תרגיל ריצה 4 – מבנה נתונים ב' –קוד הופמן

מטרת תרגיל זה היא לקבל טקסט שמייצג מידע (דיון וכדו') ולקודד אותו על פי קוד הופמן על מנת לייצג את הטקסט בצורה דחוסה יותר. בתוכן המקודד יש לשמור ראשית כל את המבנה של העץ של הופמן על מנת שאפשר יהיה לפענח את התוכן המקודד ולחזור לתוכן המקורי.

מטרת התרגיל:
בהינתן טקסט, יש ליצור טקסט מקודד עבור מידע זה.
הטקסט המקודד יראה כך:

1. שורה ראשונה - יש לשמור את מספר התווים השונים שנמצאו בקובץ, n.
2. שורה שניה: יש לשמור את האותיות על פי סדר הופעתן בעלים של עץ הקידוד.
3. שורה שלישית: יש לשמור את מבנה העץ (דורש בדיוק 1-n2 תווים).
בעזרת שתי השורות האחרונות ניתן לשחזר את הקידוד של כל אחת מהאותיות המקודדות בעץ.
4. שורה רביעית: יש לשמור את הטקסט המקודד לסיביות (על פי הקידוד של העץ שנשמר בשורות 1,2,3 של הטקסט המקודד).

על מנת להקל על התכנות, נניח כי הטקסט לא מכיל "תווים לבנים": רווח, טאב, ENTER, וכו'... זה יאפשר לנו לקרוא את הטקסט בעזרת cin.

לדוגמא, אם הטקסט של הקלט היה נראה כך:

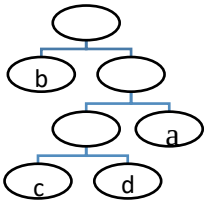
```
a
dbaabdcb
b
```

1. בשורה הראשונה בקובץ יהיה 4 (יש בקובץ ארבעה תווים שונים: a, b, c, d).

בשביל לכתוב את האותיות לפי סדר הופעתן בעץ ואת מבנה העץ נבנה את עץ הקידוד:
השכיחויות:

c-1, d-2, a- 4, b-6

במקרה זה העץ יראה כך:



העץ ייבנה תוך שימוש באלגוריתם של הופמן, תוך שימוש בתור עדיפויות, כפי שלמדנו.
עזרה לגבי אופן קידוד העץ נמצאות בהמשך התרגיל.
לאחר בניית העץ ניתן להמשיך לבנות את הקובץ:

2. בשורה השניה יופיעו התווים על פי סדר הופעתם בעץ שנוצר: bcda.
3. בשורה השלישית יהיה שמור מבנה העץ – 0100111 (הסבר מפורט על קידוד מבנה העץ בתחתית העמוד).
4. ובשורה הרביעית יהיה הטקסט המקודד- $_{11}{}^{101}{}_{0}{}^{11}{}_{11}{}^{0}{}_{101}{}^{100}{}_{0}{}^{0}$ (התווים יופיעו ברצף: 1110101111010110000 רשמנו כאן את התווים לסירוגין בכתב עילי ותחתי על מנת להסביר מה יש ברצף).

בס"ד

סיכום: כך יראה הפלט עבור הקלט שבדוגמא:

```
4
bcda
0100111
1110101111010110000
```

כמובן, בהינתן טקסט מקודד:

```
4
bcda
0100111
1110101111010110000
```

הטקסט המפוענח יראה כך:

adbaabdcbb

בעמוד הבא מוסבר איך ניתן יהיה לפענח את הקידוד של הטקסט.

על התכנית לבצע אחת משתי הפעולות הבאות על פי בחירת המשתמש:

1. קליטת טקסט, דחיסתו, והדפסת הטקסט המקודד, כפי שתואר למעלה.
2. קליטת טקסט של טקסט דחוס (שורות 1-4 למעלה, באופן שבו הטקסט נדחס), פענוח העץ של הופמן, ופענוח הטקסט המקורי.

דוגמת הרצה:

```
enter 1 to encode a text
enter 2 to decode a text
enter 3 to exit
1
enter the original text
aaabaaac
The encoded string is:
3
bca
00111
1110011101
2
enter n 3
enter the letters bca
enter the encoded structure 00111
```

enter the encoded text 1110011101
The decoded string is: aaabaaac
3

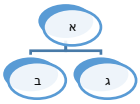
<u>נספח: קידוד מבנה עץ הופמן בצורה אופטימלית מבחינת אורך הקידוד:</u>

<u>קידוד מבנה העץ</u>- שליחת צורת העץ בלי לשלוח את הערכים של העלים (אנחנו בונים על זה שסדר תכנם של העלים כבר ידוע, הוא נשמר בנפרד מצורת העץ!).

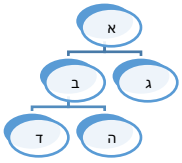
על מנת לשלוח את העץ, כיוון שמדובר על עץ הופמן, וזהו קוד תחיליות אופטימלי - כל הגדרה של צעד שמאלה, מגדירה גם צעד ימינה, ולכן יש צורך לשלוח רק את הצעדים שמאלה, ובנוסף לשלוח ביט סימן כל פעם שהגענו לעלה.

נניח הקוד הבא: 0010111.

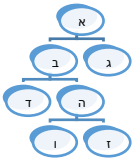
ה- 0 הראשון אומר לבנות עץ עם שלושה קודקודים (תמיד הצעד הראשון שמאלה יגדיר אותו עץ, אם מסמנים את הקודקודים באותיות, עשינו צעד שמאלה אנחנו ב- ב'):



ממשיכים עם ב'- אם התו הבא היה 1- היינו מפסיקים עם העלה ב' ועוברים לעלה הבא משמאל, ג', אך הוא 0- זה אומר שלקדקוד (ב') אנחנו מגדירים עוד בן שמאלי- ואוטומטית גם ימני. סיכום ביניים: הקוד 00 מגדיר את העץ הבא:



עכשיו יש 1, זה אומר – הגענו לעלה. אנחנו ב- ד' והוא עלה- ממשיכים להגדיר מ- ה'. אם עכשיו היה עוד 1 – גם הוא היה עלה, אך כיוון שיש 0 זה אומר שלקדקוד ה' יש בן שמאלי (וימני...), והקוד 0010 מתאים לעץ:



בשלבים הבאים- שלוש אחדות- ז"א אנחנו ב- ו' והוא עלה, אז עוברים ל- ז' גם הוא עלה, עוברים ל- ג', שגם הוא עלה. זהו. טיפלנו בכל העץ.

<u>נספח: קידוד מבנה עץ הופמן בצורה אופטימלית מבחינת אורך הקידוד:</u>

1. נגדיר מחלקה HuffmanNode עבור "קודקוד בעץ הופמן" המכילה:
   str: מחרוזת של אות אחת או יותר.
   frequency: התדירות של אות זו, או סכום התדירויות של כל העלים הנמצאים בעלי תת העץ.
   left: מצביע left לבן שמאלי של עץ הופמן.
   right: מצביע right לבן ימני של עץ הופמן.

2. נגדיר מחלקה שתקרא compareNode ומטרתה השוואת 2 מצביעים לקדקודים בעץ הופמן, כאשר ההשוואה תעשה על פי התדירות של התווים. תוכן המחלקה:

```
class compareNode{
public:
bool operator()(HuffmanNode* const & n1, HuffmanNode* const & n2)
{
return n1->frequency > n2->frequency;
}
};
```

מטרתה להחזיר את תוצאת ההשוואה של שני קודקודי עץ הופמן (על פי התדירות שלהם).
כאשר יהיה צורך להגדיר תור עדיפויות של מצביעים לקודקודים, הוא יוגדר באופן הבא:

```
compareNode> pQueue;   priority_queue<HuffmanNode*, vector<HuffmanNode*>,
```

על מנת שסדר הקודקודים יוגדר על פי השכיחות שלהם.
המחלקה מוגדרת בספריה <queue>

תזכורת:

- כיוון שהמחלקה CompareNode משתמשת בשדות פרטיים של HuffmanNode יש להגדיר את המחלקה CompareNode בתוך המחלקה של HuffmanNode כחברה (friend class).
- מותר להצהיר על 2 המחלקות האלו באותו קובץ h.

צריך להצהיר על המחלקה של CompareNode לפני ההגדרה של המחלקה של HuffmanNode.

בהצלחה רבה!