עזריאלי
מכללה אקדמית להנדסה
ירושלים

תרגיל 3 – חישוב מבוזר של פולינומים
מערכות הפעלה, סמסטר ב'
להגשה: 28.5.2023

עדכונים מ 17.5.2023

שאלה 1

תרגיל: מחשבון פולינום

בשאלה זו אתם נדרשים לכתוב תוכנית בשפת C המבצעת פעולות אריתמטיות בסיסיות על פולינומים. התוכנית צריכה לאפשר למשתמש להזין שני פולינומים, לקלוט מהמשתמש פעולה לביצוע: **חיבור, חיסור או כפל** ואז להציג את התוצאה בהתאם.

דרישות- קלט:

להלן דוגמא של הקלט והפלט של התוכנית. התוכנית קולטת מהמשתמש פעולה לחישוב ומדפיסה את הפלט ככה בלולאה עד אשר מקישם את המחרוזת END. כאשר נקלטת מחרוזת END התוכנית יוצאת. בדוגמא שלהלן רואים שלוש איטרציות של התוכנית, השורות האי-זוגיות הם הקלט והזוגיות הם הפלט. (המספרים של השורות הם רק לצורך נוחות ולא יופיעו בתוכנית עצמה)

```
(2:3,2,1)ADD(3:2,4,0,-1)
2x^3 + 7x^2 + 2x
(2:3,2,1)SUB(3:2,4,0,-1)
-2x^3 - 1x^2 + 2x + 2
(2:3,2,1)MUL(3:2,4,0,-1)
6x^5 + 16x^4 + 10x^3 + x^2 - 2x - 1
```

הסבר מפורט:

1. הקלט מורכב מזוגות של פולינומים המוקפים בסוגריים. לכל פולינום שני חלקים: הדרגה והמקדמים. אחרי דרגת הפולינום מופיע לפני הנקודתיים (:). המקדמים אחרי ומופרדים בפסיקים (,). לדוגמה, (2:3,2,1) מייצג פולינום של דרגה 2 עם מקדמים 3, 2 ו-1.

פעולות:

1. פעולת "ADD" מתבצעת חיבור בין שני פולינומים. לדוגמה, `(2:3,2,1)ADD(3:2,4,0,-1)` מייצג את התוספת של הפולינום $2x^3 + 4x^2 - 1$ והפולינום $3x^2 + 2x^1 + 1$

2. פעולת "SUB" מבצעת חיסור של הפולינומים המוגדרים בסוגרים.

3. פעולת "MUL" כנ״ל מכפלה של שני הפולינומים המוגדרים בסוגרים.

פלטים:
הפלטים מייצגים את התוצאה של הפעולות המתאימות:

1. תוצאת פעולת "ADD" : הפלט מציג בשורה 2, את סכום שני הפולינומים בפורמט פולינום סטנדרטי.
2. תוצאת חיסור : הפלט בשורה 4 מייצג חיסור הפולינומים משורה 3
3. תוצאת הכפל : הפלט בשורה 6 מייצג את תוצאת כפל הפולינומים מהשורה 5

הערה : הקפד לעקוב אחר פורמט הקלט והפלט במדויק, כולל שימוש בנקודתיים (:) להפרדת התואר והמקדמים, ופסיקים (,) להפרדת המקדמים של כל פולינום.

הטמיעו פונקציות נפרדות לחיבור, חיסור וכפל, שלוקחות את מערכי המקדמים של שני הפולינומים כקלט ומחזירות את הפולינום שנוצר. (קוד מדולרי, ראינו דוגמא בשיעור לפונקציה מחשבון עם מספים – בנו על פי זה מחשבון לפולינומים.

אפשר להניח שהקלט תקין

לא ניתן להניח דבר על גודל הפולינום. אורך המחרוזת המקסימלי שניקלט מהמשתמש הוא עד 128 תווים.

שמות קבצים.
1. Ex3q1.c

עזריאלי
מכללה אקדמית להנדסה
ירושלים

שאלה 2

בשאלה זו נרחיב את השאלה לחישוב בעזרת שתי תוכנית יצרן-צרכן בשיטת ה bounded buffer.

פורמט הקלט והפלט יהיה זהה לזה של שאלה 1, אך הפעם יהיו שני תוכניות אחת תקרא את הקלט (בפורמט של שאלה 1) ותכתוב אותו לזיכרון המשותף. גודל זיכרון משותף הינו -- 1280 עד 10 פעולות חישוב על פולינומים, hותוכנית שניה (צרכן) תקרא את הנתונים ותחשב את פעולות הפלינומים.

התוכנית בשאלה זו, תשתמש בפרדיגמת יצרן-צרכן בשיטת ה bounded buffer.
-- באמצעות זיכרון משותף. תהליך היצרן יפיק פולינומים ויכתוב אותם לזיכרון משותף, בעוד שתהליך הצרכן יקרא את הפולינומים מהזיכרון המשותף ויחשב את ערכיהם.

דרישות:

1. התוכנית צריכה ליצור זיכרון משותף לאחסון הפולינומים. ניתן ליישם את הזיכרון המשותף באמצעות הפונקציות `shmat`, `shmget` ו.`shmdt` או כל דרך אחרת
2. תהליך היצרן צריך ליצור פולינומים שנקראו מהמשתמש.
3. תהליך היצרן צריך לכתוב את הפולינומים לזיכרון המשותף באופן מסונכרן ובכל שיטה שתבחרו – השתמשו בסמפורים או במנעולי mutex לסנכרון.
4. תהליך הצרכן צריך לקרוא את הפולינומים מהזיכרון המשותף ולחשב את ערכם.
5. התוכנית צריכה לטפל במספר פולינומים בזיכרון המשותף. ובגודל משתנה.
6. היצרן יצא וימחק את הזיכרון המשותף עת המשתמש הקיש END, עד אז יישאר בלולאה אין סופית אשר קוראת המשתמש.

רמזים:

1. השתמש במבנים כדי לייצג את הפולינומים והמטא-נתונים שלהם, כולל התואר והמקדמים.
2. יש להשתמש בסמפורים או במנעולי mutex כדי לסנכרן את הגישה לזיכרון המשותף בין תהליכי היצרן והצרכן.

a. על שימוש בסימפורים בין תהליכים ניתן לקרוא כאן (דומה למנגנון של זיכרון משותף), ניתן להשתמש גם בדרכים אחרות. אבל זכרו – בן תהליכים לא threads
https://www.prodevelopertutorial.com/system-v-semaphores-in-c-using-semget-semctl-semop-system-v-system-calls-in-linux/

3. השתמש במנגנוני סנכרון מתאימים כדי למנוע race-conditions

שמות קבצים.

1. Ex3q3a.c
2. Ex3q3a.c

שאלה 3

בשאלה זו נרחיב את שאלה 2, לחישוב בעזרת שתי תוכנית יצרן-צרכן. כאשר הפעם החישוב של **פעולת החיבור והחיסור בלבד ! יבוצעו** באופן מבוזר ומקבילי.

קצת הסבר, על איך אפשר לעשות את זה....

- הגדר את המבנה הפולינומי- צור מבנה שייצג את הפולינום. המבנה צריך להכיל את מידת הפולינום ומערך לאחסון המקדמים.
- צור Threads מרובים, כאשר כל Threads אחראי לחישוב מקדם אחד. מספר הThreads צריך להיות שווה לדרגת הפולינומים שכן כל דרגה מתאימה למקדם אחד.
- פונקציית Threads - הגדר פונקציה שכל Threads יבצע. פונקציה זו תיקח את הפולינומים ואינדקס של המקדם לחישוב כפרמטרים. בתוך הפונקציה, בצע חיבור או חיסור של מקדמים ואחסן את התוצאה.
- ביצוע Threads - בפונקציה הראשית, צור את הפולינומים ואתחול המקדמים (אותם נקרא מהזיכרון המשותף). לאחר מכן, צור את הThreads והקצה את הנתונים המתאימים (פולינומים, פעולה לביצוע ואינדקס מקדם) לכל Threads. לבסוף, המתן עד שכל הThreads יסיימו את ביצועם.

זכור לטפל בסנכרון Threads, כגון שימוש ב mutexes-או מנגנוני סנכרון אחרים, כדי להבטיח בטיחות Threads בעת גישה למשאבים משותפים.

על ידי ביצוע שלבים אלה, אתה יכול לכתוב תוכנית C שמוסיפה או מפחיתה שני פולינומים במקביל, כאשר כל Threads אחראי לחישוב מקדם אחד. שים לב שכפל נשאר בפורמט הישן ללא חישוב בעזרת Threads

שמות קבצים.

3. Ex3q3a.c
4. Ex3q3a.c

סינטקס – הקפידו על סינטקס הזחות שמות משתנים משמעותיים

"במקרה הגרוע" אפשר להשתמש באתר הזה לסדר את ההזחות בקוד

https://codebeautify.org/cpp-formatter-beautifier

להגיש עם Readme

להגיש עם Makefile

עבודה עצמאית, מצאתי פתרון ברשת או במקום אחר או אצל סטודנט אחר או שנה אחרת – זה לא עבודה עצמאית, וגם לא משהו דומה...אנחנו גם נבדוק את זה וגם נבחן אותכם על זה בסוף השנה.
(**בבקשה, בבקשה, בבקשה!** זה רק בשבילכם, לא תעבדו עצמאית לא תלמדו...אין קיצורי דרך)
זיכרו לבדוק דליפות זיכרון עם valgrind