プログラミングI
数理物理，総合理学等向け

2018年9月24日

海谷 治彦

目次

- まえおき
- コンピュータプログラムの動作原理
- プログラムの開発手順
- Cygwinを用いた実際の開発

本授業の対象者

- 基本，数理物理学科の一年生.

- 情報科2010年度以前の入学者でかつプログラミング演習Iの単位未修得者
- 総理プロでプログラミング演習I(1コマもの)を受講する予定の学生
- プログラムの授業はコレだけの学生.
  - 生物や化学等.

他の授業を受けるべき学生

- 情報科学科の2011以降の学生
- プログラミングI演習(2コマのもの)を受講予定 or したいもの.

- I演習(2コマもの), 演習I(1コマもの)のどちらをとるべきかは, 教職等の関係を踏まえて, 各自, 調べてください.
- 演習受講予定でない総合理学の学生は, 講師側の指示で海谷 or 桑原先生の授業のどちらかをとってください.
  - 部屋のサイズの関係です.

対応する演習科目

- プログラミング演習I by 木下，武山，森本，韓先生
  - 火曜日 五限の一コマのみ
  - 「プログラミングI演習」(2コマもの，桑原先生)とは異なります．
- 同時に受講しなければならないわけではないが，
- なるだけ，演習I(I演習ではない)も同時に受講することをお勧めします．

- 「とりあえず授業科目(本科目)だけとって，演習はいいや．」という学生の判断も認めてはいる．
  - とりあえず，演習だけでいいやってのは？

評価

- 授業での演習の提出 + 期末テスト
  - 配分については口頭にて.

- 演習の提出:
  - 基本，出席点ですが，白紙や努力の跡が見られないものは未提出扱いします.
  - たとえ，完成しなくても，わかるとこまで，何かを書いてください.
- 期末テスト:
  - 普通にテストします.

本授業の目標･背景･顛末

- プログラミング言語 C の初歩的なプログラムを各受講生が読み書きできるようになること.
- C言語の背景
  - 代表的なプログラミング言語である.
    - およそ40年前に開発され，25年ほど前に今の形となった.
  - 他の数多くの実用的な言語の祖先とも言える.
    - C++, Java, Ruby, JavaScript, PHP, Perl, C#
  - 今日でも広く利用されている現役の言語である.
  - おそらく，今後の寿命も長いと思われる.
    - JavaやPHP等は数年後でも大きく変化していると思われる.

- C言語ができないのは情報系としてはモグリだ！

授業のやり方

- 講義60分くらい
  - 基本スライドで行い，ときどき教科書を参照する.
- 演習30分くらい
  - その場で出してください，簡単なので.

- 教科書
  - 参考書，辞書程度というスタンスです.
    - なんか本があったほうがいいので，どっちか買って下さい.

  - [レ] 新版C言語プログラミングレッスン入門編

  以下は参考書とします.

  - [明] 新・明解 C言語

さて本題

何故，プログラミングしないといけないのか？

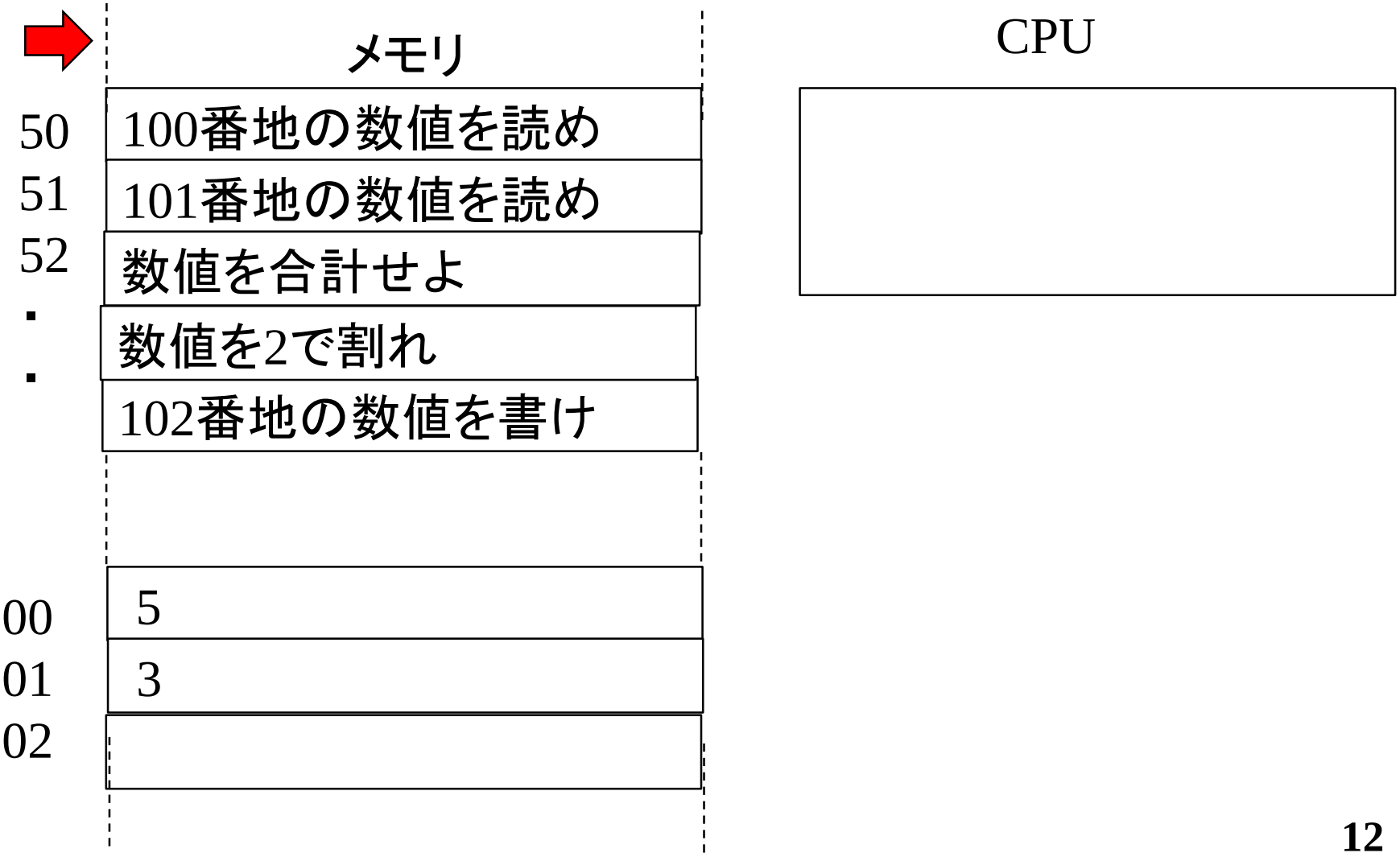
- 根本原因
  - コンピュータは人語を解さないため．
  - 我々人間もコンピュータの言葉(マシン語)を直接，読み書きできないため．
- プログラミング言語
  - コンピュータの言語と人間の言語の妥協点として開発された言語．
  - 人間も訓練すれば読み書きできる．
  - マシン語に自動翻訳できる．
    - 日本語や英語を直接にマシン語に翻訳するのは今の技術では無理．

例題によるコンピュータ動作の説明

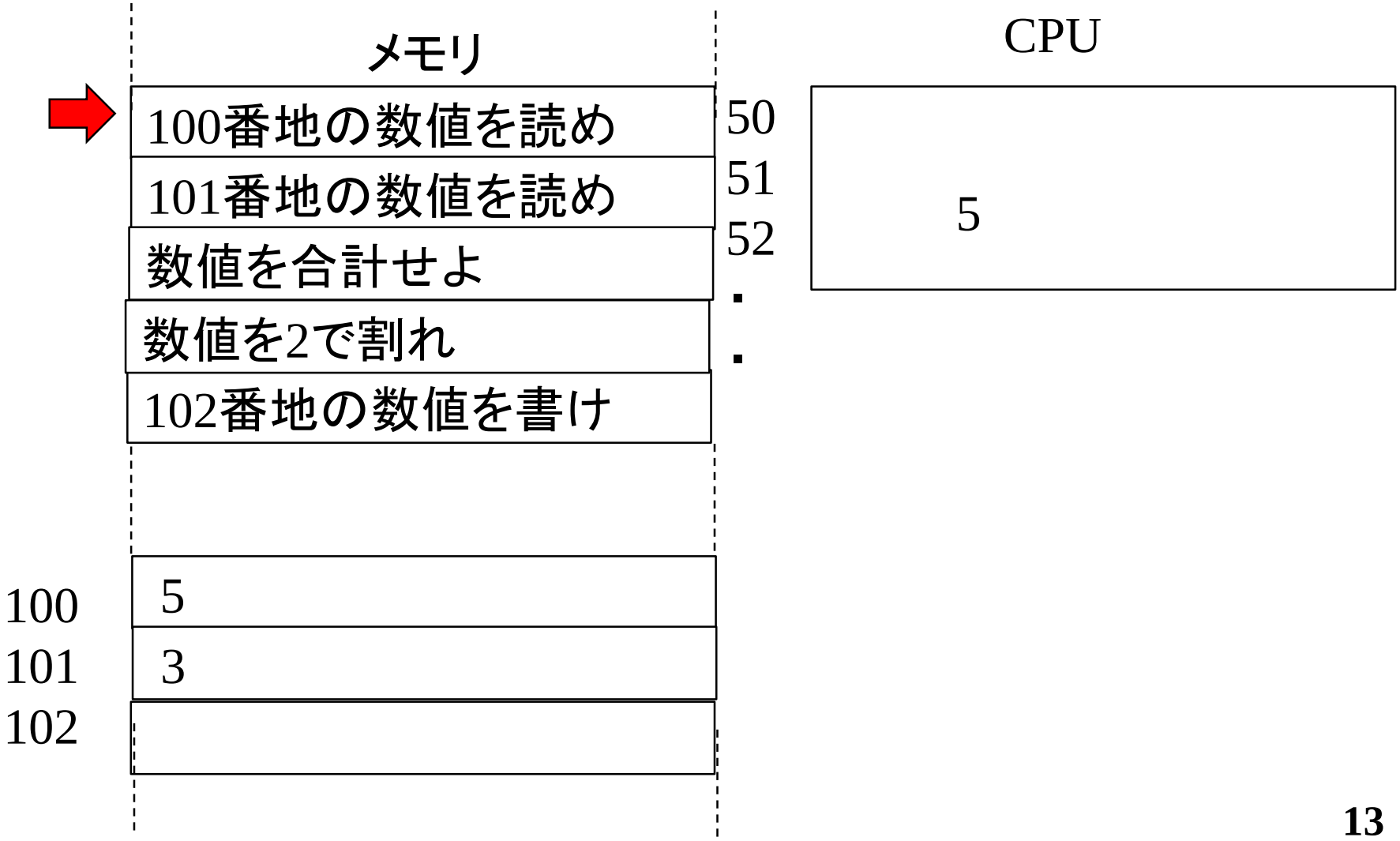
- コンピュータは，
  - 作業手順とデータを読み込み，
  - 手順に従い値を読み込んで，
  - 計算を遂行する．
- この「作業手順」がプログラムであるが，日本語で指示しても，当然，コンピュータは理解できない．

- 以降のページ群で説明のため，作業手順の各ステップを日本語で書いているが，
- 実際には，この手順を0と1の羅列であるマシン語で与えないといけないことも示す．

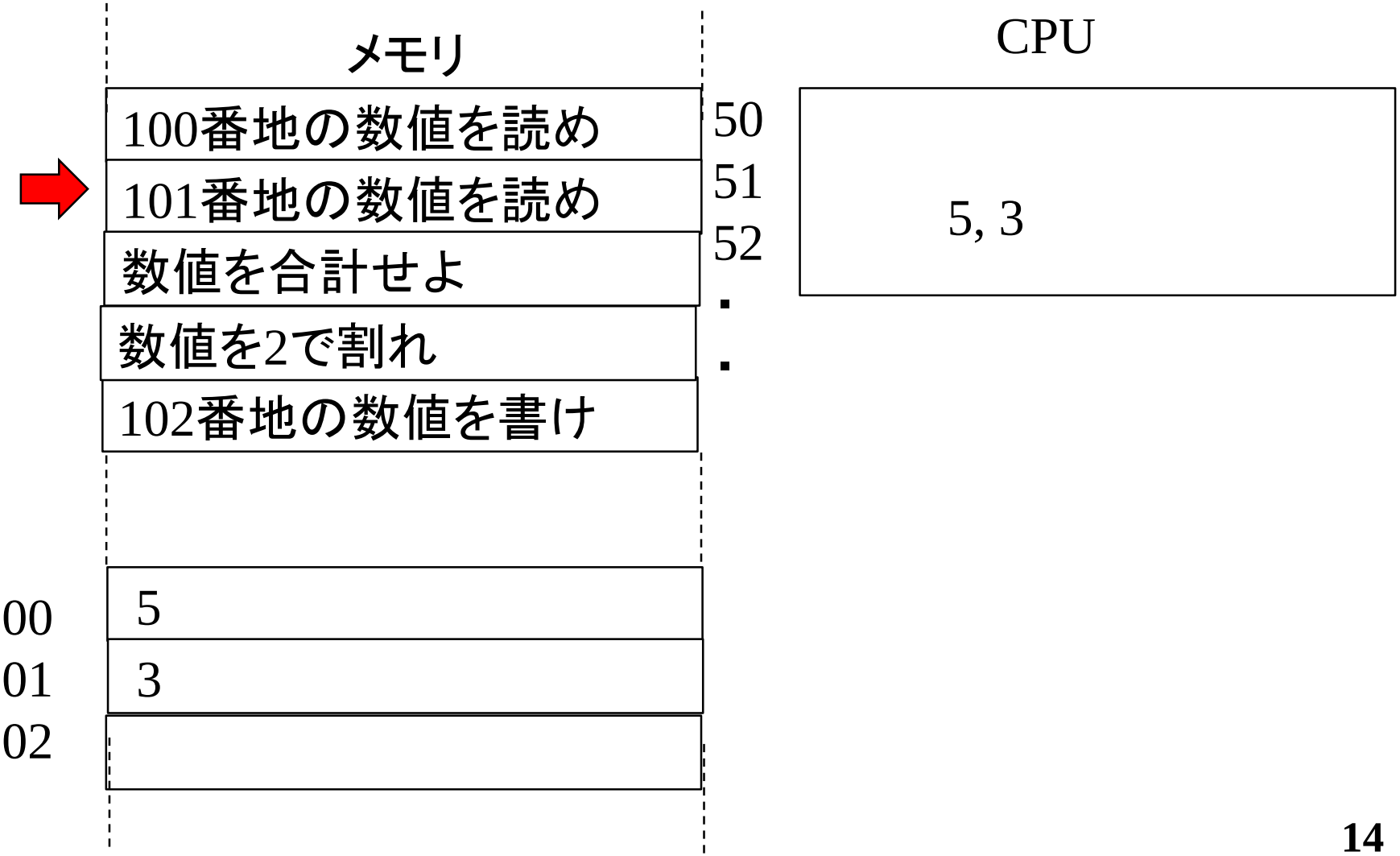
簡易な例題 ～ 二値の平均


メモリ
CPU
50
100番地の数値を読め
51
101番地の数値を読め
52
数値を合計せよ
数値を2で割れ
102番地の数値を書け
100
5
101
3
102

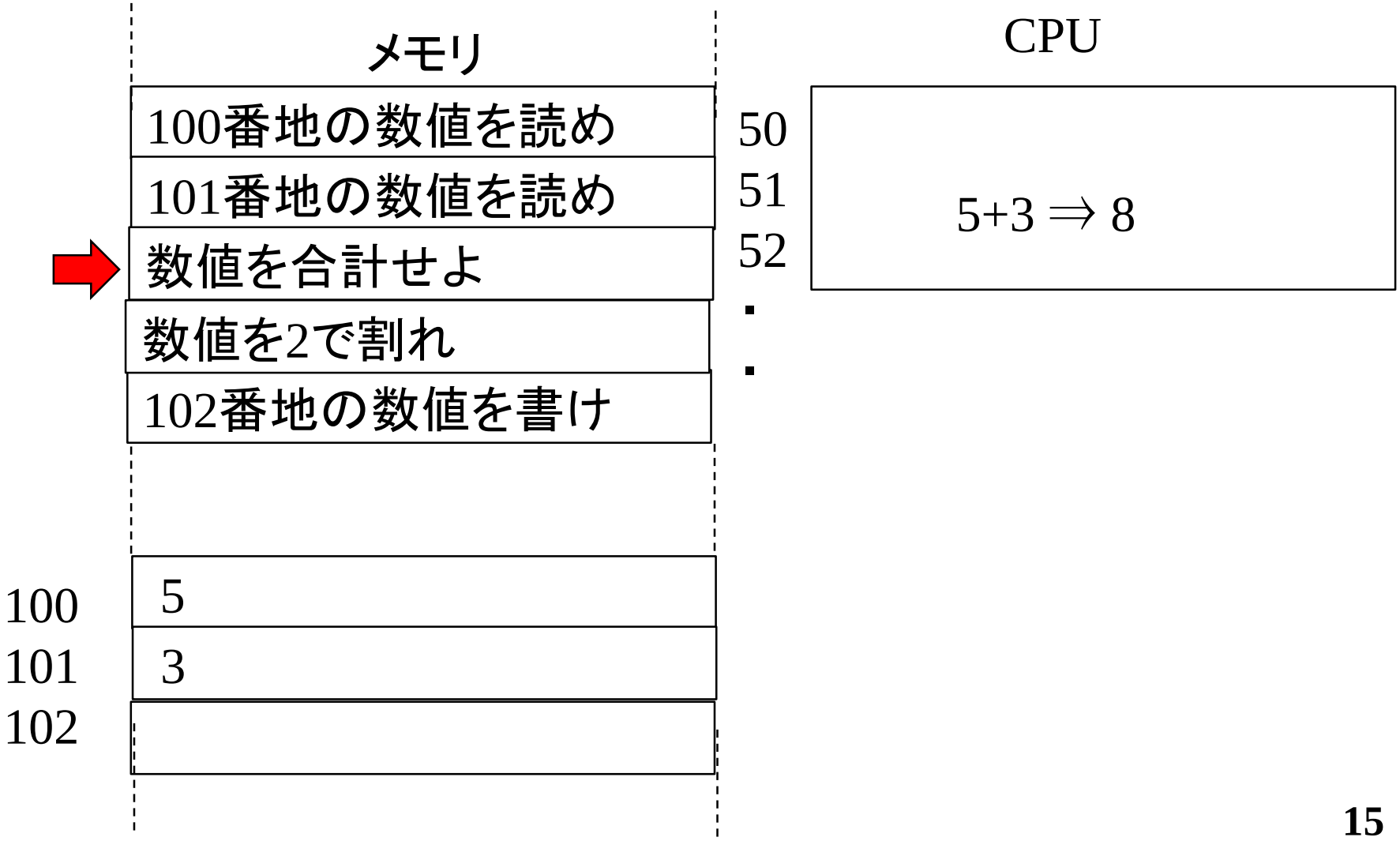
簡易な例題 ～ 二値の平均


メモリ
CPU
100番地の数値を読め
101番地の数値を読め
数値を合計せよ
数値を2で割れ
102番地の数値を書け
50
51
52
5
100
101
102
5
3

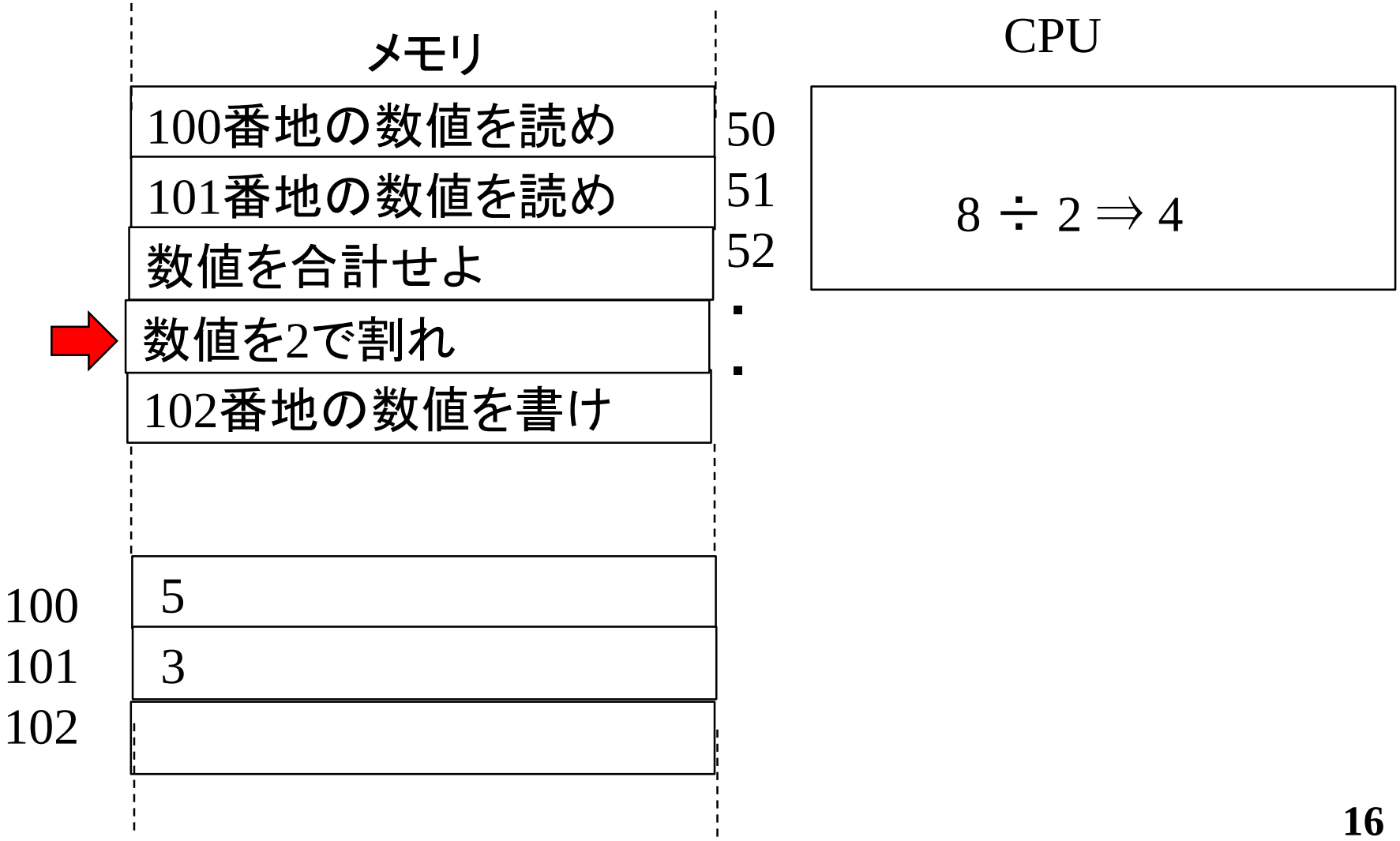
簡易な例題 ～ 二値の平均


メモリ
CPU
100番地の数値を読め
101番地の数値を読め
数値を合計せよ
数値を2で割れ
102番地の数値を書け
50
51
52
5, 3
100
101
102
5
3

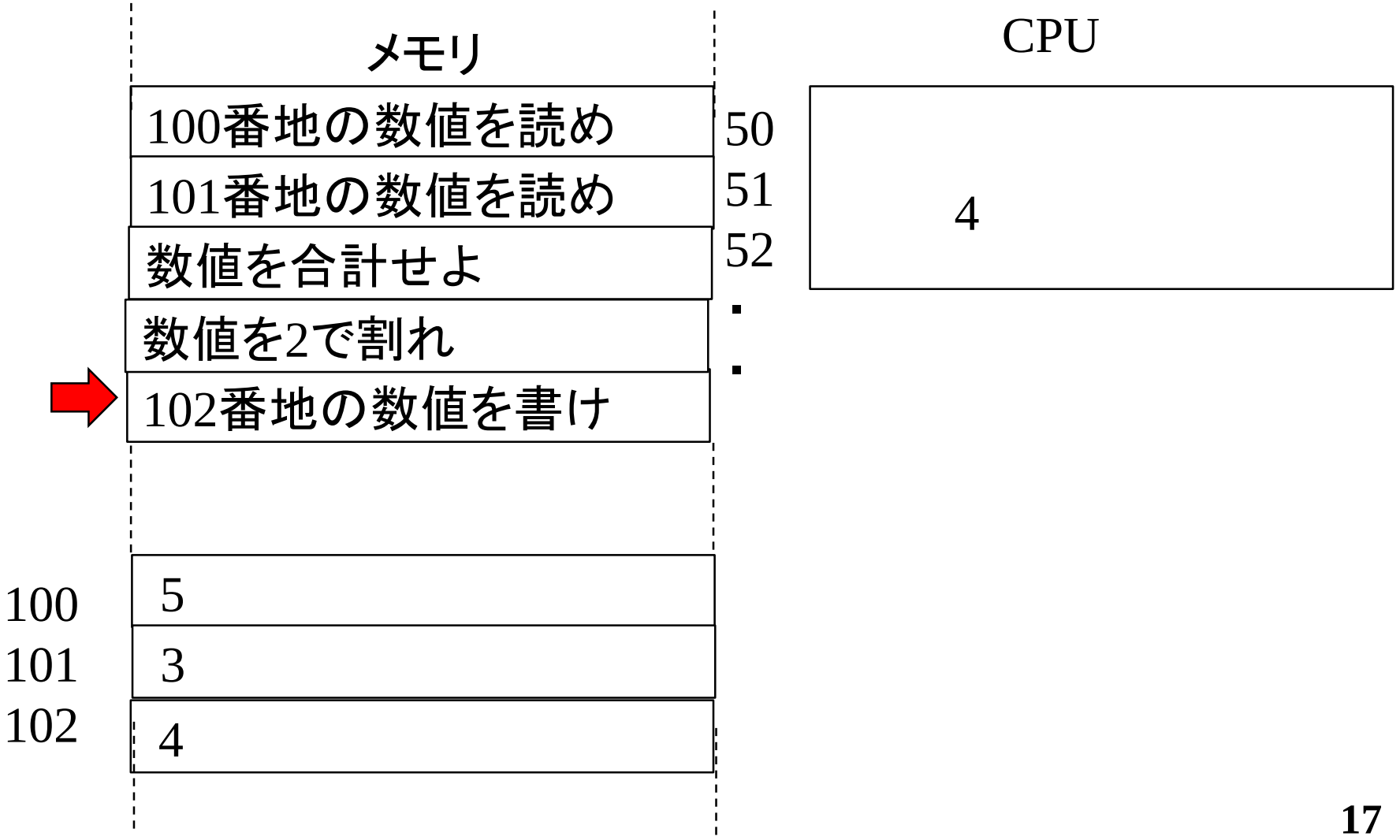
簡易な例題 ～ 二値の平均


メモリ
CPU
100番地の数値を読め
101番地の数値を読め
数値を合計せよ
数値を2で割れ
102番地の数値を書け
50
51
52
5+3 ⇒ 8
100
101
102
5
3

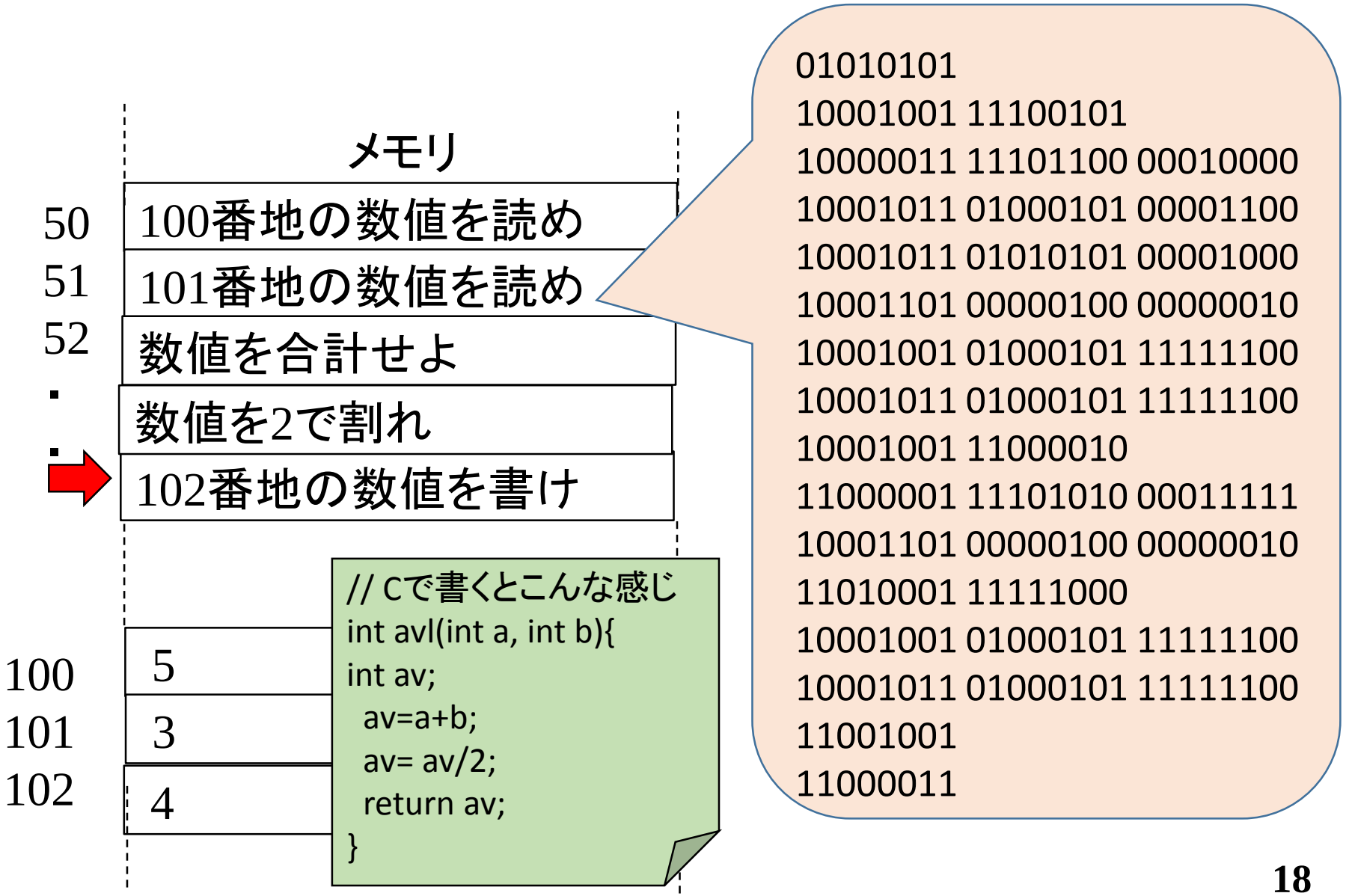
簡易な例題 ～ 二値の平均


メモリ
CPU
100番地の数値を読め
101番地の数値を読め
数値を合計せよ
数値を2で割れ
102番地の数値を書け
50
51
52
8 ÷ 2 ⇒ 4
100
101
102
5
3

簡易な例題 ～ 二値の平均

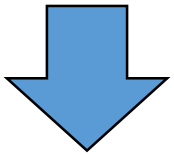

メモリ
CPU
100番地の数値を読め
101番地の数値を読め
数値を合計せよ
数値を2で割れ
102番地の数値を書け
50
51
52
4
100
101
102
5
3
4

プログラムの実際


メモリ
50
100番地の数値を読め
51
101番地の数値を読め
52
数値を合計せよ
数値を2で割れ
102番地の数値を書け
100
101
102
5
3
4
// Cで書くとこんな感じ
int avl(int a, int b){
int av;
av=a+b;
av= av/2;
return av;
}
01010101
10001001 11100101
10000011 11101100 00010000
10001011 01000101 00001100
10001011 01010101 00001000
10001101 00000100 00000010
10001001 01000101 11111100
10001011 01000101 11111100
10001001 11000010
11000001 11101010 00011111
10001101 00000100 00000010
11010001 11111000
10001001 01000101 11111100
10001011 01000101 11111100
11001001
11000011

プログラミング言語への要件

- 人間が読み書きできる程度に意味あるフレーズじゃないと困る.
  - 英語や日本語に近い表記だとうれしい.
- コンピュータへの命令に変換できる程度にあいまい性が無いものでないと困る.
  - コンピュータは「空気」よめない.「ヤバイ」の一言で全てやりすごせない.
- メモリとCPUを用いた現代のコンピュータ(ノイマン型コンピュータ)の構造を想定したものだとうれしい.
  - 値の読み書き
  - 基本，並んでる順番で命令を実行する
  - 条件によって命令を取捨選択する
  - 繰り返しを行う 等

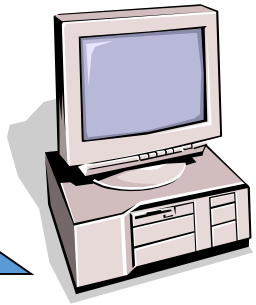


結果として，教科書にあるようなC言語になりました.

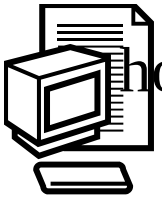
Cプログラムの開発の流れ

プログラミング
手作業
エディタを使用

コンパイル
自動変換
コンパイラを使用



実行
生成された
実行ファイルを使用



ソースプログラム
(ソースコード
hoge.c 等)

実行ファイル
(ロードモジュール
a.exe 等)

本授業での実習環境

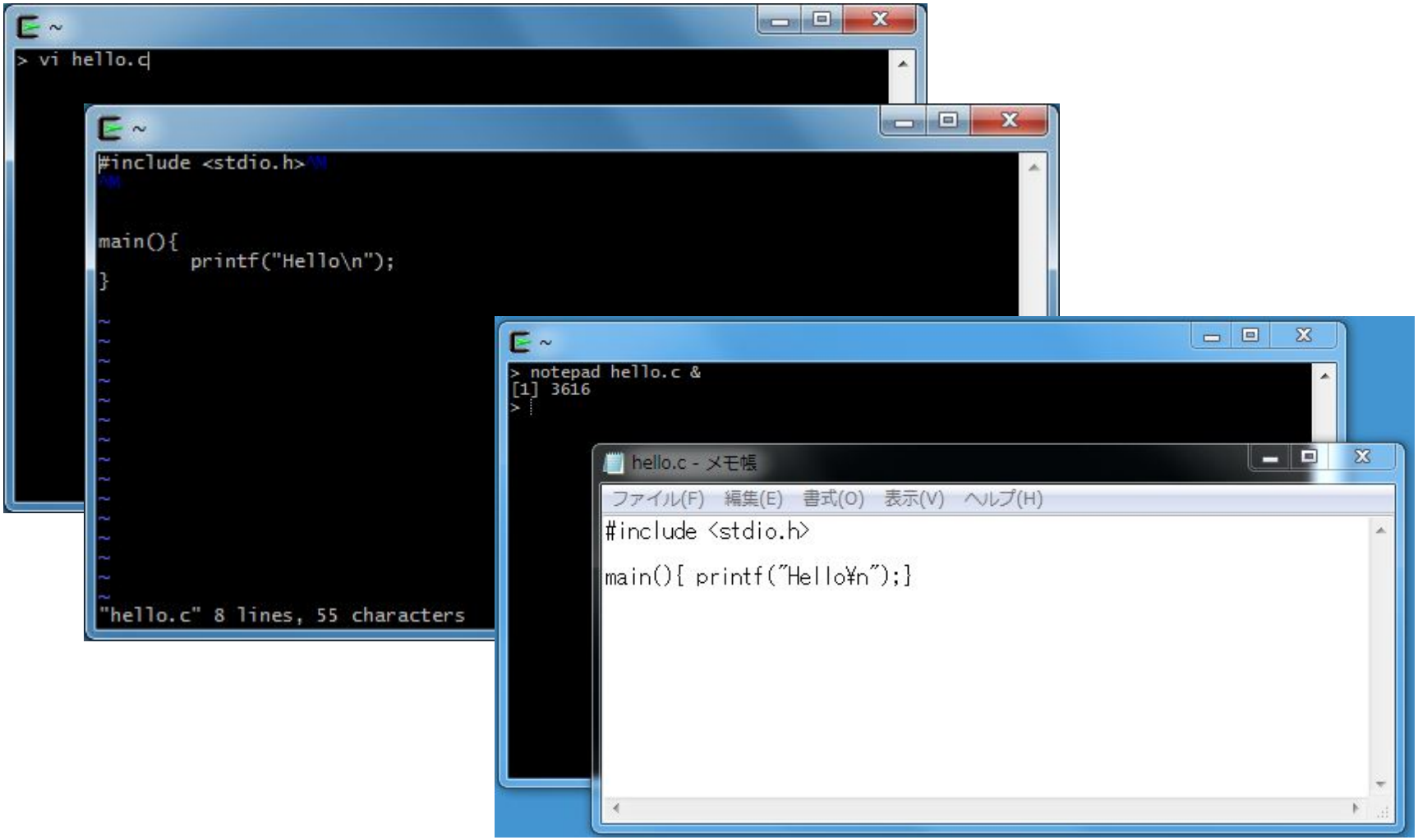
- 本端末室のPCを使います.
- Cygwinと呼ばれるWndows上でUNIXを模倣する環境を用います.
  - 教科書[レ]の「UNIX」と書いてある表記のほうを見てください.
- 自分が作成したプログラム等はバックアップしてくださいね.
  - 誤って消したりすると単位的に死にますし.
  - 基本，クラウド(Dropbox等)やUSBメモリ等にコピーを残す.

テキストエディタについて

- ソースプログラムを書くためのワープロ.

- UNIXでは vi (実際にはvimと呼ばれるviクローン)というエディタが一般的ですが，とても癖が強いです.
- Cygwinではnotepad(Windowsのメモ帳)も利用できるので，面倒な方はnotepadの利用で結構です.
- Cygwinでも emacs はありますが，なぜか，非常に重いです.

エディタ色々


~
> vi hello.c
~
#include <stdio.h>^M
^M
main(){
printf("Hello\n");
}
"hello.c" 8 lines, 55 characters
~
> notepad hello.c &
[1] 3616
>
hello.c - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
#include <stdio.h>
main(){ printf("Hello¥n");}

notepadを使う場合の &

- メモ帳(notepad)を使う場合は，コマンドとファイル名の後に & をつけてください．
- & をつけないと，メモ帳を終了するまで，ターミナルを使うことができません．

日本語の扱いについて

- ファイル内において**日本語は使わない**でください.
  - Latin1の拡張部分も使わないで！
  - ドイツ語等にあるëみたいなの.

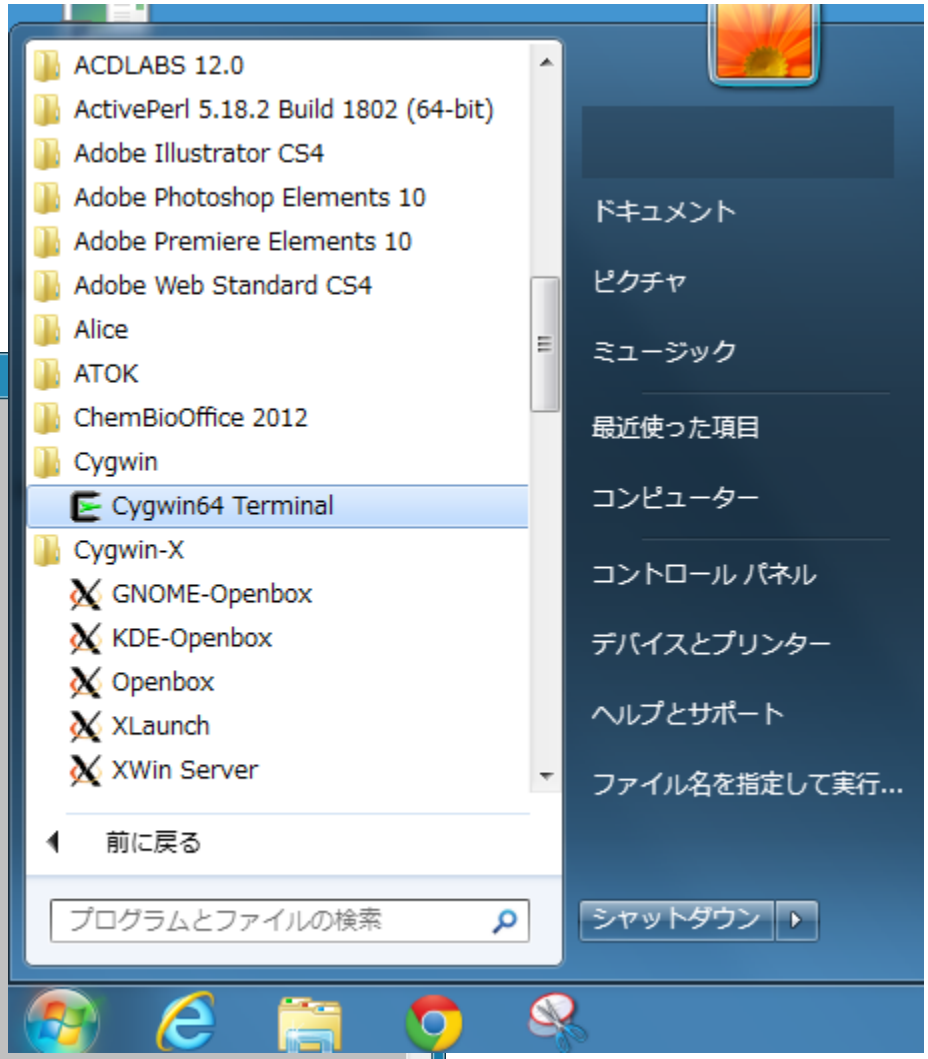
- ファイル名に日本語を使わないでください.
- ファイルの中身(ソースプログラム内)に記述しないでください.

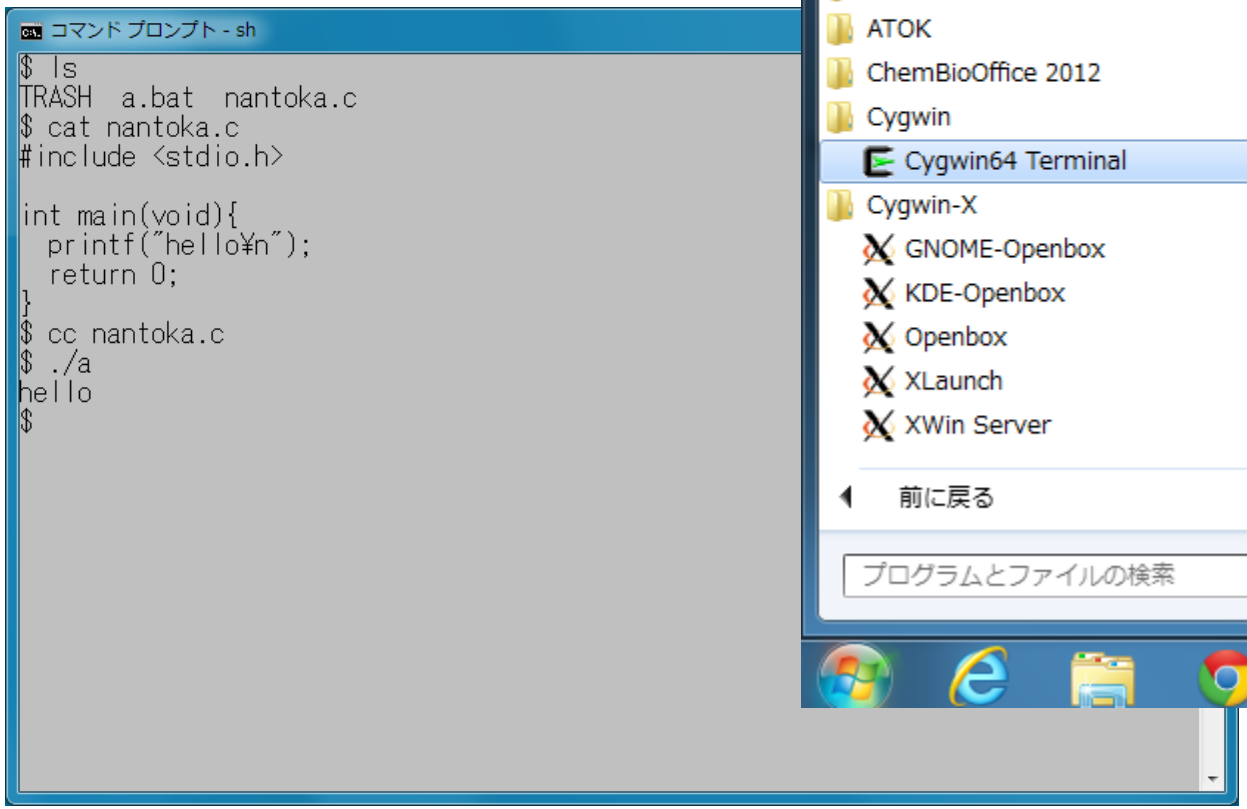
- 「日本語の空白文字」等，つまらないトラブルの原因になるので，本授業ではファイルでの日本語利用を禁止します.

- 質問等は日本語でも結構です.

Cygwinの起動

- スタートメニュー中に左図のようにあるはずです.
- 下図のような「端末」が表記されるはずです.



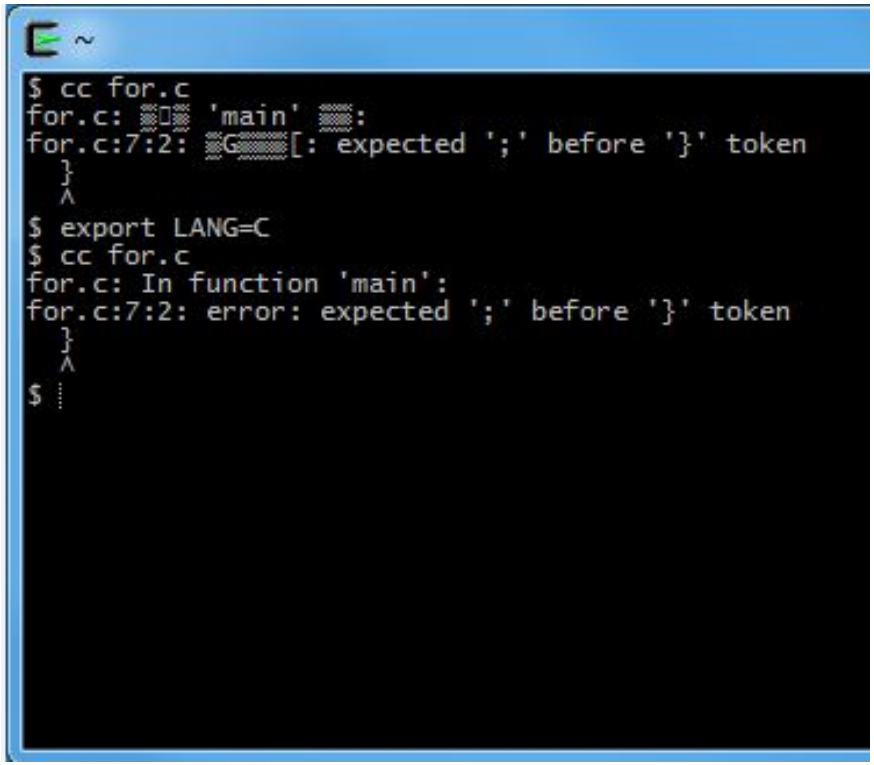


警告等が文字化けしたら

- コンパイラのエラー等の警告が左記の図のように，文字化けしたら，
- 以下のコマンドをうってください．

```
export LANG=C
```

- 警告は英語になりますが，文字化けは解消されます．



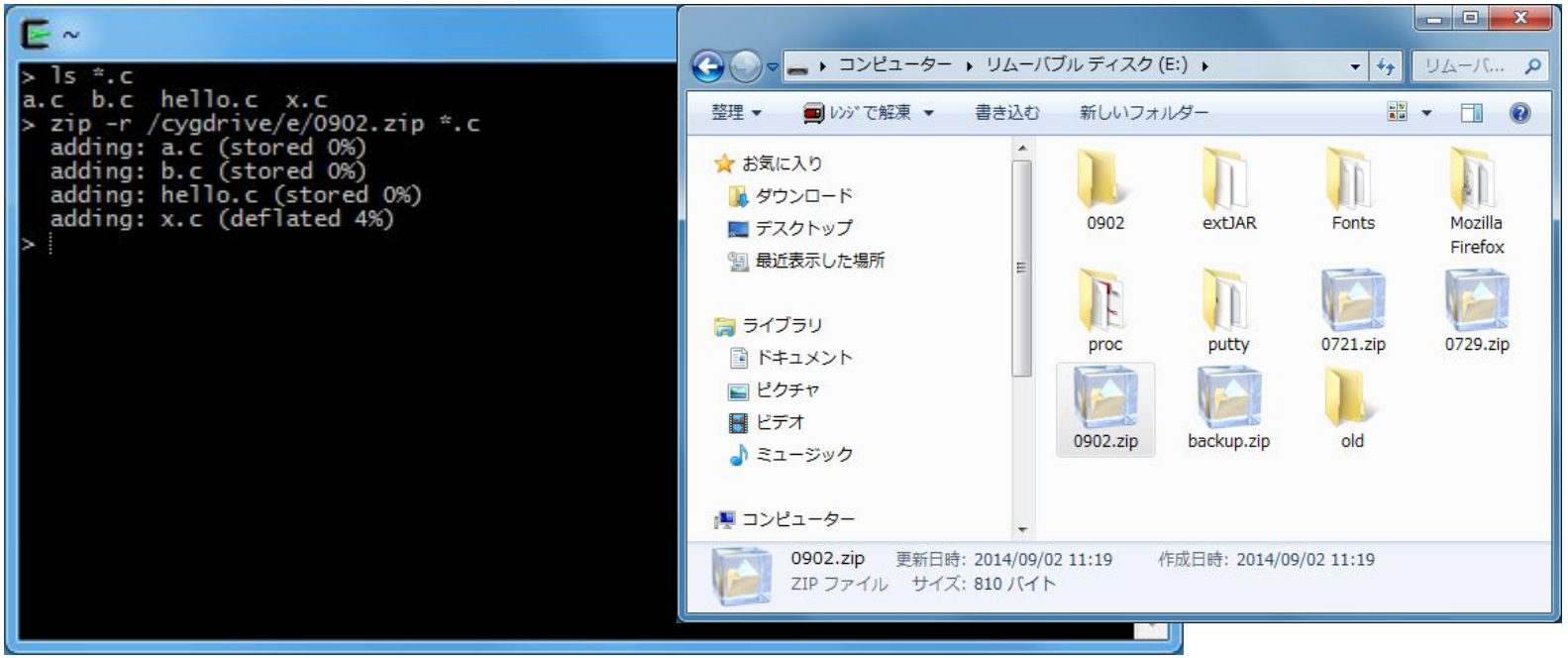
バグについて

- バグとはプログラムの誤りです.
- Defect とも呼ばれ，プログラムが意図した振る舞いをしない原因のことです.
- 以下の二種類のバグが典型的です.
  - 構文的なバグ
    - プログラムの文法が誤っているため，プログラムがコンパイルできない，もしくは，意図と異なる動作をすること.
    - 比較的，発見はたやすい．だって，動かないし.
  - 意味的なバグ
    - プログラムが意図した振る舞いと異なること.
    - 構文はあってるけど，結果がおかしいこと.
      - 数列の足し算をしてるつもりなのに，結果が合わない等
    - これを発見するのは容易ではない.

参考: バックアップについて

- USBメモリにその日の内容をバックアップするのがよいと思います. (dropbox等のネットストレージでもよいですが･･･)
- 以下の例では, USBメモリはEドライブにあるものとして, 0902.zip というファイルにその日作成した .c で終わるファイルを保存しています.

```
zip –r /cygdrive/e/0902.zip *.c
```



以上，次に続く