

التعبيرات النظامية/الاعتيادية

**REGULAR EXPRESSIONS**

- المفردات التي يجرؤها محلل المفردات إلى بطاقات tokens تتبع قواعد rules تصف فئات/أنواع المفردات الموجودة بالبرنامج المصدري.
- هذه القواعد تسمى أنماط المفردات
- **النمط:** هو وصف للشكل الذي تأخذه المفردات في بطاقة ما a token وهو بنية مركبة complex تتوافق عليها مفردات عديدة
  - بطاقة الكلمات المحجوزة تتبع نمطاً <keyword>
    - سلسلة من الحروف
  - بطاقة المعارف/المتغيرات تتبع نمطاً <identifier>
    - سلسلة من الحروف والأرقام

# EXAMPLES OF TOKENS

## أمثلة للبطاقات

```
int Index;  
Index = 2 * count +17;
```

Lexemes	Tokens
int	type
Index	identifier
=	equal_sign
2	int_constant
*	multi_op
Count	identifier
+	plus_op
17	int_constant
;	semicolon

Identifiers are the names of the variables

المعرفات هي أسماء المتغيرات

## مواصفات نمط المفردات

- نحتاج لصياغة قادرة على التعبير عن أنماط البطاقات
- التعبيرات النظامية Regular Expressions تستخدم لوصف نمط صياغة المفردات
- **مفردات** أي لغة يعبر عنها من خلال الحروف الهجائية لتلك اللغة
  - ❖ اللغة العربية, حروفها الهجائية التي تشكل جميع كلماتها هي أ – ي
  - ❖ اللغة الإنجليزية, حروفها الهجائية alphabet هي a – z, A - Z

## GREEK ALPHABET

Αα

**ALPHA**  
*Al-fah*

Ββ

**BETA**  
*Bay-tah*

Γγ

**GAMMA**  
*Gam-ah*

Δδ

**DELTA**  
*Del-tah*

Εε

**EPSILON**  
*Ep-si-lon*

Ζζ

**ZETA**  
*Zay-tah*

Ηη

**ETA**  
*Ay-tah*

Θθ

**THETA**  
*Thay-tah*

Ιι

**IOTA**  
*Eye-o-tah*

Κκ

**KAPPA**  
*Cap-ah*

Λλ

**LAMBDA**  
*Lamb-da*

Μμ

**MU**  
*Mew*

Νν

**NU**  
*New*

Ξξ

**XI**  
*Zeye (if it stands alone) or  
Zee (if followed by a letter)*

Οο

**OMICRON**  
*Om-i-cron*

Ππ

**PI**  
*Pie*

Ρρ

**RHO**  
*Row*

Σσς

**SIGMA**  
*Sig-mah*

Ττ

**TAU**  
*Taw*

Υυ

**UPSILON**  
*Oop-si-lon*

Φφ

**PHI**  
*Fee (if stands alone) or  
Feye (if followed by a letter)*

Χχ

**CHI**  
*Keye*

Ψψ

**PSI**  
*Sigh*

Ωω

**OMEGA**  
*O-may-gah*

- تعتمد التعبيرات النظامية Regular Expressions على مبادئ مادة التراكيب

## المنفصلة Discrete Structures

- يعبر عن الحروف الهجائية للغة بالرمز الإغريقي  $\Sigma$
- حيث  $\Sigma$  عبارة عن فئة محدودة finite تحتوي على جميع الرموز (حروف وأرقام) وأيضاً العلامات (الفواصل, النقاط, وغيرها) التي يمكن أن تشكل **جملة** في لغة ما
- اللغة الإنجليزية:  $\Sigma = \{a-z, A-Z\}$

# مفردات فئة هجائية

- بفرض أن سيقما = فئة الحروف  $a, b, c, d$
- $\Sigma = \{a, b, c, d\}$
- المفردات الممكنة من الهجائية  $\Sigma$  هي:
  - $a$
  - $aa$
  - $aaa$
  - $aabbccdd$
  - $d$
  - $abab$
  - $ccccccccccccacccc$
- وهكذا, أي تركيبة من الحروف الأربعة  $a, b, c, d$

- الهجائية  $\Sigma$ : سيقما هي فئة محدودة finite تحتوي على كل مدخلات inputs الرموز characters أو العلامات symbols
- نهاية الهجائية  $\Sigma^*$ : سيقما ستار هي فئة كل المفردات الممكنة في سيقما  $\Sigma$ , ويشمل ذلك المفردة الخالية (إبسيلون) empty string  $\varepsilon$
- اللغة الرسمية **formal language L** هي فئة جزئية من  $\Sigma^*$  سيقما ستار
- فهي فئة المفردات ذات المعنى في اللغة, كجزء من جميع المفردات الممكنة سيقما ستار  $\Sigma^*$



- الاتحاد بين لغتين  $L$  و  $M$  Union: هو فئة المفردات التي تنتمي على الأقل لأحد اللغتين  $L$  أو  $M$ , "على الأقل" تعني أن بعض المفردات قد تنتمي لكلا اللغتين.

$$L \cup M = \{s | s \in L \text{ or } s \in M\}$$

- مثال:

- $H = \{abb, baa, aba, bab\}$ ,  $K = \{doo, ree, mee, baa\}$ 
  - $H \cup K = \{abb, baa, aba, bab, doo, ree, mee\}$

- التقاطع بين لغتين  $L$  و  $M$  Intersection: هو فئة المفردات التي تنتمي إلى كلا اللغتين.

$$L \cap M = \{s | s \in L \text{ and } s \in M\}$$

- مثال:

- $L = \{a, aa, aaa, aaaa\}, M = \{bd, bbdd, bdbd\}$ 
  - $L \cap M = \{\epsilon\}$  إبسيلون: وهي فئة خالية

• لصق/اربط لغتين  $L$  و  $M$  Concatination: فئة كل المفردات على التشكيل

$st$  حيث  $s$  عبارة عن مفردة من اللغة  $L$  و  $t$  عبارة عن مفردة من اللغة  $M$

$$LM = \{st | s \in L \text{ and } t \in M\}$$

• مثال:

•  $L = \{a, aa\}, M = \{bd, bbdd\}$

•  $LM = \{abd, abbdd, aabd, aabbdd\}$

## عمليات اللغات التشكيلية (الرسمية)

- نهاية كليين اللغة  $L$  (Kleene closure) مسماة عن شخص اسمه كليين: فئة

كل المفردات الناجمة عن لصق 0 مفردة أو أكثر من مفردات لغة ما  $L$

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

عندما  $i = 0$ ,  $L^0$  تعني لصق صفر من مفردات اللغة  $L$  وهو يساوي فئة خالية  $\{\epsilon\}$ , وهي تعني لاشئ.

مثلاً:  $L = \{at, bat, cat\}$

$$L^* = \left\{ \begin{array}{l} \epsilon, at, atat, bat, cat, atbat, atcat, batat, \\ batcat, catat, catbat, \\ atbatcat, batatcat, catatbat, \dots \end{array} \right\}$$

أي فئة كل احتمالات لصق لمفردات اللغة بما فيها الفئة الخالية

## عمليات اللغات التشكيلية (الرسمية)

- نهاية إيجابية للغة  $L$  (Positive closure): فئة كل المفردات الناجمة عن لصق مفردة واحدة أو أكثر من مفردات لغة ما  $L$ , أي لا يوجد فئة خالية  $i \neq 0$

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

مثلاً:  $L = \{at, bat, cat\}$

$$L^+ = \left\{ \begin{array}{l} at, atat, bat, cat, atbat, atcat, batat, batcat, \\ catat, catbat, \\ atbatcat, batatcat, catatbat, \dots \end{array} \right\}$$

أي فئة كل احتمالات لصق لمفردات اللغة ولكن بدون الفئة الخالية

# عمليات اللغات التكوينية (الرسمية)

• مثلاً:

$L$  هي فئة الحروف الإنجليزية الصغيرة والكبيرة

$$L = \{A, B, \dots, Z, a, b, \dots, z\}$$

و  $D$  هي فئة الأرقام

$$D = \{0, 1, \dots, 9\}$$

فإن  $L(L \cup D)^*$  تعبر عن فئة كل المفردات من حروف وأرقام وتبدأ بحرف

؟

ويعتبر هذا تعبيراً عن نمط من أنماط صياغة

✓ أعط مثال:

- التعبيرات النظامية هي صياغة مختصرة لوصف نص المفردات
- المعرفات في لغات البرمجة عبارة عن حرف متبوع بصفرٍ أو أكثر من الحروف أو الأرقام
- Identifier  $\rightarrow$  letter(letter | digit)\*
- المتغير  $\leftarrow$  حرف(حرف|رقم)\*
- العمود | يعبر عن الاتحاد, أو
- الأقواس ( ) تستخدم لتجميع التعبيرات الجزئية
- النجمة \* : تعني حدوث صفر أو أكثر من ما بين الأقواس,

- التعبيرات النظامية يتم بناؤها بالتكرار **recursively** من خلال تعبيرات نظامية صغيرة, وذلك باتباع قواعد معينة **rules** وهي:

1. إبسيلون  $\epsilon$  تعبير نظامي يشير/تتبعي **denotes** إلى اللغة  $L(\epsilon)$  وتساوي  $\{\epsilon\}$ , وهي فئة خالية لا تحتوي على نص
  2. إذا كانت  $a$  رمز/علامة في الهجائية  $\Sigma$ , فإن  $a$  تكون تعبيراً نظامياً يشير إلى الفئة  $\{a\}$  وهي تحتوي على النص  $a$
- $a$  حرف من حروف سيقما, فإن وجدت لوحدها فهي تعبير نظامي للغة  $L(a)$  التي تساوي  $\{a\}$  كلغة من مفردة واحدة بطول رمز واحد



- تقليدياً, تكتب التعبيرات بالخط العريض, مثل  $a$ , وتكتب الرموز بالخط المائل

العادي, مثل  $a$ .  $L(a) = \{a\}$

لغة التعبير النظامي  $a$

- بما أن التعبيرات الكبيرة تتألف من تعبيرات نظامية صغيرة, فهذه 4 تعبيرات جزئية أساسية:

1.  $(r)|(s)$  هو تعبير "أو" ويشير إلى اتحاد اللغتين  $L(r) \cup L(s)$

2.  $(r)(s)$  هو تعبير يشير إلى لصق اللغتين  $L(r)L(s)$

3.  $(r)^*$  هو تعبير يشير إلى  $(L(r))^*$

4.  $(r)$  هو تعبير عن اللغة  $L(r)$

## أسبقيات التعبيرات النظامية

- التعبيرات النظامية تحتوي أقواس يمكن الاستغناء عنها بشرط اتباع أسبقيات مشغلاتها
  - المشغل \* له الأسبقية الأعلى
  - اللصق concatenation الأسبقية الثانية
  - المشغل | له الأسبقية الأخيرة
  - طبعاً قراءة التعبيرات من اليسار إلى اليمين
- إذن يمكن استبدال التعبير  $a|b^*c$  بالتعبير  $a((b)^*c)$  بالتعبير  $a|b^*c$
- كلا التعبيرين يشير إلى فئة المفردات وهي:  $a$  أو  $b$  أو عدد من  $b$  أو بلا  $b$  ثم تلحقها بـ  $c$

$\{a, c, bc, bbc, bbbc, bbbbc, bbbbbc, bbbbbbc, \dots\} =$

# أمثلة التعبيرات النظامية

- ما هي لغة التعبير  $(a|b)(a|b)$
- $\{aa,ab,ba,bb\}$  a أو b يلتصق مع a أو b
- هل يمكنك كتابة تعبير آخر يعطي نفس اللغة الناتجة من التعبير  $(a|b)(a|b)$
- $aa|ab|ba|bb$
- ما هي لغة التعبير  $a^*$
- $\{\epsilon, a, aa, aaa, aaaa, \dots\}$  لاشئ, و كل احتمالات a
- ما هي لغة التعبير  $(a|b)^*$
- $\{\epsilon, a, b, \dots\}$  لاشئ, و كل احتمالات a أو احتمالات b
- ما هي لغة التعبير  $a|a^*b$
- $\{a,b,ab,aab,aaab, \dots\}$  المفردة a مع 0 أو عدد من a مع b

مثال رياضي	وصف	قانون المسألة Axiom
$3+6 = 3+6$	تبديلية commutative	$r s = s r$
$4+(3+6) = (4+3)+6$	ترابطية associative	$r (s t) = (r s) t$
	اللتصق ترابطية	$(rs)t = r(st)$
$4(3+6) = (4 \times 3 + 4 \times 6)$	اللتصق توزيعي على	$r(s t) = rs rt$ $(s t)r = sr tr$
$3+0 = 3$ $0+3 = 3$	ε محايد لللتصق identity	$\epsilon r = r$ $r\epsilon = r$
	علاقة ε مع *	$r^* = (r \epsilon)^*$
	* تكرارها كحدوثها مرة واحدة idempotent	$r^{**} = r^*$

- يمكن كتابة صيغة لتعريفات نظامية على النحو التالي:

- $d_1 \rightarrow r_1$

- $d_2 \rightarrow r_2$

- ...

- $d_n \rightarrow r_n$

- حيث أن كل  $d_i$  عبارة عن اسم لتعريف definition يطلع على تعبير نظامي  $r_i$

- فئة هجائيات هذه التعريفات تتكون من جميع الرموز التي تعبر عنها التعريفات  $\Sigma \cup \{d_1, d_2, \dots, d_n\}$  وتساوي جميع رموز هذه التعريفات

$letter \rightarrow (a|b|c|...|z|A|B|C|...|Z)$

$digit \rightarrow (0|1|2|3|4|5|6|7|8|9)$

$id \rightarrow letter(letter|digit)^*$

$integer \rightarrow (+|-|\epsilon)digit^+$

$decimal \rightarrow integer.digit^*$

$real \rightarrow (integer|decimal)E(+|-)digit^+$

*letter*  $\rightarrow (a|b|c|...|z|A|B|C|...|Z)$

*letter*  $\rightarrow [a - z A - Z]$  : يمكن أن تختصر:

*digit*  $\rightarrow (0|1|2|3|4|5|6|7|8|9)$

*digit*  $\rightarrow [0 - 9]$  : يمكن أن تختصر:

# REGULAR DEFINITIONS

## أمثلة لتعريفات نظامية

*if* → if

*then* → then

*else* → else

*relop* → < | > | = | <= | >= | <>

*while* → while

*int* → int



موضوعنا التالي:

تمارين عن التعبيرات النظامية