גירסה 1.00 - 4.5.2003

ביטויים רגולריים

מסמך זה הורד מהאתר http://underwar.livedns.co.il.
אין להפיץ מסמך זה במדיה כלשהי, ללא אישור מפורש מאת המחבר.
מחבר המסמך איננו אחראי לכל נזק, ישיר או עקיף, שיגרם עקב השימוש במידע המופיע במסמך, וכן לנכונות התוכן של הנושאים המופיעים במסמך. עם זאת, המחבר עשה את מירב המאמצים כדי לספק את המידע המדויק והמלא ביותר.

כל הזכויות שמורות ל**ניר אדר**

Nir Adar
Email: underwar@hotmail.com
Home Page: http://underwar.livedns.co.il

אנא שלחו תיקונים והערות אל המחבר.

ביטויים רגולריים

ביטויים רגולריים (Regular Expressions) הם דרך נוחה כדי לתאר תבניות מורכבות בתוך טקסט.
אנחנו יכולים להיעזר בהם כדי לאתר תבניות בתוך טקסט. לאחר שאיתרנו את התבניות אנו מסוגלים להחליפן באחרות, ולבצע מניפולציות על הטקסט.
ביטויים רגולריים הם כלי רב עוצמה המאפשר לפתור בעיות רבות הקשורות לניתוח מחרוזות במהירות.
למרות שהשימוש בהם יוצר לעיתים קוד שירוץ לאט יותר מקוד שתוכנן לבצע פעולה כלשהי על מחרוזות, נעדיף פעמים רבות להשתמש בביטויים רגולריים – התוכנית שנכתוב תצא קצרה יותר, ברורה יותר, וניתנת לשינוי ביתר קלות.
אלגוריתם שנכתב כדי לנתח מחרוזת יכול להיות אלגוריתם מסובך הדורש הוכחות ובדיקות רבות. במידה והדרישה תשתנה מעט, יהיה צורך להחליפו באלגוריתם חדש. לעומתו ביטוי רגולרי ניתן לכתיבה ושינוי במהירות, ולאחר שצוברים מעט ניסיון קל לראות האם הוא מבצע את הנדרש ממנו.

כאשר נדגים ביטויים רגולריים, וכאשר נשתמש בפונקציות הקשורות לביטויים רגולריים, נשתמש במונחים **טקסט** (text) **ותבנית** (pattern).
טקסט זהו הטקסט בו אנחנו מחפשים מחרוזת או מחרוזות כלשהן. זאת המחרוזת אותה אנו רוצים לנתח.
בדרך כלל נקבל את הטקסט מהמשתמש בתוכנית שלנו.
התבנית היא ביטוי המציין את התבנית של תת המחרוזת אותה אנו מחפשים בטקסט.
כאשר אנחנו מוצאים תת מחרוזת המתאימה לתבנית בתוך הטקסט, אנחנו אומרים כי מצאנו **התאמה** (match).

תחביר בסיסי

התבנית הפשוטה ביותר שתשמש כביטוי רגולרי היא אות, או רצף של אותיות.
כל מופע של האות/האותיות ייחשב כהתאמה.
למשל, עבור התבנית `"a"`, סומנו ההתאמות שימצאו בטקסט הבא:

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.

נשים לב שאותיות קטנות אינן שוות לאותיות גדולות. עבור התבנית "a" לא מצאנו את האות הגדולה A שבתחילת המילה And.

עבור התבנית `"Mary"` ימצאו בטקסט ההתאמות הבאות:

Mary had a little lamb.
And everywhere that **Mary**
went, the lamb was sure
to go.

הסימן ^ מסמל "מתחיל ב" – לדוגמא, התבנית `"^The"` תמצא התאמה בכל מחרוזת המתחילה במילה The.
הסימן $ מסמל "מסתיים ב" – לדוגמא, התבנית `"final$"` תמצא התאמה בכל מחרוזת שתסתיים במילה final.
ניתן לשלב בין סימנים אלו. התבנית `"^abc$"` תמצא כל מחרוזת המתחילה ומסתיימת ב-abc. זוהי יכולה להיות רק המחרוזת "abc" עצמה.

תווי בריחה: נחפש את התבנית `".*"` במחרוזת הבאה:

Special characters must be escaped.*

ההתאמה שנקבל תהיה כל המחרוזת. הסיבה: . ו-* הם תווים מיוחדים, בדומה ל-^ ול-$. כאשר אנו רוצים לחפש תווים אלו בטקסט, עלינו להשתמש בתו בריחה מיוחד שהוא \ כדי לומר שאנו מתכוונים לתווים אלו כתווים, ולא כאל סימנים מיוחדים שיש לפרשם.

נחפש כעת באותה מחרוזת את התבנית `"\.\*"` ונראה מה תהיה ההתאמה:

Special characters must be escaped.*

רק שני התווים בסוף המחרוזת מהווים הפעם את ההתאמה.

התו . (נקודה) משמש כ-wildcard בתבניות. קוראים המכירים את מערכת ההפעלה DOS יכולים לזהות אותו עם התו ? ב-DOS. משמעותו – "כל תו".
למשל, עבור התבנית `".a"` נסמן את ההתאמות בטקסט הבא:

Mary **ha**d **a** little **la**mb.
And everywhere t**ha**t **Ma**ry
went, the **la**mb **wa**s sure
to go.

כאשר אנו מסתכלים על ביטוי רגולרי, כל אות בו מכונה אטום. נראה בהמשך אופרטורים הפועלים על אטום. כדי להגדיר אטום בן יותר מאות אחת, נשתמש בסוגריים מעוגלות.
לדוגמא עבור `"(Mary) ( )(had)"` נקבל את ההתאמה הבאה:

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.

ניתן להגדיר מחלקות תווים. במקום לדרוש שאות מסוימת אחת תופיע, נוכל לדרוש שאות אחת מבין קבוצה שנגדיר היא זו שתופיע. למשל, התבנית `"[abc]"` תתאים לכל מחרוזת הכוללת אחת מהאותיות הקטנות a, b או c. ניתן להגדיר טווח של אותיות, על ידי כתיבת האות הראשונה, לאחריה מקף, ואז את האות האחרונה בטווח. התבנית `"[A-Z]"` תתאים לכל אות גדולה באלף בית האנגלי. דוגמא: עבור התבנית `"[a-z]a"` נקבל את ההתאמה:

Mary **ha**d a little **la**mb.
And everywhere t**ha**t Mary
went, the **la**mb **wa**s sure
to go.

הסימן ^ מציין לרוב התחלה של מחרוזת, אולם כאשר הוא מופיע בתוך מחלקת תווים הוא מקבל משמעות אחרת – שלילה. הכוונה של ^ בראש מחלקת תווים היא הפיכת משמעות המחלקה – כלומר – "כל תו שאינו מופיע במחלקת התווים". לדוגמא, עבור התבנית `"[^a-z]a"` נקבל את ההתאמה הבאה:

Mary had **a** little lamb.
And everywhere that **Ma**ry
went, the lamb was sure
to go.

שפת הסימונים שלנו מכילה סימנים/קבוצות סימנים להם משמעות מיוחדת:
תווים מיוחדים:

- "\t" מתאים ל-TAB.
- "\n" מתאים לסוף שורה.

קבוצות מיוחדות:

- "\d" מתאימה לכל מספר דצימלי. יש לה משמעות זהה לתבנית "[0-9]".
- "\D" מתאימה לכל תו שאיננו מספר דצימלי. היא זהה לתבנית "[^0-9]".
- "\s" מתאימה לכל ה-whitespaces. יש לה משמעות זהה לתבנית "[\t\n\r\f\v]".
- "\S" מתאימה לכל תו שאינו whitespace. יש לה משמעות זהה לתבנית "[^ \t\n\r\f\v]"
- "\w" מתאימה לכל אות או מספר. היא זהה לתבנית "[a-zA-Z0-9_]".
- "\W" מתאימה לכל תו שאיננו אות או מספר. היא זהה לתבנית "[^a-zA-Z0-9_]".

התו | משמש כאופרטור OR.

- התבנית "hi|hello" מתאימה לכל המחרוזות בהן המילה "hi" או המילה "hello".
- התבנית "(a|b)c" מתאימה לכל המחרוזות בהן האות a או b, ולאחריה האות c.

<u>חזרות</u>

התווים "*", "+" ו-"?" מציינים תו/תווים החוזרים על עצמם מספר פעמים.
"*" – התו חוזר אפס פעמים או יותר.
"+" – התו חוזר פעם אחת או יותר.
"?" – התו מופיע אפס פעמים או פעם אחת בלבד.

דוגמאות:

- התבנית "ab*" מתאימה מחרוזות שיש בהן a, ולאחריו אפס או יותר b-ים: "a", "ab", "abbb" וכו'.
- התבנית "ab+" מתאימה מחרוזות שיש בהן a, ולאחריו לפחות b אחד: "ab", "abbb" וכו'.
- התבנית "ab?" מתאימה מחרוזות שיש בהן a, ולאחריו ייתכן שיש b או לא.

ניתן להשתמש בגבולות כדי להגדיר את מספר חזרות. נעשה זאת על ידי שימוש בסוגריים מסולסלות והגדרת התחום של מספר ההופעות הרצוי.

- התבנית "ab{2}" מתאימה מחרוזות שיש בהן a ולאחריו שני b-ים: "abb".
- התבנית "ab{2,}" מתאימה מחרוזות שיש בהן a ולאחריו לפחות שני b-ים: "abb", "abbbb" וכו'.
- התבנית "ab{3, 5}" מתאימה מחרוזות שיש בהן a ולאחריו בין 3 ל-5 b-ים: "abbb", "abbbb", "abbbbb".
- התבנית "^.{3}$" תתאים לכל מחרוזת שיש בה בדיוק 3 תווים.

ניתן לסמן גם אטומים שיחזרו על עצמם מספר פעמים.

- התבנית "(ab)+" מתאימה מחרוזות בהן מופיע ab לפחות פעם אחת ברצף: "ab", "abab" וכו'.
- התבנית "a(ab){2, 5}" מתאימה מחרוזות המתחילות ב-a, ולאחריו הרצף ab המופיע בין פעמיים לחמש פעמים: "aabab", "aababab" וכו'.

חמדנות

מנוע ה-regex הוא בעל תכונת חמדנות – כאשר ניתן לו מחרוזת ותבנית הוא יחפש את תת המחרוזת הגדולה ביותר העומדת בדרישה.
לדוגמא, נניח שיש בידינו טקסט שאנו רוצים למצוא את כל המילים בו המתחילות ב-th ומסתיימות ב-s.
ננסה את התבנית הבאה: "th.*s". נקבל:

```
-- I want to match the words that start
-- with 'th' and end with 's'.
this line matches just right
this # thus # thistle
```

על ידי שימוש באופרטור ? אחרי אופרטור *, אנו אומרים למנוע ה-Regex למצוא את תת המחרוזת הקטנה ביותר המתאימה, ולא הגדולה ביותר. ננסה את התבנית "th.*?s", ונקבל:

```
-- I want to match the words that start
-- with 'th' and end with 's'.
this # thus # thistle
this line matches just right
```

כדי להגיע רק אל המילים הרצויות, נוסיף רווח בסוף התבנית, כך: "th.*?s " ונקבל:

```
-- I want to match the words that start
-- with 'th' and end with 's'. (FINALLY!)
this # thus # thistle
this line matches just right
```

דוגמא מסכמת

נפתח כעת בשלבים תבנית שתקבל מספר, האמור לייצג סכום כסף.

נתחיל מהתבנית הזו: "^[1-9][0-9]*$". התבנית תמצא התאמה בכל מספר שלם שאינו אפס.
אולם, 0 הוא מספר שיכול לייצג סכום כסף חוקי. לפיכך נשנה את התבנית: "^(0|[1-9][0-9]*)$".
כעת היא תקבל גם את המספר אפס.
אם נרצה לייצג גם מספר שליליים נצטרך לשנות שוב את התבנית: "^(0|-?[1-9][0-9]*)$"

כעת ננסה לזהות מספרים, אולם נשנה מעט את ההגבלות. איננו רוצים שהתבנית תקבל מספרים שליליים, אולם נחליט כי המספר יכול להתחיל ב-0. כמו כן, נרצה לתמוך במספרים עשרוניים כגון 3.32 וכו'.
נשתמש בתבנית הבאה: "^[0-9]+(\.[0-9]+)?$". נשים לב שלפי תבנית זו, "10." זוהי איננה התאמה, ואילו "10" או "10.2" הן התאמות.
נוכל להגביל את המחרוזת לשתי ספרות אחרי הנקודה בלבד: "^[0-9]+(\.[0-9]{2})?$".
נוכל להיות פחות קשוחים, ולאפשר ספרה אחת או שתיים לאחר הנקודה:
"^[0-9]+(\.[0-9]{1,2})?$".
לסיום נרצה לתמוך גם באפשרות שבמחרוזת יהיו פסיקים, להפריד בין אלפים. למשל: 122,722.
נשתמש בתבנית הבאה: "^[0-9]{1,3}(,[0-9]{3})*(\.[0-9]{1,2})?$"

EOF