ביה"ס למדעי המחשב ומתמטיקה 29.3.2017
המכללה האקדמית נתניה

מבחן מועד ב'
יסודות מערכות פתוחות
סמסטר חורף, תשע"ז

- משך המבחן: שלוש וחצי שעות.
- יש לענות על כל השאלות.
- מותר השימוש בחומר עזר כלשהו, פרט למחשבים, (מחשבונים מותר).
- יש להקפיד על כתיבה ברורה ומסודרת של התשובות.

1. (15 נקודות)

חלק א (5 נקודות)

נניח שנתון קובץ בשם P1.1 שתכנו הוא:

```
echo -n "Enter string: "
read str
echo -n "Enter num1: "
read num1
echo -n "Enter num2: "
read num2
w1=$(echo $str | cut -c1-$[num1-1])
w2=$(echo $str | cut -c$num1-$num2)
w3=$(echo $str | cut -c$[num2+1]-)
echo The result is ${w3}${w2}${w1}
```

ונניח שנתון קובץ בשם P1 שתכנו הוא:

```
#!/usr/bin/expect
set x [lindex $argv 0]
set y [lindex $argv 1]
set z [lindex $argv 2]
log_user 0
spawn P1.1
log_user 1
expect "string" {send "$x\n"}
expect "num1" {send "$y\n"}
expect "num2" {send "$z\n"}
expect "The" {exit}
```

מה יתקבל בפלט (דהינו יוצג על המסך) לאחר הפעלת הפקודה:

```
P1 abcdefghijk 4 7
```

חלק ב (10 נקודות)

מה יתקבל בפלט (דהינו יוצג על המסך) לאחר הפעלת קטע הקוד הבא:

```
echo "output 1:"
x="(1a2b3c4d5)*2"
echo $[$(echo $x|tr a-f "+")]

echo "output 2:"
echo "ab@%c5d&x34z" | tr -d "a-z" | tr -c "0-9\n" "*"

echo "output 3:"
echo "  abhi1z1 ab cd" >|F1
echo "  aa hiz abxza ab" >> F1
echo "  az1 1hi ab1 " >> F1
echo "  a2hz xi ab" >>F1
egrep "[ ]+[ab]([^ ab]+)[ ]+" F1

echo "output 4:"
echo 12 >| F2
echo 3 >> F2
echo 10 >> F2
awk '{s+=$1;getline;print s}' F2

echo "output 5:"
echo "ab::cd:ef"|awk 'BEGIN {FS=":"} {print $3,NF}'
```

2. (10 נקודות)

נגדיר שמילה היא רצף של תווים ללא תווי רווח וסוף שורה.

כתוב תכנית Script ב- sed בשם P2 שמקבלת כפרמטר שם קובץ ומדפיסה את כל השורות שמכילות 3 או יותר תווים רצופים זהים. בנוסף התכנית מדפיסה את מספר השורות של הקובץ שלא הופיעו בפלט.

על המבנה של התכנית P2 להיות כדלהלן:

התכנית מכילה שורה אחת או יותר של פקודות sed לדוגמה:

```
sed s/dog/cat/ $1
```

אין להשתמש בפקודות שאינן של sed ושאינן מתחילות ב- sed.

מותר לכתוב לקובץ ביניים כמו למשל:

```
sed s/dog/cat/ $1 >| tmp
```

מותר גם לקרוא מקובץ ביניים כמו למשל:

```
sed s/dog/cat/ < tmp
```

אסור להשתמש ב- pileline זאת אומרת המבנה הבא אסור:

```
sed s/dog/cat $1 | sed s/abc/def
```

לדוגמה, נניח שתוכן הקובץ F2 הוא:

```
abc2 1abababab
dea 13333
12112112
abc  cd aabbcc
adddb
aa bb ccccccc11
hhh
```

לאחר הפעלת התכנית ע"י הפקודה:

```
P2 F2
```

יתקבל הפלט:

```
dea 13333
adddb
aa bb ccccccc11
hhh
3
```

3. (25 נקודות)

בחברה להשכרת רכבים שומרים נתונים על השכרות בקובץ ששמו rents שכל שורה שלו היא במבנה הבא:

תאריך החזרה : תאריך השכרה : דגם הרכב : שם השוכר : מחיר ליום

עבור רכב שהושכר ולא הוחזר מופיע – (מינוס) בשדה של תאריך ההחזרה.

לדוגמה השורה הבאה:

```
120:Yossi Levi:Toyota 2008:1/1/2016:22/1/2016
```

מצינת שרכב מדגם טויוטה 2008 הושכר על ידי יוסי לוי החל מתאריך 1/1/2016 ועד תאריך 22/1/2016 (כולל), במחיר של 120 ש"ח ליום.

חלק א (10 נקודות)

כתוב פונקציה ב- awk בשם is_date_less_than(date1,date2)

שמקבלת כפרמטרים שני תאריכים בפורמט של dd/mm/yyyy ומחזירה YES אם התאריך הראשון קטן מהתאריך השני, אחרת הפונקציה מחזירה NO.

אין להשתמש בפקודות system בפונקציה זו.

לדוגמה הקריאה:

```
is_date_less_than(1/1/2016,1/12/2016)
```

מחזירה YES.

הקריאה:

```
is_date_less_than(1/1/2016,02/1/2016)
```

מחזירה YES.

הקריאה:

```
is_date_less_than(1/1/2016,01/1/2016)
```

מחזירה NO.

ניתן להניח שיום מתואר על ידי 1 או 2 ספרות, חודש מתואר על ידי 1 או 2 ספרות ושנה מתוארת על ידי 4 ספרות.

חלק ב (15 נקודות)

כתוב/כתבי תכנית ב- awk בשם P3 שמקבלת כפרמטרים שם שוכר ושני תאריכים,ומדפיסה לפלט בשורה הראשונה את שם השוכר והתאריכים המבוקשים, ובשורות שלאחר מכן את כל השורות שמתארות השכרות של השוכר שנמצאות בתחום שבין שני התאריכים שהועברו כפרמטרים לתכנית. (תיתכן חפיפה חלקית לתחום כפי שמתואר בדוגמה שבהמשך). בסוף כל שורה יופיע תו רווח ולאחריו מספר שמציין את המחיר הכולל של ההשכרה בתחום שמתקבל מהחפיפה בין התחום שמתואר על ידי השורה לבין התחום שמתואר על ידי התאריכים המבוקשים (כפי שמתואר בדוגמה שבהמשך).

יש להניח שהתאריך השני שמועבר לתכנית קטן או שווה לתאריך שבו מופעלת התכנית.

אין להשתמש בפקודות system בתכנית P3.

לדוגמה אם השוכר yossi levi שכר את הרכב Toyota 2008 החל מתאריך 1/1/2016 ועד תאריך 20/1/2016 (כולל) במחיר של 120 ש"ח ליום, ואם התאריכים המבוקשים הם מתאריך 11/1/2016 ועד תאריך 22/1/2016, אז בסוף השורה שמתארת את ההשכרה הזו יופיע המספר 1200 כי החפיפה בין התחומים מכילה בדיוק 10 ימים, דהינו החל מ- 11/1/1016 ועד 20/1/2016.

לצורך הפתרון של חלק זה של השאלה ניתן להניח שקיימות 2 הפונקציות הבאות (אין צורך לכתוב את הפרוט של פונקציות אלה):

הפונקציה: is_date_less_than(date1,date2) שתוארה בחלק א

הפונקציה: number_of_days_in(date1,date2) שמחזירה מספר שמתאר את מספר הימים שנמצאים בין התאריכים date1 ו- date2 כולל date1 ו- date2

לדוגמה הקריאה:

number_of_days_in(01/01/2016,20/01/2016)

מחזירה 20.

הקריאה:

number_of_days_in(01/01/2016,01/01/2016)

מחזירה 1.

לדוגמה, נניח שתוכן הקובץ rents הוא:

```
120:Yossi Levi:Toyota 2008:1/1/2016:20/1/2016
100:Dan Cohen:Nissan 1998:2/01/2015:03/01/2015
400:Yossi Levi:Toyota 2015:11/1/2015:12/1/2016
200:Yossi Levi:Toyota 2010:1/1/2015:20/1/2015
300:Yossi Levi:Toyota 2016:18/1/2016:-
```

לאחר הפעלת התכנית על ידי הפקודה:

```
P3 "Yossi Levi" 11/1/2016 22/1/2016
```

מתקבל הפלט:

```
yossi levi 11/1/2016 22/1/2016

120:Yossi Levi:Toyota 2008:1/1/2016:20/1/2016 1200
400:Yossi Levi:Toyota 2015:11/1/2015:12/1/2016 800
300:Yossi Levi:Toyota 2016:18/1/2016:- 1500
```

שימו לב שהשורה:

```
200:Yossi Levi:Toyota 2010:1/1/2015:20/1/2015
```

אינה מופיעה בפלט כי אין חפיפה בין התחום המבוקש של התאריכים ל לתחום התאריכים של שורה זו.

שימו לב שבגלל שבשורה:

```
300:Yossi Levi:Toyota 2016:18/1/2016:- 1500
```

יש מינוס "-" בתאריך ההחזרה, ההנחה היא שעד התאריך 22/1/2016 לא הסתימה תקופת ההשכרה. ולכן החישוב של המחיר הכולל (דהינו 1500) עבור השורה הנ"ל התבצע עבור 5 ימים החל מ- 18/1/2016 ועד 22/1/2016.

4. (25 נקודות)

הגדרה: עבור מטריצה A (לא בהכרח ריבועית) נגדיר את הקילוף ה – i,j של המטריצה, כמטריצה שמתקבלת מהמטריצה A לאחר הוצאת שורה i ועמודה j.

לדוגמה, עבור המטריצה A הבאה:

```
10 20 30 40 50
1 2 3 4 5 6 7
60 70 80
8 9
```

הקילוף ה 2,3 של המטריצה A הוא המטריצה הבאה:

```
10 20 40 50
60 70
8 9
```

חלק א (10 נקודות)

כתוב/כתבי תכנית ב- bash בשם P4.1 (אין להשתמש בפקודות awk ו- sed בשאלה זו) שמקבלת כפרמטרים שם קובץ שמכיל מטריצה של מספרים, ושני מספרים i ו- j ומדפיסה לפלט את המטריצה שמתקבלת לאחר הקילוף ה- i,j של המטריצה.

לדוגמה, נניח שהמטריצה A היא כפי שתואר לעיל.

לאחר הפעלת התכנית על ידי הפקודה:

```
P4.1 A 1 2
```

יתקבל הפלט:

```
1 3 4 5 6 7
60 80
8
```

חלק ב (15 נקודות)

כתוב/כתבי תכנית ב- bash בשם P4.2 (אין להשתמש בפקודות awk ו- sed בשאלה זו) שמקבלת כפרמטר שם קובץ שמכיל מטריצה של מספרים (בהמשך נקרא לה מטריצה A) ובודקת האם קימים 2 מספרים i,j כך שהקילוף ה- i,j של A מכיל 2 עמודות שסכום המספרים שלהן שווה. במידה והתשובה היא כן התכנית מדפיסה את שני המספרים (עם רווח אחד בינהם), במידה ולא התכנית מדפיסה NO.

בחלק זה ניתן להשתמש בתכנית P4.1 שהוגדרה בחלק א. (ניתן להשתמש בתוכנית P4.1 גם אם לא עניתם על חלק א).

לדוגמה, נניח שנתונה מטריצה B הבאה:

```
10 20 40 50
60 70 20 28
8 9 50 40
```

לאחר הפעלת התכנית על ידי הפקודה:

```
P4.2 B
```

יתקבל הפלט:

```
1 2
```

כי לאחר הקילוף 1,2 של המטריצה B מתקבלת המטריצה הבאה:

```
60 20 28
8 50 40
```

ובמטריצה זו יש 2 עמודות שהסכום שלהן שווה, דהינו עמודות 1 ו- 3.

דוגמה נוספת, נניח שנתונה המטריצה C הבאה:

```
1 2 3
4 5 6
7 8 9
```

לאחר הפעלת התכנית על ידי הפקודה:

```
P4.2 C
```

יתקבל הפלט:

```
NO
```

כי בכל קילוף i,j של המטריצה C אין שתי עמודות שסכומן שווה.

5. (25 נקודות)

הגדרה: נגדיר ששם קובץ סופי הוא טוב עבור תיקיה d, אם בתיקיה d ובכל אחת מתתי התיקיות שלה (בעומק כלשהו) קיים קובץ בשם זה (כשמו הסופי של הקובץ).

כתוב/כתבי תכנית ב- Bash (דהינו קובץ Script) בשם P5 שמקבלת כפרמטר שם תיקיה ומדפיסה לפלט את כל השמות הסופיים הטובים עבור תיקיה זו. כל שם קובץ סופי טוב יופיע בשורה נפרדת ועל סדר השמות בפלט להיות לפי סדר לכסיקוגרפי עולה.

לדוגמה, להלן מבנה תיקיה d1 שנמצאת בתיקיה הנוכחית בה מריצים את התכנית P5. המבנה מתואר כפי שמתקבל על ידי הפעלת הפקודה tree על תיקיה זו.

```
d1
|-- A
|-- B
|-- C
|-- D
|-- E
|-- d4
|   |-- A
|   |-- B
|   |-- C
|   |-- D
|   |-- d5
|   |   |-- A
|   |   |-- B
|   |   |-- D
|   |   |-- E
|   |   `-- F
|   |-- f1
|   `-- f2
`-- e
```

לאחר הפעלת התכנית ע"י הפקודה:

```
P5 d1
```

יתקבל הפלט:

```
A
B
D
```

בהצלחה !