תרגיל 6, מבוא לתכנות מערכות, אביב 2020

הגשה **בזוגות או ביחידים** דרך המודל
התרגיל הוא להגשה עד ליום חמישי, 18/6/2020 בשעה 23:55
בתרגיל זה תתרגלו עבודה עם ביטים וכן עבודה עם קבצים (בינארים וטקסט).

DNA

DNA היא מולקולה המכילה את המידע הגנטי בכל האורגניזמים המוכרים וגם ברב הוירוסים. ה-DNA מורכב מרצף ארוך של ארבעה נוקלאוטידים המסומנים באותיות A, C, G ו-T. בד"כ מצוי ה-DNA בסליל כפול שבו זוגות של נוקלאוטידים משלימים מצויים אחד מול השני (ראו ציור): A מול T ו-G מול C. כלל החומר הגנטי של אורגניזם נקרא גנום. גנום האדם, למשל, הוא באורך של 3 מיליארד נוקלאוטידים לעומת גנומים של חיידקים שהוא בד"כ באורך של 2-3 מיליוני נוקלאוטידים.

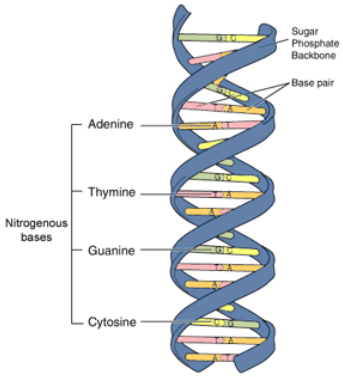

Image adapted from: National Human Genome Research Institute.

חקר ה-DNA של יצורים שונים (תחום שנקרא גנומיקה) הוא אחד מתחומי המחקר החשובים במדעי החיים ומתבסס על ריצוף של DNA מהגנום של אורגנזמים ווירוסים. בתהליך הריצוף נקראת מולקולת ה-DNA והנוקלאוטידים שלה נכתבים כרצף לקובץ על מחשב. קיימים מספר פורמטים לקבצי DNA, הנפוץ שבהם הוא FASTA. בפורמט זה כל רצף מיוצג ע"י שני שדות:

1. שורה המתחילה בתו "<" וכוללת את שם הרצף והתאור שלו. שם הרצף אינו מכיל רווחים ומגיע בצמוד לתו ה-"<". בתרגיל זה הניחו ששם הרצף חייב להופיע ואינו יכול להיות ארוך מ-256 תוים. תאור הרצף הוא אופציונאלי ויכול להכיל כל טקסט שהוא. התאור מתחיל ברווח שמגיע אחרי שם הרצף ונמשך עד סיום השורה.
2. אפס שורות (עבור רצף ריק) או יותר של הרצף עצמו. מספר התוים בכל שורה של הרצף אינו קבוע ויכול להשתנות. הרצף ממשיך גם לאחר תו ירידת שורה עד להתחלת הרצף הבא שיתחיל עם סימן "<" (או הגעה לסוף הקובץ).

לצורך התרגיל הניחו שבקבצים אתם תעבדו יהיה רצף אחד בלבד: כלומר השורה הראשונה תהיה זו שמתחילה ב-"<" ואחריה יופיעו שורות הרצף. הניחו גם שהאותיות שיכולות להופיע בחלק של הרצף הן רק A, C, G או T או a, c, g, t. הרצף מסתיים בסוף הקובץ.

בדוגמא הבאה יש רצף אחד בפורמט fasta:

- שם הרצף הוא `gi|1255213|gb|U50772.1|PCU50772`
- תאור הרצף הוא `Procryptotermes corniceps 16S ribosomal RNA`
- הרצף עצמו מתחיל באותיות `CGCCTGTTT` ומסתיים באותיות `GATAACAGCGTAA`

```
>gi|1255213|gb|U50772.1|PCU50772 Procryptotermes corniceps 16S ribosomal RNA
CGCCTGTTTAACAAAAACATCTCTTCTCGTTTATTGAAAGGATGACCTCCTGCCCAATGTTATTGAAGGG
CCGCGGTATTTTGACCGTGCAAAGGTAGCATAATCATTAGTTCTTTAATTGTGGACTGGTATGAATGGTT
TGACGAGATGCTAGCTGTCTTTGTTTAAGATTTATTGAATTTGTTGTTTGGGTAAAAATGCCCAGATTTT
TTTATGGGACGAGAAGACCCTATAGAGTTTTATATAATGTGTTCTTTTTGGTTTGTTTGGTTTTTATGAA
TTATTTGTATTTTGTTGGGGTGATGGGAAGATAAGTGTAACTCTTCTTTTTTGGTCTCACTTGTTTGTGT
TTTTATGATCCTTTATTATTGATTGTAAGATTAAATTACCTTAGGGATAACAGCGTAA
```

משימות

בתרגיל הזה תכתבו תוכנית שדוחסת קובץ fasta כך שיתפוס פחות מקום, או פותחת קובץ דחוס לקובץ fasta. לתוכנית תהיינה שתי אפשרויות:

- **`--zip`** – מקבלת קובץ fasta לא דחוס (טקסט) ושם של קובץ דחוס, יוצרת את הקובץ הדחוס.
- **`--unzip`** – מקבלת שמות של קובץ דחוס וקובץ לא דחוס, יוצרת קובץ לא דחוס מהדחוס.

למשל, שורת הפקודה הבאה תיצור קובץ דחוס בשם MEL.A1.fasta.compressed מהקובץ MEL.A1.fna:

```
./dna --zip MEL.A1.fasta MEL.A1.fasta.compressed
```

כדי לפתוח את הקובץ הדחוס שיצרנו לקובץ בשם MEL.A1.fasta.uncompressed נריץ

```
./dna --unzip MEL.A1.fasta.compressed MEL.A1.fasta.uncompressed
```

ממשו את הפונקציות הבאות (החתימות של כולן נמצאות בקובץ dna.c המצורף):

toNA: מקבלת תו DNA (A, C, G, T או a, c, g, t) ומחזירה את הקידוד שלו לפי A/a=0, C/c=1, G/g=2, T/t=3. השתמשו בהגדרת Nucleic_acid שהוא enum שהוגדר עבורכם בשביל להחזיר את הערך המתאים. הדפיסו הודעת שגיאה וסיימו את ריצת התוכנית אם התו שהתקבל אינו חוקי.

toChar: מקבלת Nucleic_acid ומחזירה את התו המתאים: 'A' עבור 0, 'C' עבור 1,'G' עבור 2 ו-'T' עבור 3. אם התקבל קוד מחוץ לטווח 0-3 הפונקציה תדפיס הודעת שגיאה ותסיים את ריצת התוכנית.

dna_seq_length: מקבלת שם של קובץ fasta לא דחוס, מחזירה את אורך הרצף. האורך כולל רק את התווים ששייכים לרצף עצמו (A, C, G, T) ללא שורת הכותרת (שמתחילה ב-<) וללא תווי ירידות השורה.
למשל, אורך הרצף kuku הוא 12:

```
>kuku
CGCCT
CCGCG
TT
```

zip: מקבלת שם של קובץ fasta קיים ושם של קובץ דחוס אליו ייכתב הפלט ומבצעת את התהליך הבא:

1. פותחת את קובץ ה-fasta לקריאה כקובץ טקסט
2. פותחת את הקובץ הדחוס לכתיבה כקובץ בינארי
3. קוראת את שורת הכותרת מקובץ הטקסט (השורה שמתחילה ב-< ומסתיימת ב-\n). הניחו שאורך השורה הוא לכל היותר 256 תווים, כולל תו ירידת השורה. ניתן לקרוא את השורה בעזרת getline או בעזרת fgets.
4. שומרת בתחילת הקובץ הבינארי את אורך שורת הכותרת.
5. שומרת את שורת הכותרת עצמה כפי שקראתם אותה. השורה הזאת לא נדחסת.
6. מחשבת את אורך הרצף, לא כולל תווי ירידות שורה. לצורך כך תשתמשו בפונקציה dna_seq_length.
7. שומרת בקובץ הבינארי את אורך הרצף.
8. קוראת את תווי הרצף אחד אחרי השני ודוחסת כל ארבעה נוקלאוטידים ל-unsigned char אחד באופן הבא:
   - כל נוקלאוטיד יישמר ב-2 ביטים על פי הקידוד: A=0, C=1, G=2, T=3. השתמשו בפונקציה toNA בשביל לקבל את הקוד המתאים לתו כלשהו.

- בכל ביית (8 ביטים) שימרו את הנוקלאוטיד הראשון בביטים 0-1, את השני בביטים 2-3, את השלישי בביטים 4-5 ואת הרביעי בביטים 6-7.

למשל: ארבעת הנוקלאוטידים ATGT יישמרו כ-11101100 (ה-MSB בצד שמאל, הרצף מתחיל ב-A ונגמר ב-T). התעלמו מתווי ירידת שורה (\n).

9. כאשר מילאתם ביית כיתבו אותו (בעזרת fwrite) לקובץ הבינארי. במידה ואורך הרצף אינו כפולה של 4 אז הבית האחרון לא יהיה מלא. אין חשיבות לתוכן החלק אותו לא מילאתם.
10. כאשר סיימתם סיגרו את שני הקבצים.

unzip: מקבלת שם של קובץ fasta דחוס קיים ושם של קובץ fasta לא דחוס אליו ייכתב הפלט ומבצעת את התהליך הבא:

1. פותחת את קובץ ה-fasta הדחוס לקריאה כקובץ בינארי
2. פותחת את הקובץ הלא דחוס לכתיבה כקובץ טקסט
3. קוראת את אורך שורת הכותרת
4. קוראת את שורת הכותרת מהקובץ הבינארי. מספר הבתים שצריך לקרוא הוא המספר שקראתם בשלב הקודם. ודאו שבסיום השורה שקראתם יהיה 0\.
5. כותבת את שורת הכותרת לקובץ הלא דחוס.
6. קוראת את אורך הרצף
7. קוראת את הבתים שמכילים את רצף ה-DNA הדחוס, מדפיסה את ארבעת התוים שבכל בית לקובץ הלא דחוס. השתמשו בפונקציה toChar שמימשתם כדי לקבל את התו (ערך ה-ASCII) שמתאים לכל קוד.
8. עבור הרצף, הכניסו תו ירידת שורה כל 80 תווים. כלומר: לאחר שהדפסתם 80 תווים מהרצף, הדפיסו \n.
9. כאשר סיימתם לקרוא את מספר התווים הנדרש סיגרו את שני הקבצים.

תכנית הרצה

במודל תמצאו תבנית של הקבצים אותם אתם צריכים לכתוב: dna.h, dna.c ו-main.c. אתם צריכים להכניס את הקוד שלכם לקבצים אותם תורידו. לאחר מכן הריצו את התוכנית שלכם על הקובץ שנתון במודל (MEL.A1.fasta) וודאו שלאחר דחיסה ופתיחה אתם מקבלים את אותו הקובץ.

דגשים

- יש לבדוק תקינות קלטים לכל הפונקציות, ולהציג הודעות שגיאה בהתאם.
- במקרים של שימוש בזיכרון דינאמי, יש לוודא כי ההקצאות אכן ניתנו ע"י מערכת ההפעלה, וכן יש לנהל בקפידה את הזיכרון ולדאוג שבתום התכנית כל זיכרון דינאמי אכן משוחרר.
- במימוש פונקציות, מומלץ לעבוד עם מצביעים (כתובות) למבנים הנדרשים, הן כארגומנטים והן כערכי חזרה, כפי שנלמד.
- יש לוודא כי התכנית עוברת קומפילציה <u>ללא כל שגיאות או אזהרות כלשהן על שרת החוג</u>, ורצה בהצלחה.

הגשה

- **הכניסו את שמות ותעודות הזהות של הסטודנטים המגישים כהערה בהתחלת main.c.**
- **הגשה אלקטרונית**: יש להגיש במערכת Moodle קובץ tar בשם **`ex4.tar`** או **`ex4.zip`** (אבל לא 7zip או rar!) המכיל את כל קבצי ה-h וה-c של התוכנית.

בהצלחה!