京都産業大学　コンピュータ理工学部　CS・NM・IS学科　プログラミング言語

開講にあたって

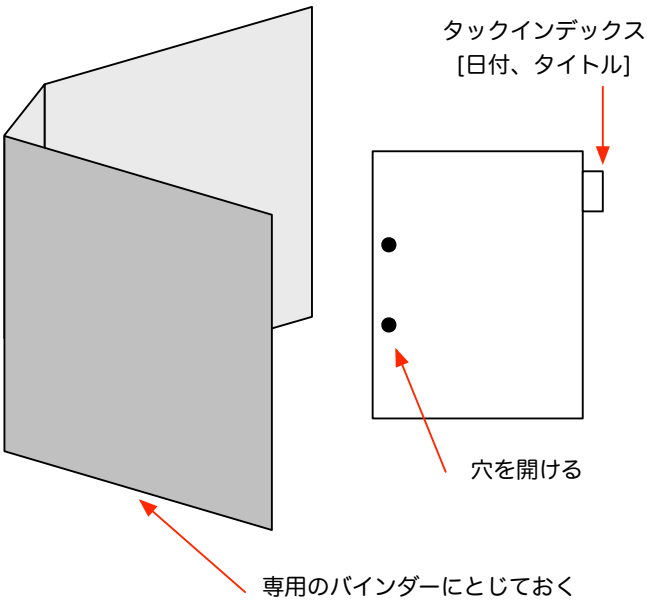
(授業目的・授業内容・評価方法・教材など)

まずやってみる　それから学ぶ

青木 淳
aokisanhe@gmail.com
http://www.cc.kyoto-su.ac.jp/~atsushi/
研究室： 第2実験室棟 3階 73研究室

配付資料を大切に

皆さんには釈迦に説法のようで失礼ですが…



シラバスを最初に入れる

順次、配付資料を
綴じ込んでゆこう

青木の講義の受け方

パソコンを必ず持ってくること！（手習いがあります）

ペンを持たずに講義を受けてはなりませんよ！

配付資料の余白に書き込みをしてくださいね！

プログラミング言語を論ずるのに「絵図」を多用しています

配付資料を後で見直したときに、思い出せるようにメモを
レポートを書くときにも、思い出せるようなメモを

配付資料

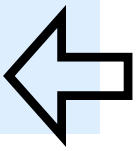
http://www.cc.kyoto-su.ac.jp/~atsushi/Students/

講義の配付資料

講義で援用するドキュメントやプログラムなどは、できる限り開示（ウェブにて公開を）しています。ドキュメント集やプログラム集のページをご覧ください。また、学習支援システムmoodleもチェックしてくださいね。講義の配付資料（zipアーカイブ）は以下からアクセス可能であり、その展開パラフレーズ（パスワード）は授業中にお知らせしています。

2014

- コンピュータシステム［秋学期：火曜3限］10201（10号館）--- 全学共通教育センター
  ［2014/09/23］［2014/09/30］［2014/10/07］［2014/10/14］［2014/10/21］
  ［2014/10/28］［2014/11/11］［2014/11/18］［2014/11/25］［2014/12/02］
  ［2014/12/09］［2014/12/16］［2014/12/23］［2015/01/13］［2015/01/20］
- コンピュータ理工学部特別研究IIB［秋学期：水曜4限5限］65実験室（第2実験室棟）--- コンピュータ理工学部
  ［2014/09/24］［2014/10/01］［2014/10/08］［2014/10/15］［2014/10/22］
  ［2014/10/29］［2014/11/05］［2014/11/12］［2014/11/19］［2014/12/03］
  ［2014/12/10］［2014/12/17］［2014/12/24］［2015/01/07］［2014/01/14］
- コンピュータ理工学部特別研究I［秋学期：木曜4限5限］65実験室（第2実験室棟）--- コンピュータ理工学部
  ［2014/09/25］［2014/10/02］［2014/10/09］［2014/10/16］［2014/10/23］
  ［2014/10/30］［2014/11/06］［2014/11/13］［2014/11/20］［2014/11/27］
  ［2014/12/04］［2014/12/11］［2014/12/18］［2014/12/25］［2014/01/15］
- 基礎セミナーB［秋学期：金曜1限］105（1号館）--- コンピュータ理工学部
  ［2014/09/26］［2014/10/03］［2014/10/10］［2014/10/17］［2014/10/24］
  ［2014/11/07］［2014/11/14］［2014/11/21］［2014/11/28］［2014/12/05］
  ［2014/12/12］［2014/12/19］［2014/12/26］［2014/01/09］［2014/01/16］
- プログラミング言語［秋学期：金曜4限］14103（14号館）--- コンピュータ理工学部
  ［2014/09/26］［2014/10/03］［2014/10/10］［2014/10/17］［2014/10/24］
  ［2014/11/07］［2014/11/14］［2014/11/21］［2014/11/28］［2014/12/05］
  ［2014/12/12］［2014/12/19］［2014/12/26］［2014/01/09］［2014/01/16］

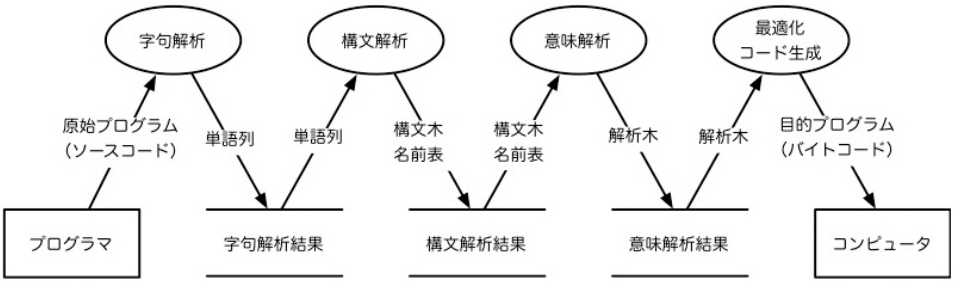


プログラミング言語

コンピュータを実際に動かす言語（たとえば機械語やアセンブリ言語）と私たちがプログラミングに用いる言語（たとえばオブジェクト指向プログラミング言語）との間には大きな溝があります。その溝を埋めるために情報技術の枠が結集しており、そこにはコンピュータの発展の歴史が詰まっていると言っても過言ではありません。

授業を通して言語処理系のモダンな構成を習得すると共に、プログラミング言語の処理系の内部で行われていることを垣間見ながら、プログラミング言語は使い切るものであるが、作り出すものでもあることの会得を目的とします。

授業中にプログラミング言語の変換系を作成します！



シラバスを参照しましょう

科目名	プログラミング言語						
英語科目		ナンバリング	Isft304				
開講期	秋学期	開講学部等	コンピュータ理工学部	配当年次	3年次	単位数	2単位
教員名	青木　淳						

※履修条件，配当年次等の詳細は履修要項をご確認ください。

授業概要／Course outline

コンピュータを実際に動かす言語（たとえば機械語）と私たちがプログラミングに用いる言語（たとえばオブジェクト指向プログラミング言語）の間に大きな溝があることは周知であろう。その溝を埋めるために情報技術（IT）の枠が結集しており、コンピュータの発展の歴史が詰まっていると言っても過言ではない。

その枠な部分（言語処理系の内部）を学びながら、コマンドインタプリタ（たとえばシェル）やスクリプト言語（たとえばPythonやAppleScript）、そして、仮想化技術（たとえばSmalltalkやJavaの仮想マシン方式）などのモダンな言語処理系（および実行系）の構成方法を知ることが目的である。

また、1年次と2年次に学んできた手続きプログラミング（C言語）とオブジェクト指向プログラミング（Java言語）に加えて、論理プログラミングと関数プログラミングについても学習する。様々なプログラミングのスタイルを垣間見ることで、新たなプログラミング言語やスタイルに出会ったときの咀嚼力と免疫力を向上させる。

本科目では、字句解析器（レクサ）や構文解析器（パーザ）の生成系（メタ系）を援用しながら、小さな言語処理系を自作するので、いままで学習してきたことの総決算（パノラマ）ができる。コンピュータに対するビジョン（鳥瞰した地図）とプログラミングの何たるか（メタプログラミングの視座）の会得も可能となる。

授業内容・授業計画／Course description・plan

この授業では「まず、やってみる、それから、学ぶ」をライトモチーフにしながら進めてゆく。毎回の授業の終わりに必ず「今日のレポート」を課す。今日のレポートであるから、授業日の23:59:59までに、指定された形式と方法でレポートを提出することを介して、学んだことや習ったことを確固たるものにする。

［第01回］

テーマ：開講にあたって

テーマ：言語処理系の概観

授業を始めるにあたって、授業目的・授業内容・評価方法・教材などを説明した後、言語処理系を構成している各々のフェーズ、字句解析・構文解析・意味解析・コード生成・最適化、それぞれをざっくりとデモンストレーションによって概観する。

［第02回］

テーマ：言語処理系の概観

テーマ：実行方式は仮想マシンへ

前回授業の復習から導入し、古典的な言語処理系と現代的な言語処理系の相違を示す。機械語に相当するスタック操作命令やメッセージ送信命令などのバイトコードを解説し、インタプリタの動きをデモンストレーションすることで、仮想マシンの存在を明

ダウンロードと展開

```
$ mkdir -p ~/PL
$ cd ~/PL
$ curl -O http://www.cc.kyoto-su.ac.jp/~atsushi/misc/zips/vw791jun797mac.zip
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 46.6M  100 46.6M    0     0  27.5M      0  0:00:01  0:00:01 --:--:-- 34.3M
$ ls -l vw791jun797mac.zip
-rw-r--r--  1 aoki  staff  48949872  8 30 10:18 vw791jun797mac.zip
$ unzip vw791jun797mac.zip
Archive:  vw791jun797mac.zip
   creating: VisualWorks791pulWithJun797ForMac/
   creating: VisualWorks791pulWithJun797ForMac/Smalltalk.app/
             :
  inflating: VisualWorks791pulWithJun797ForMac/VisualWorksWithJun/vw7.9.1pul/store/
StorePatchParcelSupport.pst
  inflating: VisualWorks791pulWithJun797ForMac/VisualWorksWithJun/vw7.9.1pul/Welcome.pdf
$ cd VisualWorks791pulWithJun797ForMac/
$ ls -l
total 0
drwxr-xr-x  3 aoki  staff  102  4  1  2011 Smalltalk.app
drwxr-xr-x  9 aoki  staff  306  7 29 13:06 VisualWorksWithJun
$ open Smalltalk.app
$
```

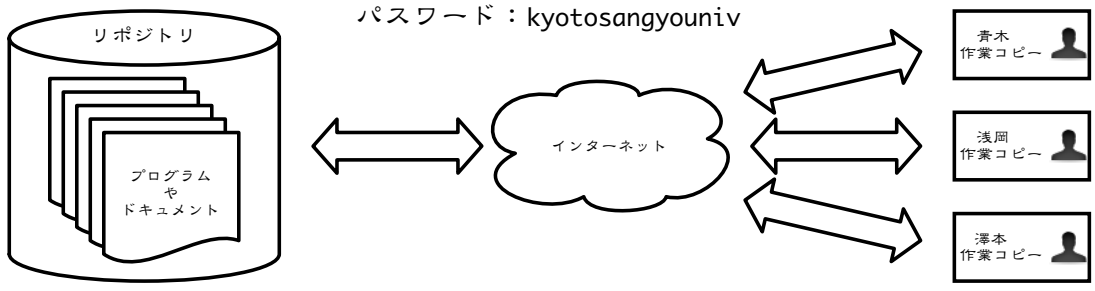
必ずNoで！

リポジトリ

ソフトウェア構成管理(Subversion)

http://axa010.cse.kyoto-su.ac.jp/repositories/PL/毎回ここを指示します/

ログイン名：student
パスワード：kyotosangyouniv



【チェックアウト：作業コピーを作る】

```
$ svn checkout --username student http://axa010.cse.kyoto-su.ac.jp/repositories/PL/毎回ここを指示します/
```

【更新：作業コピーを最新にする】

```
$ svn update
```

片づけが下手な人のためのマンガ本

池田暁子【片づけられない女のためのこんどこそ!片づける技術】文藝春秋

私の部屋が、もうずっとひっどいことになっていたのを必死で片づけたのですが、
その様子を描いたコミックエッセイです。どうぞよろしくお願いいたします！

http://www.amazon.co.jp/dp/4163690204/

整理が下手な人のためのマンガ本

池田暁子【必要なものがスグに!とり出せる整理術!】文藝春秋

汚部屋をやっとの思いで片づけ、床が出て普通の部屋（のよう）になったことにすっかり満足していたのですが、
家に来られた編集の方にマダマダな点を多々指摘され……(-_-)　押し入れの中、PC周りその他頑張って整理しました。

http://www.amazon.co.jp/dp/4840123411/

京都産業大学　コンピュータ理工学部　CS・NM・IS学科　プログラミング言語

言語処理系の概観

(字句解析・構文解析・意味解析・コード生成・最適化)

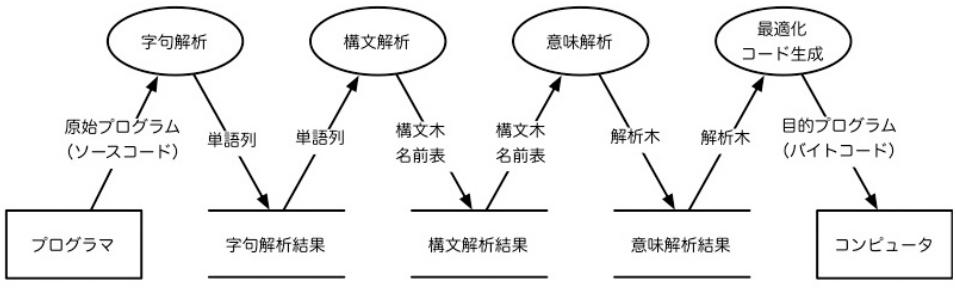
まずやってみる　それから学ぶ

青木 淳
aokisanhe@gmail.com
http://www.cc.kyoto-su.ac.jp/~atsushi/
研究室： 第2実験室棟 3階 73研究室

言語処理系を学ぶのに適切なのは

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やPython」がベスト！

言語処理系の構成




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

```
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/

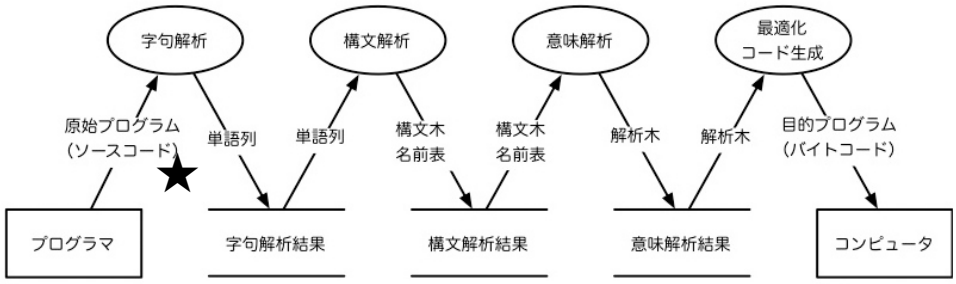
参考ページ：言語処理系の構成をプログラムで示しておきました。

原始プログラム（ソースコード）

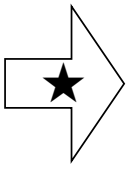
```
| aaa bbb ccc ddd |
aaa := 10.
bbb := 20.
ccc := 30.
ddd := aaa + (bbb * ccc).
^ddd
```


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）

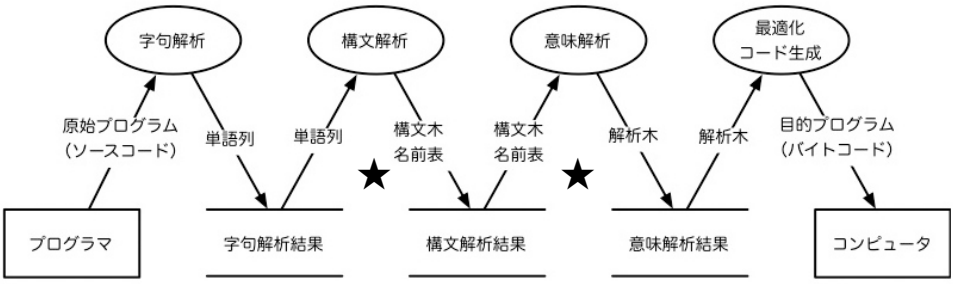
ソースプログラム（原始プログラム）をスキャン（走査）して、ソースプログラムを構成している単語（字句）を順番に列挙したもの。

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

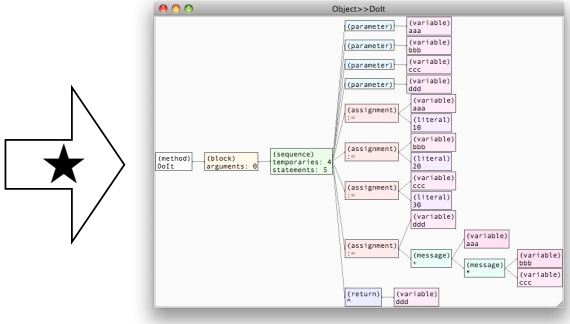
しばしば、ソースプログラムを構成している単語（字句）をトークン（token）と呼びます。

構文解析・意味解析

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

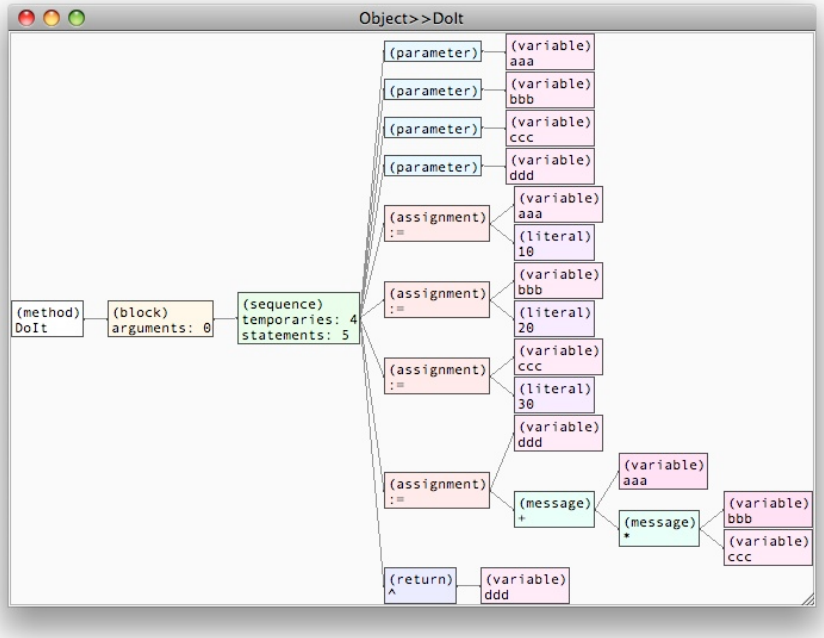



```
#| #aaa #bbb #ccc #ddd #| #aaa #':'
#= 10 #'.' #bbb #':' #= 20 #'.' #ccc
#':' #= 30 #'.' #ddd #':' #= #aaa #+
#(#bbb #* #ccc) #'.' #'^' #ddd
```



構文木・名前表・解析木

構文木
解析木



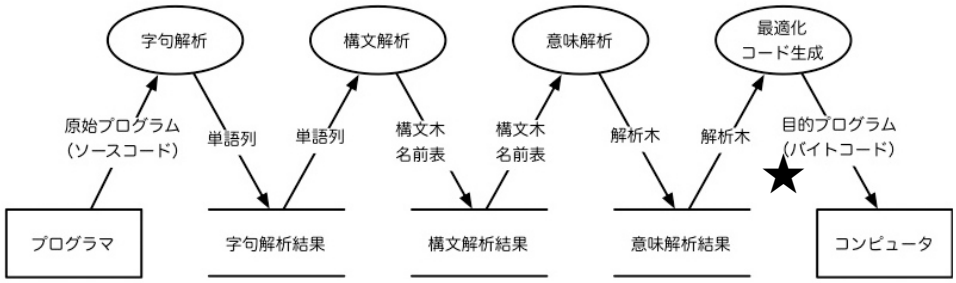
名前表

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

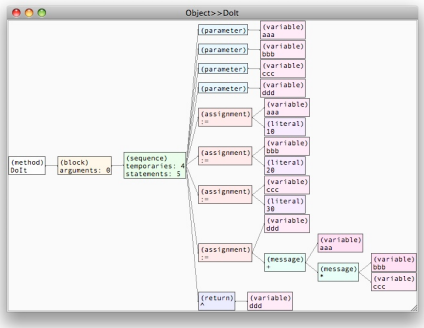
しばしば、名前表はシンボルテーブル（symbol table）と呼ばれます。要は名前から番地がひけるテーブルです。

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よね？

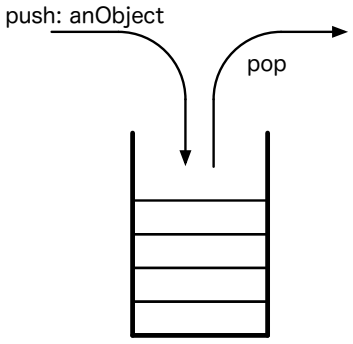
スタック操作（stack operation）の列に変換されていることがわかりますか？

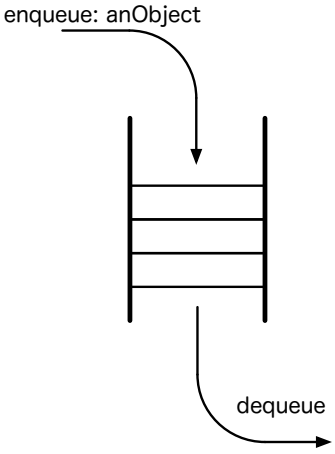
仮想マシンはバイトコード（00～FF）という機械語を高速に実行する「ソフトウェアで作られたハードウェア」です。

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。

詳しくは次のPDFをひもといてください。
http://stephane.ducasse.free.fr/FreeBooks/BlueBook/Bluebook.pdf

スタック（stack）とキュー（queue）



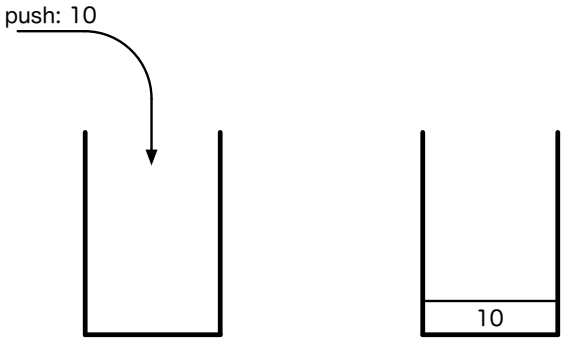


```
★ 1 <D8 0A> push 10
  3 <4C> store local 0; pop
  4 <D8 14> push 20
  6 <4D> store local 1; pop
  7 <D8 1E> push 30
  9 <4E> store local 2; pop
  10 <10> push local 0
  11 <11> push local 1
  12 <12> push local 2
  13 <A8> send *
  14 <A0> send +
  15 <4F> store local 3; pop
  16 <13> push local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

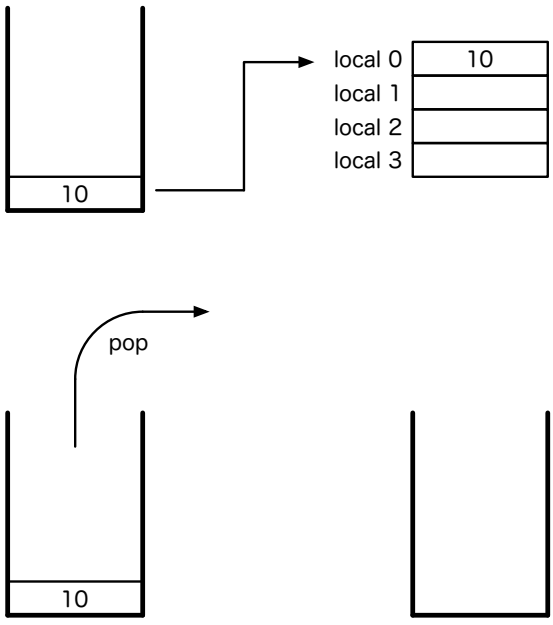



```
  1 <D8 0A> push 10
★ 3 <4C> store local 0; pop
  4 <D8 14> push 20
  6 <4D> store local 1; pop
  7 <D8 1E> push 30
  9 <4E> store local 2; pop
  10 <10> push local 0
  11 <11> push local 1
  12 <12> push local 2
  13 <A8> send *
  14 <A0> send +
  15 <4F> store local 3; pop
  16 <13> push local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

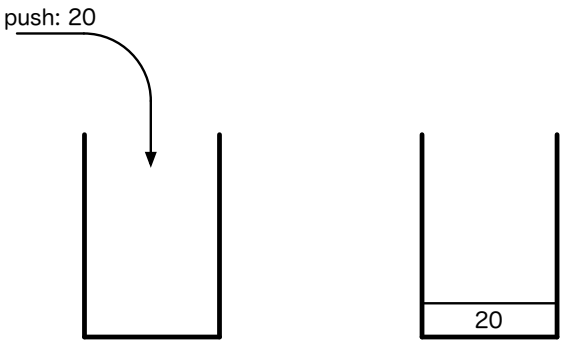


```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
★   4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
    10 <10> push local 0
    11 <11> push local 1
    12 <12> push local 2
    13 <A8> send *
    14 <A0> send +
    15 <4F> store local 3; pop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

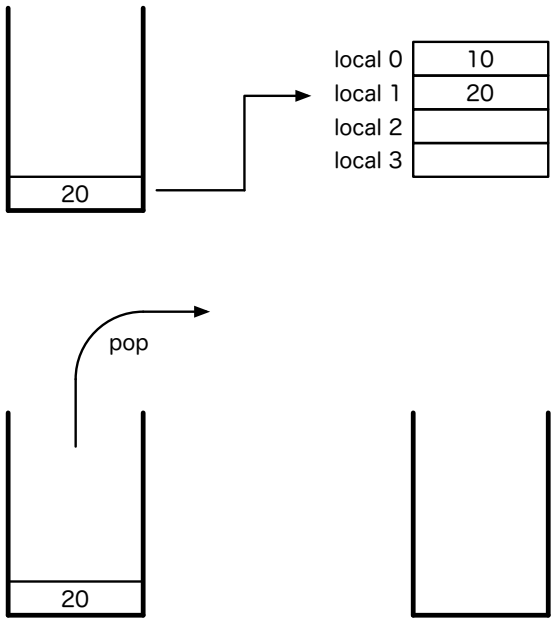



```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
★   6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
    10 <10> push local 0
    11 <11> push local 1
    12 <12> push local 2
    13 <A8> send *
    14 <A0> send +
    15 <4F> store local 3; pop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

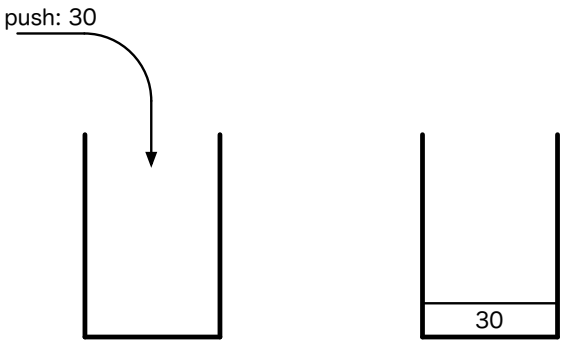


```
   1 <D8 0A> push 10
   3 <4C> store local 0; pop
   4 <D8 14> push 20
   6 <4D> store local 1; pop
★  7 <D8 1E> push 30
   9 <4E> store local 2; pop
  10 <10> push local 0
  11 <11> push local 1
  12 <12> push local 2
  13 <A8> send *
  14 <A0> send +
  15 <4F> store local 3; pop
  16 <13> push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

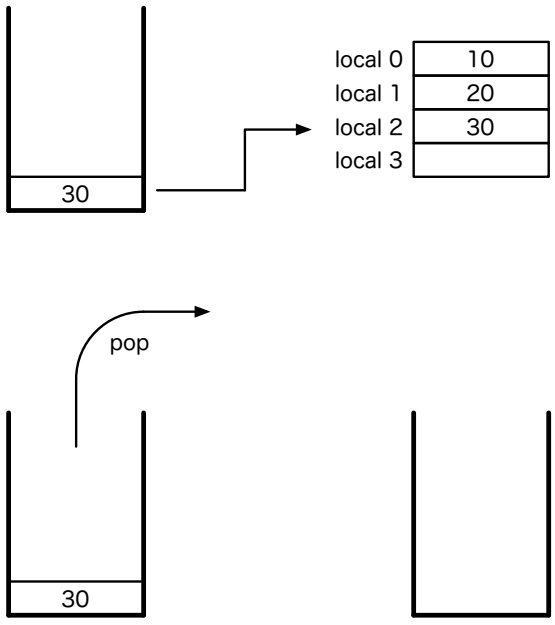



```
   1 <D8 0A> push 10
   3 <4C> store local 0; pop
   4 <D8 14> push 20
   6 <4D> store local 1; pop
   7 <D8 1E> push 30
★  9 <4E> store local 2; pop
  10 <10> push local 0
  11 <11> push local 1
  12 <12> push local 2
  13 <A8> send *
  14 <A0> send +
  15 <4F> store local 3; pop
  16 <13> push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

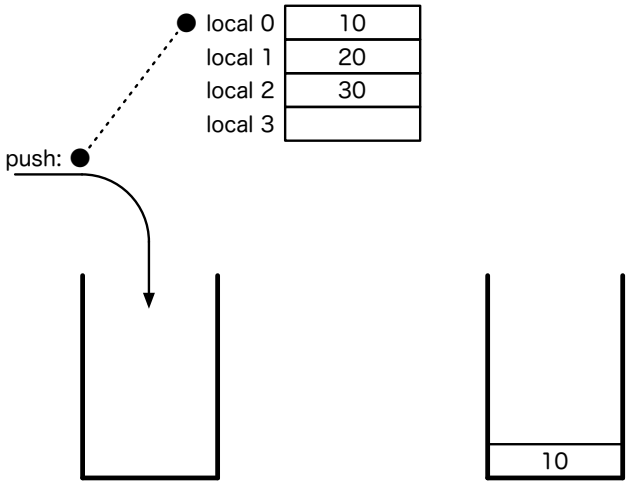


```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
★  10 <10> push local 0
   11 <11> push local 1
   12 <12> push local 2
   13 <A8> send *
   14 <A0> send +
   15 <4F> store local 3; pop
  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

< 1 0 >

```
| 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
```

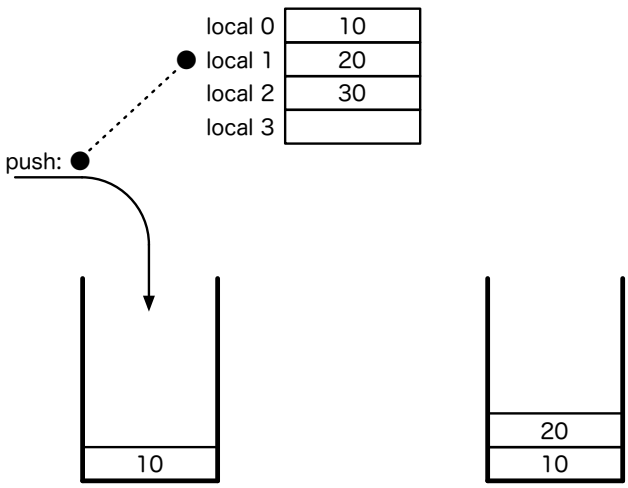



```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
★  11 <11> push local 1
   12 <12> push local 2
   13 <A8> send *
   14 <A0> send +
   15 <4F> store local 3; pop
  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

< 1 1 >

```
| 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
```

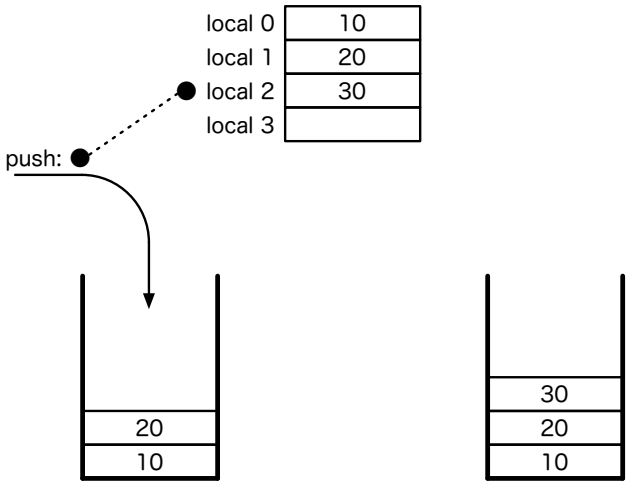


```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
   11 <11> push local 1
★  12 <12> push local 2
   13 <A8> send *
   14 <A0> send +
   15 <4F> store local 3; pop
  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

< 1 2 >

```
| 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
```

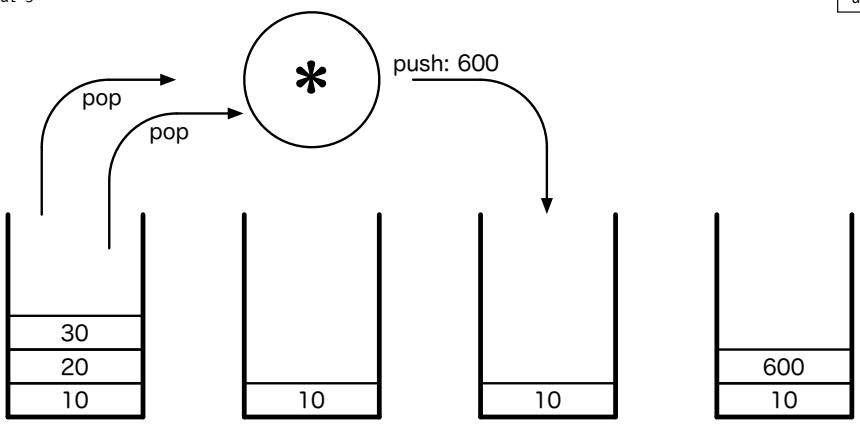



```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
   11 <11> push local 1
   12 <12> push local 2
★  13 <A8> send *
   14 <A0> send +
   15 <4F> store local 3; pop
  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

< A 8 >

```
| 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
```

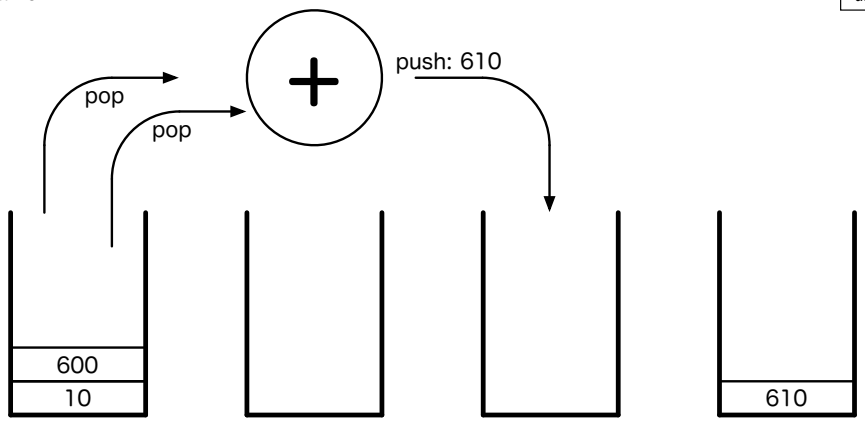


```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
   11 <11> push local 1
   12 <12> push local 2
   13 <A8> send *
★  14 <A0> send +
   15 <4F> store local 3; pop
  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

< A 0 >

```
| 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
```

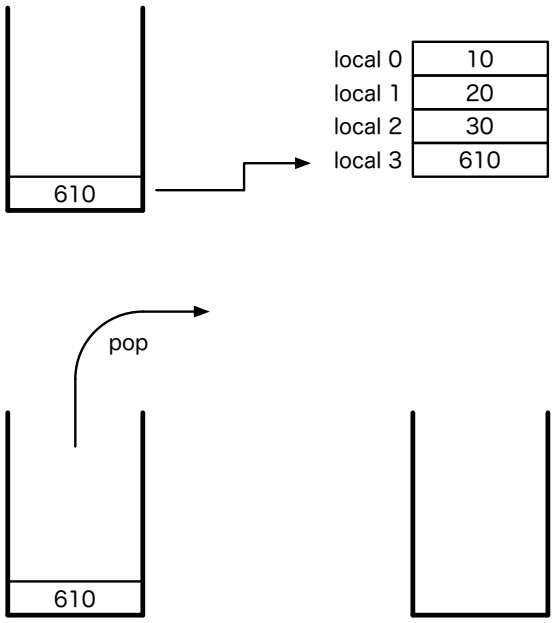



```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
   11 <11> push local 1
   12 <12> push local 2
   13 <A8> send *
   14 <A0> send +
★  15 <4F> store local 3; pop
  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

< 4 F >

```
| 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
```

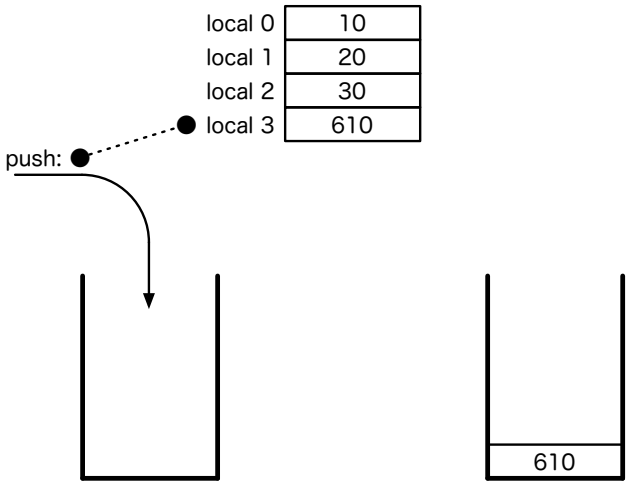


```
    1 <D8 0A> push 10
    3 <4C> store local 0; pop
    4 <D8 14> push 20
    6 <4D> store local 1; pop
    7 <D8 1E> push 30
    9 <4E> store local 2; pop
   10 <10> push local 0
   11 <11> push local 1
   12 <12> push local 2
   13 <A8> send *
   14 <A0> send +
   15 <4F> store local 3; pop
★ 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

< 1 3 >

```
| 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
```

< 6 5 >

```
| 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
```