המחלקה להנדסת מערכות תקשורת תכנות מתקדם סמסטר ב' 2016 תרגיל בית מספר 2

**תרגיל בית מספר 2 – מבנים ומבני נתונים**

# מבוא

בתרגיל זה נתרגל שימוש במבנים על מנת ליצור מהם מבני נתונים וכן נעסוק בהקצאת זיכרון בצורה דינמית. אנו נממש קוד לדחיסת טקסט על ידי שימוש בקוד האפמן.

**קוד האפמן:** מאפשר דחיסת טקסט יעילה. באמצעותו ניתן להשתמש במספר קטן של תווים כדי לאכסן את הטקסט המקורי. כך לדוגמא, טקסט שאורכו במקור הוא 100 תווים, ניתן לשמור בזיכרון באמצעות שימוש ב70 תווים בלבד (זיכרו, משקלו של תו הוא byte אחד בלבד).

ללא דחיסה כל תו דורש 8 סיביות (bit) – הטקסט 'he' לדוגמא מיוצג על ידי רצף הסיביות הבא:01101000 01100110. קוד האפמן מתאים לכל תו אשר קיים בטקסט שנרצה לדחוס רצף סיביות אחר, ייחודי. לדוגמא:הקוד האפמן עשוי להקצות את רצף הסיביות 110 עבור התו 'h' ואת רצף הסיביות 101 עבור התו 'e'. כעת קידוד הטקסט he ידרוש 6 סיביות בלבד – 110 101. הרווחים לשם נוחות הקריאה בלבד.

**עץ האפמן:** רצפי סיביות עבור כל תו מחושבים באמצעות עץ בינארי הייחודי לכל טקסט שנרצה לקודד. בתרגיל זה נקרא לעץ- עץ האפמן Huffman tree. כל צומת בעץ מכיל שני שדות:

1. תדירות (int freq)
2. תו (char ch)

כמו כן גם מצביעים לאב, לבן הימני והבן השמאלי.

typedef struct HuffmanTree {

int freq;

char ch;

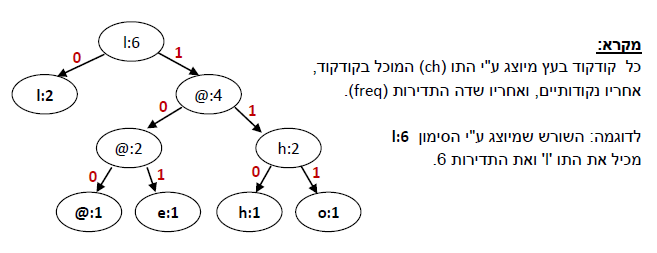
struct HuffmanTree\* parent;

struct HuffmanTree\* left;

struct HuffmanTree\* right;

} HuffmanTree;

בהינתן עץ האפמן, ניתן לקודד טקסט כרצף סיביות או לתרגם את רצף הסיביות חזרה לטקסט טרם הדחיסה. הרי דוגמא לעץ האפמן תקין.



כמו כן נשייך את הסיבית 0 לכל מעבר מצומת לבן השמאלי, 1 עבור מעבר מצומת לבן ימני. **הקידוד של כל תו יהיה רצף הסיביות הנוצר מ"טיול" משורש העץ עד לעלה המכיל את התו.** בהתאם לכך קידוד התו e יהיה 101. אם נרצה לקודד את הטקסט hello@ נקבל את הרצף הבא:



**פעולת הפיענוח:** יש לעקוב אחרי העץ מראשו עד להגעה לעלה. התו בעלה יהיה הסימן שקודד בטקסט המקורי וכך על ידי מעבר על כלל עלי העץ ניתן לייצר מילון פענוח.\

**כיצד בונים עץ האפמן:**  אנו נבנה את העץ על ידי פרוצדורה הממזערת את מספר הסיביות הנדרשות לקידוד של הטקסט, מכאן שטקסטים שונים יכולים לגרום לבניית עצי האפמן שונים.

כדי לבנות את העץ עבור טקסט נתון ניעזר ברשימה מקושרת חד כיוונית. כל חוליה ברשימה תכיל מצביע ל Huffman tree ומצביע לחוליה הבאה.

typedef struct LinkedListNode {

HuffmanTree \*tree;

struct LinkedListNode \*next;

} LinkedListNode;

נגדיר יחס "עדיפות" בין שני עצים.

נאמר שהעץ tree2 עדיף על עץ tree1 אם יתקיימו:

1. התדירות שתופיע בשורש העץ tree2 גבוהה מזו של tree1.
2. במידה והתדירויות שוות- ערך הascii של התו ch בשורש העץ tree2 גבוה מזה של התו בשורש העץ tree1.

\*שימו לב שכדי לקבל ערך ascii של תו יש להמירו לunsigned char .

**בניית העץ )עבור טקסט נתון( תתבצע להלן:**

1. עבור כל תו ch בטקסט צור חוליה – linkedListNode המכילה עץ ובו קודקוד יחיד המכיל את התו ch ואת מספר הפעמים בו הופיעה בכל הטקסט (שדה freq).
2. צור רשימה מקושרת ממויינת בסדר עדיפויות עולה כפי שהוגדר שאיבריה הם החוליות שייצרת בסעיף הקודם. חוליה המכילה עץ בעדיפות נמוכה תהא קודמת לחוליה המכילה עץ בעל עדיפות גבוהה.
3. צור קודקוד huffmanTree חדש שבנו השמאלי הוא העץ עם העדיפות הנמוכה ביותר (מבין העצים המצויים בחוליות של הרשימה המקושרת) ובנו הימני הוא העץ עם העדיפות השנייה הנמוכה ביותר. תדירות הקודקוד החדש תהא סכום תדירויות הבנים שלו והתו אוו הוא מכיל יהיה שווה לערך התו של בנו השמאלי.
4. צור חוליה חדש LinkedListNode המכילה את הקודקוד שיצרת בסעיף 3. הוסף את החוליה לרשימה המקושרת. דאג שהרשימה המקושרת תמיד תהא ממוינת.
5. הסר מן הרשימה המקושרת את שתי החוליות המכילות את העצים עם העדיפות הנמוכה בהם השתמשת בסעיף 3.
6. חזור על סעיף 3 עד שברשימה המקושרת תישאר חוליה אחת בלבד, העץ המוכל בחוליה זו יהיה עץ האפמן תקין.

* דוגמא לעץ האפמן ניתן לראות באיור הקודם, עץ זה נוצא עבור הטקסט hello@.

# הערות כלליות

יש להגיש את התרגיל במקום המתאים באתר הקרוס במודל, ההגשה עד התאריך **5/5/2016** בשעה 23:55

* את התרגיל יש להגיש בזוגות
* הקובץ אותו אתם מגישים יהיה קובץ קוד המקור) קובץ עם סיומת c )
* הקפידו על הזחות נכונות והערות.
* אין להשתמש במספרי קסם (magic numbers), יש להגדירם כקבועים.
* רשמו הערות אינפורמטיביות על קטעי קוד משמעותיים ו/או מסובכים. כמו כן יש לתעד בהערות כל פונקציה, מה היא מבצעת, מה ערכי ההחזרה וכן מהם הפרמטרים שהיא מקבלת.
* **נא לכתוב בבלוק של הערה בתחילת הקובץ אותו אתם מגישים את שמות המגישים בצורה הבאה**

/\* Assignment: 1

Author1: Israel Israeli,

ID: 01234567

Author2: Learsi Learsi,

ID: 7654321

\*/

**חלק א:**

**משימה 1**

למשתמש יוצג תפריט בן שלוש אפשרויות.

1. **Create a Huffman tree from text**
2. **Deserialize a Huffman tree**
3. **Exit**

**Enter your choice:**

המשתמש יתבקש להזין קלט של מספר שלם בין 1 ל 3. לא ניתן להניח דבר על הקלט אותו יקליד המשתמש מלבד שאורך המחרוזת אותה יקליד קטן מ10 תווים. התפריט יוצג שוב עד קבלת קלט חוקי.

**משימה 2 – Create a Huffman tree from text:**

באופציה זו יתבקש המשתמש להזין מחרוזת שאורכה לא יעלה על 10000 תווים המסתיימת בתו @. התו @ יופיע פעם אחת בלבד ובסוף המחרוזת. ניתן להניח תקינות המחרוזת. ממחרוזת זו עליכם לבנות עץ האפמן בדיוק כפי שהוגדר לעיל ולהחזיק מבנה נתונים זה בזיכרון.

**משימה 3 - Deserialize a Huffman tree:**

בבחירה באופציה זו תקלט מחרוזת מן המשתמש שאורכה לא יעלה על 15 תווים. המחרוזת תהיה בנויה מתו כלשהו יחיד אחריו הסימן ':' ואחריו מספר חיובי שלם שמשמעותו היא מספר הפעמים שהתו הופיע בטקסט. הקלט יחזור על עצמו עד קבלת מחרוזת ריקה (כלומר, המשתמש לא הכניס קלט כלל ולחץ ישירות על enter).

לדוגמא: הקלט **e:1** אחר כך **@:1** אחר כך **h:1** אחר כך **l:2** אחר כך **o:1** ולבסוף לחיצה על enter ללא קלט. בהינתן קלטים אלו יש לבנות Huffman tree המתאים לתדירויות הנתונות עבור כל תו. ניתן להניח את תקינות הקלט (פורמט נכון ואין חזרה על תווים) מבנה נתונים זה יש לשמור בזיכרון.

**חלק ב:**

**משימה 1**

למשתמש יוצג תפריט בן שלוש אפשרויות.

1. **Encode text using the Huffman Tree**
2. **Bit-encode text using the Huffman Tree**
3. **Decode using the Huffman Tree**
4. **Bit-Decode using the Huffman Tree**
5. **Serialize the Huffman Tree**
6. **Free the in-memory Huffman Tree**

**Enter your choice:**

גם כאם לא ניתן להניח את חוקיות הקלט. ניתן להניח כי אורך המחרוזת שנקלטה אינה גדולה מ10 תווים.

1. **Encode text using the Huffman Tree**

תקלט מחרוזת מהמשתמש שאורכה לא יעלה על 10000 תווים. הפלט יהיה רצף סיביות המתקבל מקידוד התווים על פי העץ שבזיכרון עם תוספת מינימלית של אפסים לרצף כל שמספר הסיביות יהיה כפולה של שמונה. לדוגמא עבור העץ שבאיור הקודם אם נקודד את **lo@**  נקבל את רצף הסיביות הבא: **0111100**. מאחר שמספר הסיביות לקידוד הטקסט הינו 7 סיביות ואינה כולה של שמונה נוספה הסיבית 0. ניתן להניח כי הקלט חוקי.

1. **Bit-encode text using the Huffman Tree**

תקלט מחרוזת מהמשתמש שאורכה לא יעלה על 10000 תווים. הפלט יהיה קידוד רצף הסיביות שהיה מתקבל מאופציה 1, אך המקום הדפסת הסיביות יודפסו תווי ה-ascii המיוצגים על ידי אותן סיביות.

עבור **lo@** יודפס **x**.

גם כאן ניתן להניח נכונות הקלט.

1. **Decode text using the Huffman Tree**

תקלט מחרוזת מהמשתמש שאורכה לא יעלה על 10000 תווים המורכבת מאחדות ואפסים בלבד שמספרן הכולל הוא כפולה של 8. הפלט יהיה פענוח רצף הסיביות בעזרת עץ ההאפמן שבזיכרון. ניתן להניח שהטקסט המפוענח מסתיים בתו @, יש להתעלם מכל מה שבא אחריו.

עבור **01111000**  לפי העץ שבאיור יביא לפלט**lo@. שימו לב כי להתעלם מן התו 0 האחרון.**

ניתן להניח את חוקיות הקלט.

1. **Bit-Decode using the Huffman Tree**

תקלט מחרוזת מהמשתמש שאורכה לא יעלה על 10000. הפלט יהיה פענוח רצף הסיביות המיוצג על ידי התווים בעזרת העץ שבזיכרון. כמו מקודם ניתן להניח שהטקסט המפוענח מסתיים בתו@ ויש להתעלם מסיביות המופיעות אחריו.

עבור הקלט **x** (תו יחיד במקרה זה) ערכו בascii הוא 120 ועל כן ייצוגו בבסיס 2 הוא **01110000,** לכן הפלט שיתקבל יהיה זהה לפלט שיתקבל מאופציה שלוש עבור הקלט **01110000**.

ניתן להניח את חוקיות הקלט.

1. **Serialize the Huffman Tree**

יודפס תוכנם של כל העלים שבעץ שבזיכרון. כל עלה יודפס בשורה נפרדת. ההדפסה תתבצע במתכונת In-Order: כל **העלים** שלתת העץ השמאלי ואחר כך הדפסת כל העלים של תת העץ הימני. תוכן על עלה יודפס בצורה הבאה: תוכן התו ':' תדירות לדוגמא l:2.

1. **Free the in memory Huffman Tree**

ישוחרר העץ מן הזיכרון יחד עם כל הזיכרון שהוקצה עבור יצירתו. בחירה באופציה זו תביא להדפסת התפריט שבחלק א.

מספר הערות

* מאחר שמטרת התרגיל היא מימוש רשימות מקושרות ועצים אין צורך בבדיקות תקינות לקלט פרט קלט לתפריטים.
* בכל הסעיפים אתם מתבקשים להשתמש בgets על מנת לקרוא מחרוזות, כתוצאה מכך לא ניתן לקלוט טקסטים בני מספר שורות.
* ניתן להניח שכל טקסט שתפענחו יסתיים בתו @.
* עליכןם להשתמש במבנים HuffmanTree ו-LinkedListNode. יש לבנות את העץ בדיוק כפי שהוצג.
* יש לשחרר כל זיכרון שהוקצה בצורה דינמית טרם סיום התוכנית.
* זיכרו, עליכם לייצר בידוק את אותו הפלט כמו בתוכנית שניתן לכם לבדיקה.

**בהצלחה**