ソフトウェア概論 A/B

-- メモリモデル/printf --

数学科 栗野 俊一 / 渡辺 俊一

2013/12/20 ソフトウェア概論

伝言

私語は慎むように !!

- 色々なお知らせについて
  - 栗野の Web Page に注意する事

    http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino
- 廊下側の一列は遅刻者専用です(早く来た人は座らない)
- 講義開始前に済ませておく事
  - PC の電源を入れる
  - ネットワークに接続しておく事
  - 今日の資料に目を通しておく事
- 講義前の注意
  - 講義前は、栗野は準備で忙しいので TA を捕まえてください
- やる気のある方へ
  - 今日の資料は、すでに上っています
    - どんどん、先に進んでかまいません

今後の予定

- 今後の予定(後ろから)
  - 2014/01/24 (講議最終日)
    - 試験を行う
  - 2014/01/17 (講議最終日)
    - 模擬試験を行う
  - 2014/01/10 (試験前)
    - 講議最終日 (メモリモデルとポインター (2) / 落穂拾い)
  - 2014/01/03, 2014/12/27
    - 冬期休暇期間中 : この講議はない
  - 2014/12/20 (本日)
    - メモリモデルとポインター

前回 (2013/12/13) の復習

- 講議
  - 大域変数：関数(ブロック)の外で宣言された変数
    - 関数間で共有できる (<スコープ,エクステント> = <全域,永遠>)
    - cf. 局所(auto)変数：ブロックの中で宣言された変数 (<ブロック内,一時的>)
    - 引数によらない、関数間の値の共有に利用できる(便利だが、危険)
  - 静的変数：static 宣言付きの変数
    - 制限付き大域変数 (<ブロック(ファイル)内,永遠>)
    - スコープがある：大域変数に比べて安全
    - エクテントが永遠：(同じ関数の)複数の呼出し同士で値を共有できる(値の蓄積ができる)
- 演習
  - コマの設置判定とコマの自動反転
    - 仕様/設計/コーディング
  - コンピュータの自動対応

本日の予定

- 講義
  - メモリモデル
    - メモリモデルとは？
    - 変数とメモリ
  - 複合型
    - 配列と構造体
  - printf/scanf
- 演習
  - 課題の提出

前回 (2013/12/13) の課題

- 前回 (2013/12/13) の課題
  - なし

本日の課題 (2013/12/20)

- 本日 (2013/12/20) の課題
  - 課題 1:
    - ファイル名 : 20131220-1-QQQQ.c (QQQQ は学生番号)
    - 内容 : メモリ操作での和
    - ファイル形式 : テキストファイル(C 言語プログラムファイル)
  - 課題 2:
    - ファイル名 : 20131220-2-QQQQ.c (QQQQ は学生番号)
    - 内容 : アドレスを利用した間接参照
    - ファイル形式 : テキストファイル(C 言語プログラムファイル)

メモリ

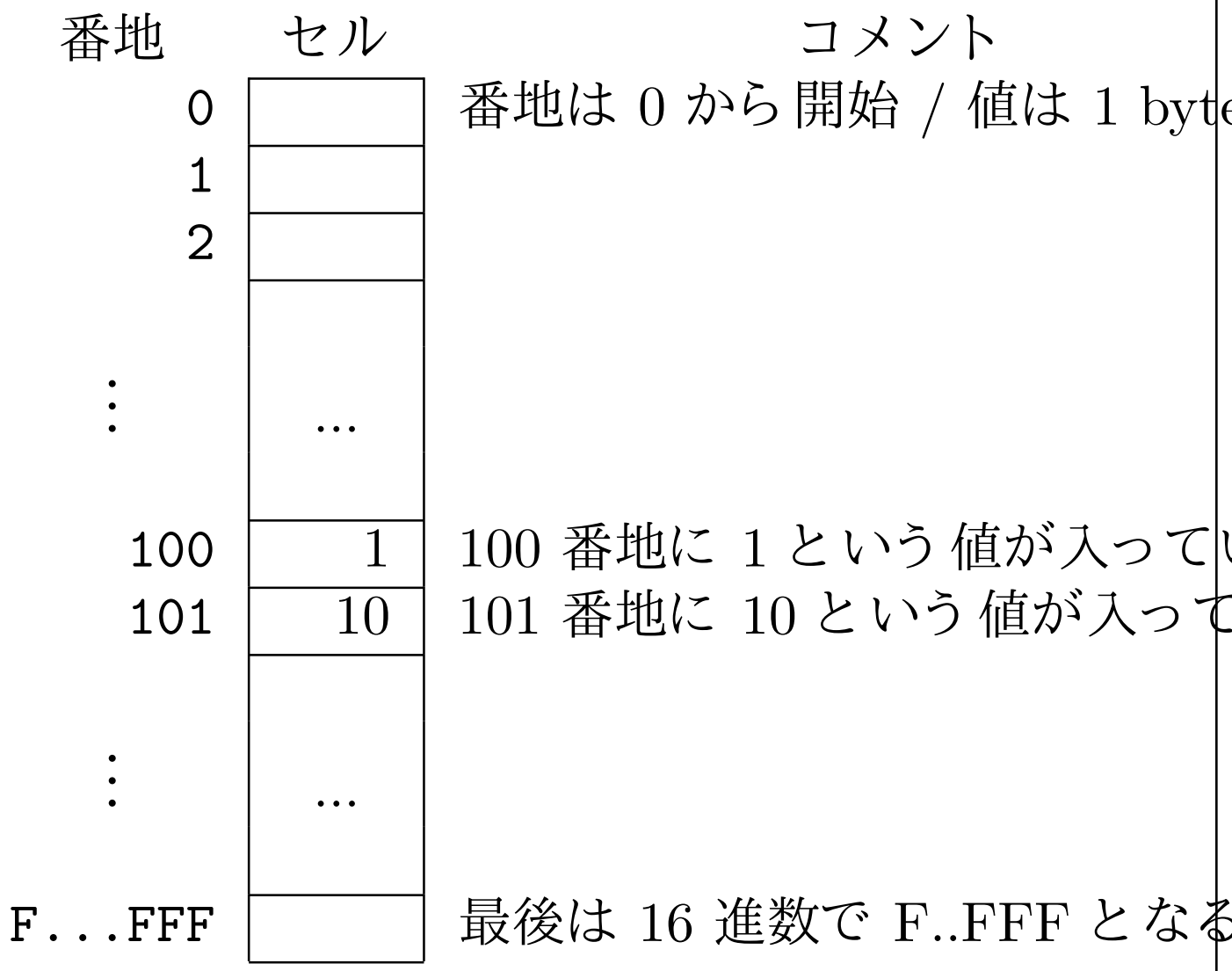
- (主)メモリ(記憶領域)とは？
  - 情報を記憶する小さなメモリセルの集まり
    - 一つのセルでは 0 ～ 255 の 256 (= 2^8 : byte) 種類の状態の内の一つ(情報)が記録されている
    - 個々のセルには(その位置を表す)番地(アドレス)がついている
  - [アナロジ] メモリ:ホテル, セル:部屋, アドレス:部屋番号, 情報:宿客
- メモリの操作：メモリセルの「記憶能力」(sample-001.c)
  - 情報の記録 (set_memory_value_at)
    - アドレスと記録する情報を指定して、そのアドレスのセルに情報を記録する
  - 情報の参照 (set_memory_value_at)
    - アドレスを指定して、そのアドレスのセルに記録された情報を取り出す

メモリセルの性質

- メモリセルの性質
  - 情報の参照は何度でもできる (sample-002.c)
    - 最後に記録した情報は何度でも取り出せる (cf. 不思議なポケット)
  - 記録できるのは一つだけ
    - 最後に記録したものだけが記録され、参照できる (sample-002.c)
  - メモリセルは独立 (sample-003.c)
    - 異るメモリセル(メモリセルの区別は番地で行う)は独立に振る舞う
  - メモリセルの記憶容量 (sample-004.c)
    - メモリセルが記録できるのは 0 ～ 255 (1 byte 内) の数値

メモリのイメージ

- メモリ
  - セルの並んだ物
    - セルのサイズは 1 byte
  - アドレス(番地)がついている
    - アドレスは 0..0 ～ F..F (16 進)
  - セルの機能 (変数と同じ)
    - 情報を記録できる
    - 情報を取り出せる



メモリモデル

- メモリモデル
  - C 言語の変数のモデルの一つで、「変数をメモリセルの組み合わせ」として理解する
    - C 言語の「変数の振舞い」を「考えるための仕組み(モデル)」
    - !! 「何かモデル」とは何かを理解するために利用可能な、「より簡単な仕組み」の事」
    - !! 「C 言語の変数」を「メモリモデル」を通じて理解する/ 簡単なの理解しやすい
- 実は..
  - 多くの場合、「C 言語の変数」は実際に「メモリセルの組み合わせ」になっている
    - 変数の性質(代入)はメモリの性質(記憶能力)から説明できる
- char 型変数とメモリモデル (sample-005.c)
  - char 型変数は、一つのメモリセルだと考える事ができる
    - char 型変数は address を持つ
  - char 型変数をメモリセルと同様に扱う事ができる

メモリモデルと配列

- 文字列とメモリモデル (sample-006.c)
  - 文字列は、文字の並び
    - 文字は char 型変数で記録できるので、文字列は char 型変数の並び
- 文字変数の並びと文字列 (sample-007.c)
  - アドレスがわかれば、変数の内容をアドレス経由で操作できる
- 配列宣言 (sample-008.c)
  - 配列とは
    - 「複数の変数の並び」の事 (個々の変数を「配列の要素」と呼ぶ)
  - 一次元の配列宣言 (sample-008.c)
    - 変数と同様に型名の後ろに「配列名[サイズ]」の形で宣言
    - 「サイズ」の個数だけの変数が宣言される。
    - 配列の要素は「配列名[0] ～ 配列名[サイズ-1]」という「名前」になる
    - 添字：「[」と「]」の間には整数値が指定でき、配列の何番目の要素かを表す

文字列と文字型の一次元配列の関係

- 文字列と文字型の一次元配列の関係
  - C 言語の文字列
    - 文字の並んだもの (文字コードが連続に記録されている)
  - C 言語の文字型変数
    - 文字コードを一つだけ記憶できる
  - C 言語の文字型の一次元配列
    - 複数の文字型変数が並んだもの
  - C 言語の文字列型の一次元の配列で文字列を記憶する事ができる
- 文字列の一部の操作方法 (sample-009.c)
  - 文字配列の要素を変更すればよい
- 文字列を利用した文字配列の初期化 (sample-010.c)
  - 文字配列の要素を文字列を利用して初期化できる

配列の添字, 間接(参照)演算子,アドレス演算子

- □ 文字列の操作 (復習)
  - ○「*」:間接(参照)演算子：文字列の先頭の文字を取り出す
    - ▹ *"abc" == 'a'
  - ○「[]」:添字演算子：「[n]」で n (整数値)で「n+1番目の文字」意味する
    - ▹ "abc"[0] == 'a', "abc"[1] == 'b', ..
  - ○「*」と「[]」の関係；文字列[n] == *(文字列+n)
    - ▹ "abc"[0] == *("abc"+0) == *("abc") == *"abc" == 'A'
- □ アドレス演算子「&」
  - ○ アドレス演算子「&」は 間接演算子「*」の逆演算を行う
    - ▹ == "abc"
  - ○ 変数に関しては逆が成立する
    - ▹「*(&var)」==「var」
  - ○ アドレス演算子「&」の正体
    - ▹ 変数に対応したメモリセルの「アドレス」を得る演算子