

## תרגיל מס' 3 – פונקציות (מאיר)

### הוראות הגשה

- מועד אחרון להגשה: 5/12/2006
- יש לשלוח את הקבצים באמצעות פקודת submitex לפני חלוף התאריך האחרון להגשה. שם ההגשה: func3
- יש להקפיד מאוד על כל הוראות עיצוב הקלט והפלט, כמפורט בכל סעיף וסעיף. על הפלט להיראות בדיוק כמו בדוגמאות. אין להוסיף או להשמיט רווחים או תווים אחרים ואין להחליף אותיות גדולות בקטנות או להיפך ⊗ אי-הקפדה על פרטים אלה עלול לגרום ירידה משמעותית ביותר בציון התרגיל עד כדי 0.0. ראו הוזהרתם!
- להזכירכם, העבודה היא אישית. "עבודה משותפת" דינה כהעסקה.
- יש להדפיס למסך פלטי תוכנית בלבד.
- בכל מקום בו אתם נדרשים לתת הודעה, אין להדפיסה במרכאות כפולות.
- עבור כל בדיקה שהנכם מבצעים שלא ציינו במפורש את הודעת השגיאה, יש להדפיס "error".
- בכל הקלטים בתכנית הנכם יכולים להניח שלא ייקלט enter בלבד ללא קלט אמיתי.
- במסך המשתמש אתם תראו גם את הקלטים וגם את הפלטים אך אל דאגה כאשר אנו נריץ נראה רק את הפלטים. לכן וודאו כי הפלט מופיע במסך בפורמט הנדרש.
- אין להשתמש בפונקציות מספריות מלבד הספריות math.h ו-stdio.h (ע"מ לקמפל תוכנית שעושה שימוש ב-math.h יש להשתמש בפקודה הבאה: gcc -lm main.c)
- אין להשתמש בחומר שעדיין לא נלמד בכיתה לפיתרון התרגיל, כגון מערכים.

### התכניות

כתוב תכנית המשמשת כמחשבון עבור שברים פשוטים (שבר פשוט הוא שבר הקטן ממש מ-1). התוכנית תקלוט בכל פעם רצף של תווים המהווה ביטוי חשבוני המכיל שברים פשוטים, כגון:

$$1/5 + 4/9 * 2/4 =$$

### חוקיות ביטויים

- ראשית, התוכנית תוודא שמדובר בביטוי חשבוני חוקי. המקרים המפורטים להלן אינם חוקיים:
- ביטוי המכיל מספר לא מתאים של אופרטורים (סימני החישוב) ואופרנדים (עליהם מבצעים את החישוב) או שהסדר בו ניתנו אינו נכון. לדוגמא:  $= - 1/3$  הוא אינו חוקי. בסיום כל ביטוי חייב להופיע =, אחרת הביטוי אינו חוקי.
  - אופרטור (מפעיל) לא חוקי. האופרטורים החוקיים יפורטו בהמשך, כל אופרטור שלא פורט אינו חוקי.
  - אופרנד (מופעל) לא חוקי. הניחו כי אופרנד יהיה תמיד מהצורה integer/integer, אך חובה עליו להיות **שבר פשוט** (או מספר שלם במקרה של חזקת שלם בלבד – יפורט בהמשך). שבר שהמכנה שלו הוא 0 אינו שבר חוקי. הניחו כי לא יינתנו שברים שליליים כלומר לא יהיה שבר  $-3/4$ .

במידה והביטוי אינו חוקי, יודפס למסך "input error" בשורה נפרדת, והתוכנית תמתין לקלט חילופי. שימו לב שבמידה וגיליתם שהביטוי אינו חוקי עליכם לסיים לקלוט את כולו (עד ה enter) ע"מ לאפשר קליטת ביטוי חדש.

### חישוב תוצאות ביטויים

לאחר שהתוכנית ווידאה שמדובר בקלט חוקי, יש להדפיס למסך את תוצאת החישוב כשבר גם כן, לפי אותו פורמט של integer/integer.

### אופרטורים

האופרטורים החוקיים הם האופרטורים הבאים:

אופרטור	פעולה	צורת חישוב
+	חיבור	$a/b + c/d = (a*d + b*c) / (b*d)$
-	חיסור	$a/b - c/d = (a*d - b*c) / (b*d)$
*	כפל	$a/b * c/d = (a*c) / (b*d)$
/	חילוק	$a/b / c/d = (a*d) / (b*c)$
^	חזקה (הניחו שהמעריך בקלט שלם)	$a/b ^ c \Rightarrow (a/b)^c = a^c / b^c$
=	סיום חישוב הביטוי	

### סדר הפעולות

במחשבון שלכם לא מוגדר סדר קדימות של מפעילים (operator precedence). לכן, המופעל השמאלי בכל פעולה יהיה הערך שחושב עד כה, ואילו המופעל הימני יהיה הערך המופיע מייד אחרי המפעיל הנוכחי. לדוגמה (הדוגמא כוללת מספרים שלמים אבל הרעיון זהה):

$$3 + 5 * 2 = 8 * 2 = 16$$

### חישוב הביטוי - חלוקה לפונקציות

על-מנת לקלוט ולחשב ביטוי, יש לקרוא לפונקציה CalculateExpression.

```
void CalculateExpression ()
```

### קליטת אופרנד ראשון

ראשית, CalculateExpression קולטת את האופרנד הראשון (מונה ומכנה) ומוודאת את חוקיותו.

### קליטת אופרטור

במידה והאופרנד הראשון חוקי, תקרא CalculateExpression לפונקציה InputOperator הקולטת אופרטור ומוודאת את חוקיותו. הערך המוחזר ע"י פונקציה זו הוא ערך המוגדר ע"י enum ומציין את פעולת האופרטור החוקי שנקלט. הגדירו תחילה את הטיפוס enum Operator, שערכיו יהיו שמות משמעותיים עבור המופעלים, המוגדרים לעיל. תנו את המזהה Operator לטיפוס זה (באמצעות typedef). בתור ערך ראשון מהטיפוס Operator, הגדירו את הערך illegal. ערך זה יוחזר מן הפונקציה במקרה של אי-חוקיות.

```
Operator InputOperand ()
```

### קליטת אופרנד נוסף

במידה והאופרטור חוקי, CalculateExpression תקלוט ותוודא את חוקיות האופרנד הנוסף, תבצע את החישוב וחוזר חלילה עד לסיום הביטוי (קליטת =).

**ביטוי חוקי**

אם הביטוי כולו חוקי, CalculateExpression מפעילה את פונקציה האופרטור המתאימה. לכל אופרטור יש לכתוב שתי פונקציות מתאימות נפרדות המבצעות את אותו חישוב – האחת מחזירה את המונה והשניה מחזירה את המכנה של התוצאה (מאחר ופונקציה יכולה להחזיר ערך בודד בלבד). לדוגמא, עבור + יש ליצור את הפונקציות:

```
int AdditionNominator(int,int,int,int)
```

```
int AdditionDenominator(int,int,int,int)
```

בכל פעם שיש צורך להפעיל אופרטור בין שני שברים, יש לקרוא לשתי הפונקציות וכך לקבל כתוצאה שבר חדש (יתכן, וסביר מאוד, ששבר זה לא יהיה פשוט, אך אין בכך בעיה).

**צמצום שברים**

מאחר וישנה סבירות גבוהה שהשבר החדש אינו מוחזק בצורתו המינימלית, עליכם לצמצמו ככל האפשר. על מנת לעשות זאת, קראו לפונקציות צימצום (אחת עבור המונה ואחת עבור המכנה) המנסות לחלק את המונה והמכנה לכל המספרים הראשוניים הקטנים ממאה ומשני המספרים המהווים את המונה והמכנה ומחזירות את המונה והמכנה לאחר הצמצום המקסימלי האפשרי. שימו לב כי  $0/x$  יצומצם תמיד ל-  $0/1$ .

```
int ReduceNominator (int,int)
```

```
int ReduceDenominator (int,int)
```

**סיום הביטוי**

עם קליטת האופרטור =, CalculateExpression תדפיס את ערך הביטוי למסך כשבר פשוט.

**ביטוי לא חוקי**

אם באחד השלבים של הקליטה הסתבר כי הביטוי אינו חוקי, CalculateExpression תדפיס את הודעת השגיאה למסך. מאחר ויש לדלג על כל הקלט הנוטר עד התו newline (enter), יש לממש את הפונקציה skip\_to\_newline, שתיקרא ע"י CalculateExpression במקרה זה לפני ההמתנה לקלט חילופי.

```
void SkipToNewline();
```

**התחלת וסיום התוכנית**

לפני כל קליטת ביטוי תודפס למסך השאלה:

```
calculate expression? [y/n]
```

אם המשתמש הגיב ע"י הקשת 'y' **בלבד**, התוכנית תערך לקליטת ביטוי ותקרא לפונקציה CalculateExpression.

אם המשתמש הגיב ע"י הקשת 'n' **בלבד**, התוכנית תסתיים. בתגובה להקשת כל ערך אחר, תופיע השאלה שוב.

**הנחות:**

1. כל המספרים שיקלטו הם שלמים.
2. התוכנית תצפה לקלוט את כל הביטוי בשורה אחת ו- ENTER בסופה.
3. לא יתכן מצב שאחרי = יהיה משהו אחר מלבד ENTER (אך יתכן מצב בו יחסר =) .
4. שימו לב לרווחים לפני כל אופרנד ואופרטור. הניחו כי הרווחים ינתנו תמיד לפי הדוגמא.
5. הניחו כי לא יקלט = בלבד, ללא ביטוי או רק עם רווח לפניו.
6. הניחו כי לא ייקלטו שברים שליליים.
7. הניחו שהמעריך הוא שלם בפונקציית החזקה.

**בהצלחה!**

This document was created with Win2PDF available at <http://www.daneprairie.com>.  
The unregistered version of Win2PDF is for evaluation or non-commercial use only.