרת את המתחרה הממויין לפי id וגם מהעץ הכללי של הנבחרת הממויין לפי המיון החדש. כמו כן, נבדוק מאיזה משלושת העצים הממויינים לפי id נוציא את המתחרה לפי השוואה אל הערך המקסימלי בעצים האלו. נבחין כי

פעולות ההשוואה הן מספר פעולות סופי בסיבוכיות של $O(1)$ במקרה הגרוע וההסרה מן העצים הינה בסיבוכיות זמן של $O(logn)$ במקרה הגרוע. נציין כי כל הוצאה מאחד מהעצים bigById, MedById, SmallById מתבצעת בעץ הזהה לו הממויין לפי המיון החדש, bigByStrength, MedByStrength, SmallByStrength בהתאמה.
כל מתחרה שייך לכל היותר ל3 נבחרות ולכן יש לכל היותר 3 מופעים של אותו אובייקט לעדכן את שדה מערך הנבחרות אליו הוא שייך. כל חיפוש שלו בעץ הנבחרות יהיה בסיבוכיות זמן של $O(logn)$
.
.במקרה הגרוע וכל עדכון של השדה יהיה בסיבוכיות זמן של $O(1)$. נבחין כי מספר הפעולות המתבצעות כאן הוא סופי.
בנוסף, נעדכן גם את עץ הנבחרות של המדינה ואת עץ המתחרים של המדינה בסיבוכיות של $O(logn + logm)$ בדומה להסבר למעלה.
לאחר כל הכנסה נעדכן את שדות המתחרים המקסימליים בכל אחד מהעצים (לפי id ולפי strength) באמצעות השוואה למקסימום הקודם ב $O(1)$.
בנוסף, נבדוק מתוך כל הקומבינציות האפשריות להסרה של שלושה תאים מכל אחד משלושת העצים מתי ההסרה תניב ערך כוח מקסימלי של קבוצה ונעדכן את השדה "הכוח האופטימלי של קבוצה".
סה"כ מדובר במספר סופי של קומבינציות אפשריות להסרה ולכן סיבוכיות עדכון השדה "הכוח האופטימלי של קבוצה" היא ב $O(logn)$ במקרה הגרוע (כי מס' המשתתפים בכל עץ במקרה הגרוע הוא n). נזכור להוסיף את שלושת התאים חזרה לעץ לאחר בדיקת כל קומבינציה של הסרה על מנת לשמר את המידע של המבנה.

נסכם כי מתבצעות מספר סופי של פעולות בסיבוכיות זמן של $O(logn + logm)$ במקרה הגרוע.

StatusType update_contestant_strength(int contestantId ,int change):

נמצא את המתחרה בעץ הכללי של התוכנית בסיבוכיות זמן של $O(logn)$ במקרה הגרוע. אחר בדיקה האם המתחרה קיים, נסיר אותו מהעץ ונחזיר אותו חזרה עם השדה strength שהוא מעודכן(יצירת עותק של אובייקט המתחרה ועדכון של השדה שלו הוא ב$O(1)$ במקרה הגרוע. בתהליך זה השתמשנו בפונקציות -

remove_contestant_from_team(int teamId, int contestantID)
Add_contestant_to_team(int teamId, int contestantId)

שכל אחת מהן מעדכנת את העצים הבאים: עץ המתחרים הכללי של התוכנית, עץ הקבוצות הכללי של התוכנית (אליהן שייך המתחרה, במידה ושייך), עץ המתחרים של המדינה אליה שייך ועץ הנבחרות של המדינה(אליהן שייך המתחרה, במידה ושייך).

הסיבוכיות של פונקציות אלו היא $O(logn + logm)$ במקרה הגרוע ולכן זו הסיבוכיות הנדרשת.

output_t<int> get_strength(int contestantId):

נחפש את המתחרה בעץ המתחרים בסיבוכיות זמן של $O(logn)$ במקרה הגרוע.לאחר בדיקה האם המתחרה קיים, נחזיר את שדה הכוח שלו בסיבוכיות זמן של $O(1)$. לכן, אם נסכם, סיבוכיות הזמן של הפונקציה הזו הינה ב $O(logn)$ במקרה הגרוע.

output_t<int> get_medals(int countryId):

נחפש את המדינה בעץ המדינות בסיבוכיות זמן של $O(logk)$ במקרה הגרוע.לאחר בדיקה האם המדינה קיימת, נחזיר את שדה המדליות שלה בסיבוכיות זמן של $O(1)$. לכן, אם נסכם, סיבוכיות הזמן של הפונקציה הזו הינה ב $O(logk)$ במקרה הגרוע.

output_t<int> get_team_strength(int teamId):

נחפש את הנבחרת בעץ הנבחרות בסיבוכיות זמן של $O(logm)$.לאחר בדיקה האם הנבחרת קיימת, נסכום של שלושת השדות ששמרנו מראש והם המפתחות המקסימליים של כל אחד מהעצים ממויין לפי המיון החדש. הגישה לשדות והסכימה היא בסיבוכיות זמן של $O(1)$ במקרה הגרוע.
נציין כי נבדוק מראש באמצעות שדה מספר המשתתפים בנבחרת האם המספר מתחלק ב-3 ללא שארית ב $O(1)$ במקרה הגרוע. במידה וכן, נבצע את השלב לעיל, אחרת, נחזיר שהכוח הוא 0.
סה"כ הסיבוכיות של הפונקציה היא ב $O(logm)$ במקרה הגרוע.

StatusType unite_teams(int teamId1,int teamId2):

במידה ומספרי הזהות של הנבחרות לא תקין או שלאחר חיפוש בעץ ב$O(logm)$ נגלה כי אלו אינן קיימות, נחזיר שגיאה בסיבוכיות זמן של $O(1)$ במקרה הגרוע. אחרת, נאתר את שתי הנבחרות בסיבוכיות זמן של $O(logm)$ במקרה הגרוע. ניצור שני מערכים ריקים על פי האלגוריתם לאתחול מערך הנלמד בהרצאה בסיבוכיות זמן של $O(1)$ במקרה הגרוע. נעביר כל אחת מהמדינות למערך באמצעות סיור inorder בסיבוכיות זמן של $O(n_{Team1Id})$ עבור העץ של הנבחרת הראשונה ובסיבוכיות זמן של $O(n_{Team2Id})$ עבור העץ של הנבחרת השנייה. עתה נאתחל מערך נוסף ריק ב $O(1)$ באורך $n_{Team1Id} + n_{Team2Id}$. עתה נמזג את שני המערכים אל המערך הנ"ל באמצעות אלגוריתם merge לפי id של המתחרים בסיבוכיות זמן של $O(n_{Team1Id} + n_{Team2Id})$. עתה ניצור עץ כמעט שלם מהמערך הממויין בסיבוכיות זמן של גודל המערך כפי שנלמד בהרצאה, לכן סיבוכיות הזמן ליצירת העץ היא $O(n_{Team1Id} + n_{Team2Id})$ במקרה הגרוע.את שלב הInorder אל תוך שני מערכים ולאחר מכן ביצוע merge נעשה לכל אחד משמונת העצים ששמרנו בשדות 1-8(בעמוד 2 ניתן לראות את השדות) ונעדכן את שדות 9-11 על פי מציאת מינימום ומקסימום בכל אחד משמונת העצים בסיבוכיות זמן של $O(log(n_{Team1Id} + n_{Team2Id}))$ במקרה הגרוע (מכיוון שיכולים להיות לכל היותר n מתחרים באותו העץ ועתה נוכל להשתמש בעץ כמעט שלם כעץ avl באמצעות יצירת עץ avl ריק והשמה של השורש להיות השורש של העץ כמעט שלם שבנינו, ועתה הסיבוכיות היא כמו בעץ avl של $O(log(n_{Team1Id} + n_{Team2Id}))$ בגרוע). את המערכים בשדות 12, 13 נעדכן לפי חיסור בין מספר התאים בעץ הi לבין מספר התאים בעצים האחרים ובחירת התוצאה המקסימלית והשמה שלה במערך סטטוס הוספה במיקום הi (שדה 12). ובחירת התוצאה המינימלית והשמה שלה במערך הסטטוס הסרה במיקום ה i (שדה 13).
במידה ויש כעת הפרש בין כמות התאים בעצים שגדול ממש מ-1, נדלל את העצים בהתאמה (נעביר לאחד מהעצים השכנים את המינמום או את המקסימום בהתאם למצב). לסיום, נעדכן את שדה ה"כוח האופטימלי של קבוצה" באמצעות האלגוריתם שתיארנו בפונקציה add_contestant_To_team לעדכון שדה זה. עדכון השדות היה מספר סופי של פעולות ב $O(1)$ וב $O(log(n_{Team1Id} + n_{Team2Id}))$ ולכן סה"כ פונקציה זו היא במימוש בסיבוכיות זמן של

$$O(logm) + O(log(n_{Team1Id} + n_{Team2Id})) + O(n_{Team1Id} + n_{Team2Id})$$

הניתנת לחסימה ע"י

$$O(logm) + O(n_{Team1Id} + n_{Team2Id})$$

כנדרש.

StatusType play_match(int teamId1, int teamId2):

נבדוק האם הנבחרת משתתפות באותו הספורט באמצעות שני חיפושים בעץ הנבחרות הכללי של התוכנית Olympics בסיבוכיות זמן של $O(logm)$ במקרה הגרוע. עתה ניגש לשדה מס' הזהות של המדינה של אותן הנבחרות בסיבוכיות זמן של $O(1)$ במקרה הגרוע.נבצע חיפוש למדינות של כל אחת מהנבחרות (במידה והמדינה אינה זהה, אחרת נבצע רק חיפוש אחד) בסיבוכיות זמן של $O(logk)$ במקרה הגרוע. עתה לכל אחת מהנבחרות נחשב את הכוח של הקבוצה, שהוא הסכום של שלושת השדות המצויינים בסעיף 11 (בדף 2 לעבודה), זהו מספר סופי של פעולות ב $O(1)$ במקרה הגרוע הודות לכך שעדכנו את השדות האלו לאחר כל שינוי שהיה בעצים מראש. עתה נבדוק האם הכוח בתוספת המדליות של קבוצה 1 גדול מהכוח בתוספת המדליות של קבוצה 2. במידה וכן, נוסיף מדליה לקבוצה 1 באמצעו עדכון שדה המדינה אליה שייכת (בסיבוכיות זמן של $O(1)$ במקרה הגרוע עבור ההוספה), או להיפך אם התוצאה היא שקטנה יותר. במידה ויש שוויון, לא נעשה שום שינוי ונחזיר סטטוס הצלחה בלבד. כמו כן בשני המקרים הראשונים גם נחזיר סטטוס הצלחה.

סה"כ ביצענו מספר סופי של פעולות ב $O(1)$, $O(logm)$, $O(logk)$ במקרה הגרוע ולכן הסיבוכיות של הפונקציה הזו הינה $O(logm + logk)$ במקרה הגרוע כנדרש.

output_t < int > austerity_measures(int teamId):

ראשית, נחפש את הנבחרת בעץ הנבחרות הכללי של התוכנית ונבדוק האם היא קיימת, זאת בסיבוכיות של $O(logm)$. עתה, ניגש לשדה כמות המשתתפים של אותה הנבחרת ונבדוק האם הוא קטן מ-3, במידה וכן, נחזיר שגיאה ב $O(1)$ במקרה הגרוע. באותו האופן, נבדוק האם מספר המשתתפים מתחלק ב-3 ללא שארית. במידה ולא, נחזיר שגיאה ב$O(1)$ במקרה הגרוע. אחרת, נחזיר את השדה "הכוח האופטימלי של קבוצה" ששמרנו מראש ועדכנו אותו לאחר כל הוצאה, הכנסה או איחוד קבוצות. החזרת השדה הינה בסיבוכיות של $O(1)$ במקרה הגרוע ולכן סיבוכיות פונקציה זו היא $O(logm)$ במקרה הגרוע כנדרש.