Smalltalk勉強会@京都
2009年2月4日　キャンパスプラザ京都　6F第2講習室

言語処理系としてのSmalltalk

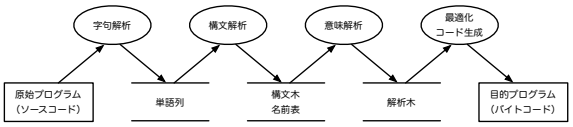
（メッセージパッシングを支える処理系の初歩）

青木 淳
atsushi@cc.kyoto-su.ac.jp
http://www.cc.kyoto-su.ac.jp/~atsushi/

京都産業大学コンピュータ理工学部
研究室： 1号館 4階 1402研究室
客員研究員： 浅岡 浩子 ・ 澤本 依里

では、では、さっそく…

コンパイラ・インタプリタ・仮想マシンを有する
プログラミング言語処理系「Smalltalk」




```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```

言語処理系の構成

```
D8 0A 4C D8 14 4D D8 1E 4E
10 11 12 A8 A0 4F 13 65
```

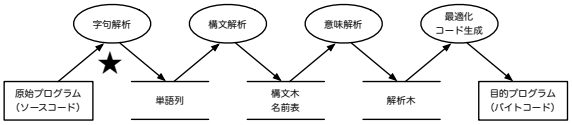
http://www.cc.kyoto-su.ac.jp/~atsushi/Programs/20080908/
参考ページ：言語処理系の構成をプログラムで示しておきました。

原始プログラム（ソースコード）

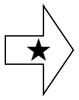
```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```


字句解析

http://www.cc.kyoto-su.ac.jp/~atsushi/Programs/20080908/




```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```



```
#| #aaa #bbb #ccc #ddd #| #aaa #':'
#= 10 #'.' #bbb #':' #= 20 #'.' #ccc
#':' #= 30 #'.' #ddd #':' #= #aaa #+
#(#bbb #* #ccc) #'.' #'^' #ddd
```


単語列（トークン並び）

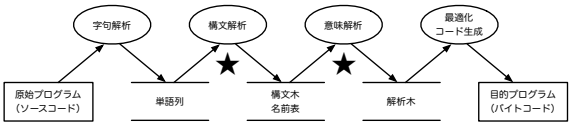
ソースプログラム（原始プログラム）をスキャン（走査）して、ソースプログラムを構成している単語（字句）を順番に列挙したもの。

```
#| #aaa #bbb #ccc #ddd #| #aaa #':'
#= 10 #'.' #bbb #':' #= 20 #'.' #ccc
#':' #= 30 #'.' #ddd #':' #= #aaa #+
#(#bbb #* #ccc) #'.' #'^' #ddd
```

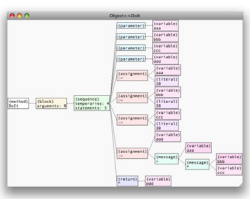
しばしば、ソースプログラムを構成している単語（字句）をトークン（token）と呼びます。

構文解析・意味解析

http://www.cc.kyoto-su.ac.jp/~atsushi/Programs/20080908/

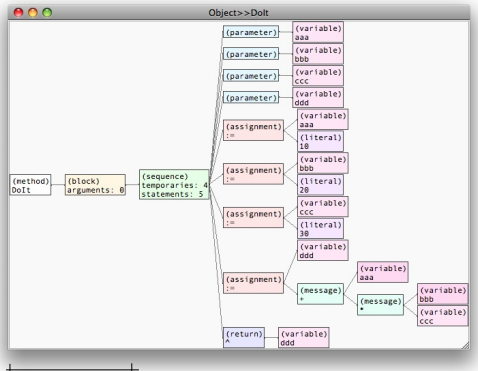



```
#| #aaa #bbb #ccc #ddd #| #aaa #':'
#= 10 #'.' #bbb #':' #= 20 #'.' #ccc
#':' #= 30 #'.' #ddd #':' #= #aaa #+
#(#bbb #* #ccc) #'.' #'^' #ddd
```



構文木・名前表・解析木

構文木
解析木



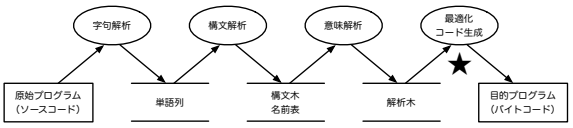
名前表

aaa	: local 0
bbb	: local 1
ccc	: local 2
ddd	: local 3

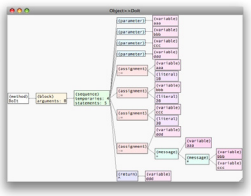
しばしば、名前表はシンボルテーブル（symbol table）と呼ばれます。要は名前から番地がひけるテーブルです。

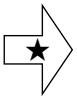
最適化・コード生成

http://www.cc.kyoto-su.ac.jp/~atsushi/Programs/20080908/




```
D8 0A 4C D8 14 4D D8 1E 4E 10 11 12 A8 A0 4F 13 65
```





```
 1 <D8 0A> push 10
 3 <4C> store local 0; pop
 4 <D8 14> push 20
 6 <4D> store local 1; pop
 7 <D8 1E> push 30
 9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```


目的プログラム（バイトコード）

```
D8 0A 4C D8 14 4D D8 1E 4E 10 11 12 A8 A0 4F 13 65
```

```
 1 <D8 0A> push 10
 3 <4C> store local 0; pop
 4 <D8 14> push 20
 6 <4D> store local 1; pop
 7 <D8 1E> push 30
 9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

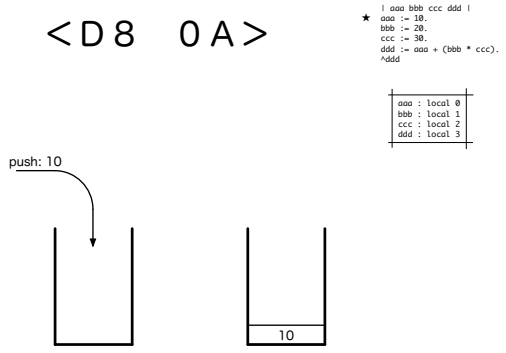
機械語＝マシン（コンピュータ）が理解できる言語に変換されていますね？

スタック操作（stack　operation）の列に変換されていることがわかりますか？

仮想マシンはバイトコード（00〜FF）という機械語を高速に実行する「ソフトウェアで作られたハードウェア」です。

バイトコードには、
【スタック操作命令】
【メッセージ送信命令】
【分岐命令】
【リターン命令】
があります。

http://users.ipa.net/~dwighth/smalltalk/bluebook/bluebook_chapter28.html



★
```
 1 <D8 0A> push 10
 3 <4C> store local 0; pop
 4 <D8 14> push 20
 6 <4D> store local 1; pop
 7 <D8 1E> push 30
 9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

＜D8　0A＞

★
```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```

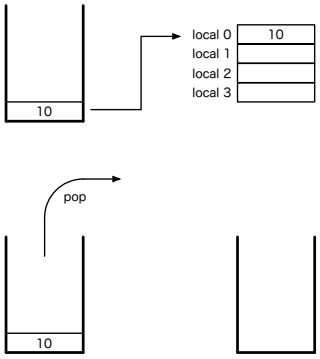
aaa	: local 0
bbb	: local 1
ccc	: local 2
ddd	: local 3

★
```
 1 <D8 0A> push 10
 3 <4C> store local 0; pop
 4 <D8 14> push 20
 6 <4D> store local 1; pop
 7 <D8 1E> push 30
 9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

＜4C＞

★
```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```

aaa	: local 0
bbb	: local 1
ccc	: local 2
ddd	: local 3

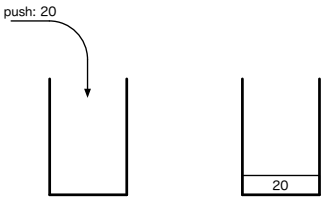


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<D8 14>

```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3

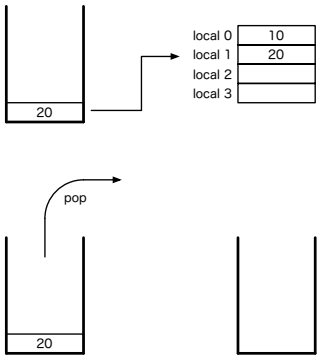



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<4D>

```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3

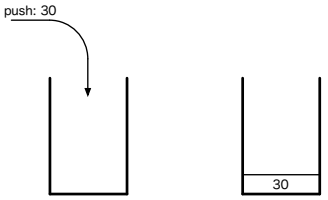



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<D8 1E>

```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3

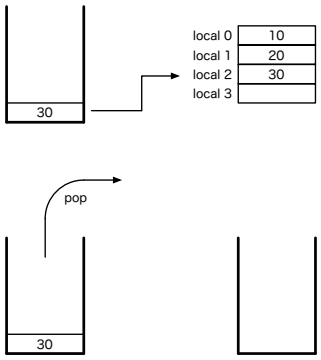



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<4E>

```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3

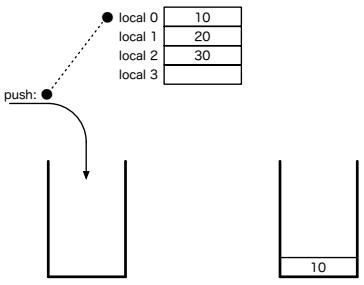



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<10>

```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3

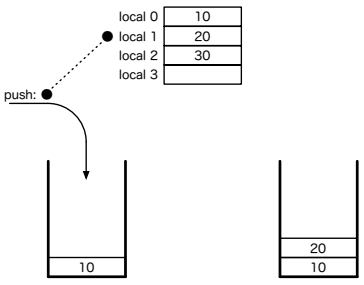



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<11>

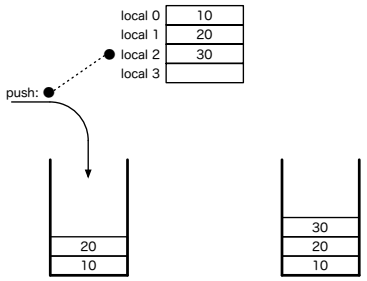
```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3



```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
★ 12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<12>

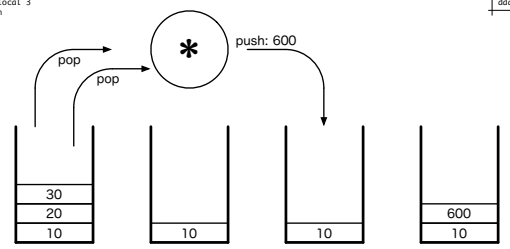



```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
★ 13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<A8>

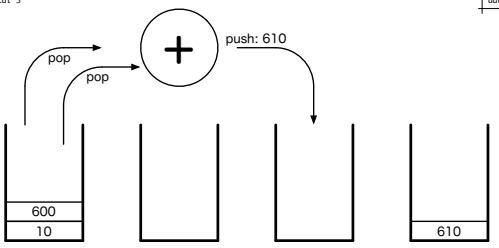



```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
★ 14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<A0>

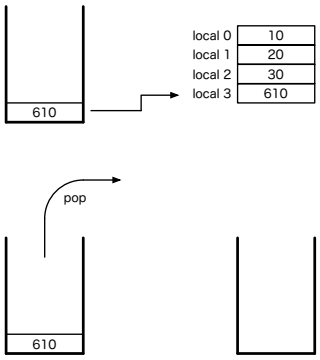



```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
★ 15 <4F> store local 3; pop
16 <13> push local 3
17 <65> return
```

<4F>

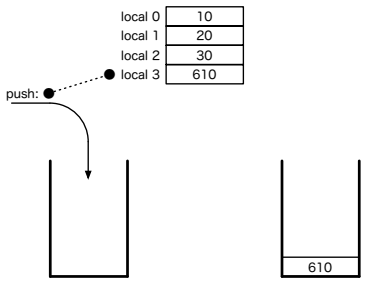



```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
★ 16 <13> push local 3
17 <65> return
```

<13>

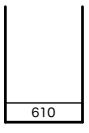



```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3


```
1 <D8 0A> push 10
3 <4C> store local 0; pop
4 <D8 14> push 20
6 <4D> store local 1; pop
7 <D8 1E> push 30
9 <4E> store local 2; pop
10 <10> push local 0
11 <11> push local 1
12 <12> push local 2
13 <A8> send *
14 <A0> send +
15 <4F> store local 3; pop
16 <13> push local 3
★ 17 <65> return
```

<65>




```
| aaa bbb ccc ddd |
aaa :- 10.
bbb :- 20.
ccc :- 30.
★ ddd :- aaa + (bbb * ccc).
^ddd
```

aaa : local 0
bbb : local 1
ccc : local 2
ddd : local 3