Programming II 0x09

ポインタ（2011.06.09）

塩澤秀和 http://vilab.org

ファイル入出力(復習)

プログラム中では
「¥」の代わりに「¥¥」

- ファイルとパス(path)
  - ファイル＝名前をつけて保存したデータ
  - 絶対パス: C:¥フォルダ名¥フォルダ名¥ファイル名
  - 相対パス: フォルダ名¥フォルダ名¥ファイル名

- ファイル入出力の手順(p.127)
  (1) ファイルを“オープン”する(開く) → fopen
  (2) ファイルに対して読み書きする → fprintf / fscanf
  (3) ファイルを“クローズ”する(閉じる) → fclose

- ファイルポインタ(p.128)
  - オープンしたファイルの現在状況を保持する変数

```
FILE * fp;
```

ファイルのオープン（復習）

- fopen関数（p.129）
  - ファイルを開く（オープンする）

    fp = fopen("ファイル名", "モード");
  - 戻り値はファイルポインタ（失敗ならNULL）

教科書の説明は
ちょっと変…

よく使うモード	説明
`"r"`	すでに存在するファイルを、読み込み専用で開く
`"w"`	空の新しいファイルを作り、書き出し専用で開く
`"r+"`	すでに存在するファイルを、読み書き両用で開く
`"w+"`	空の新しいファイルを作り、読み書き両用で開く

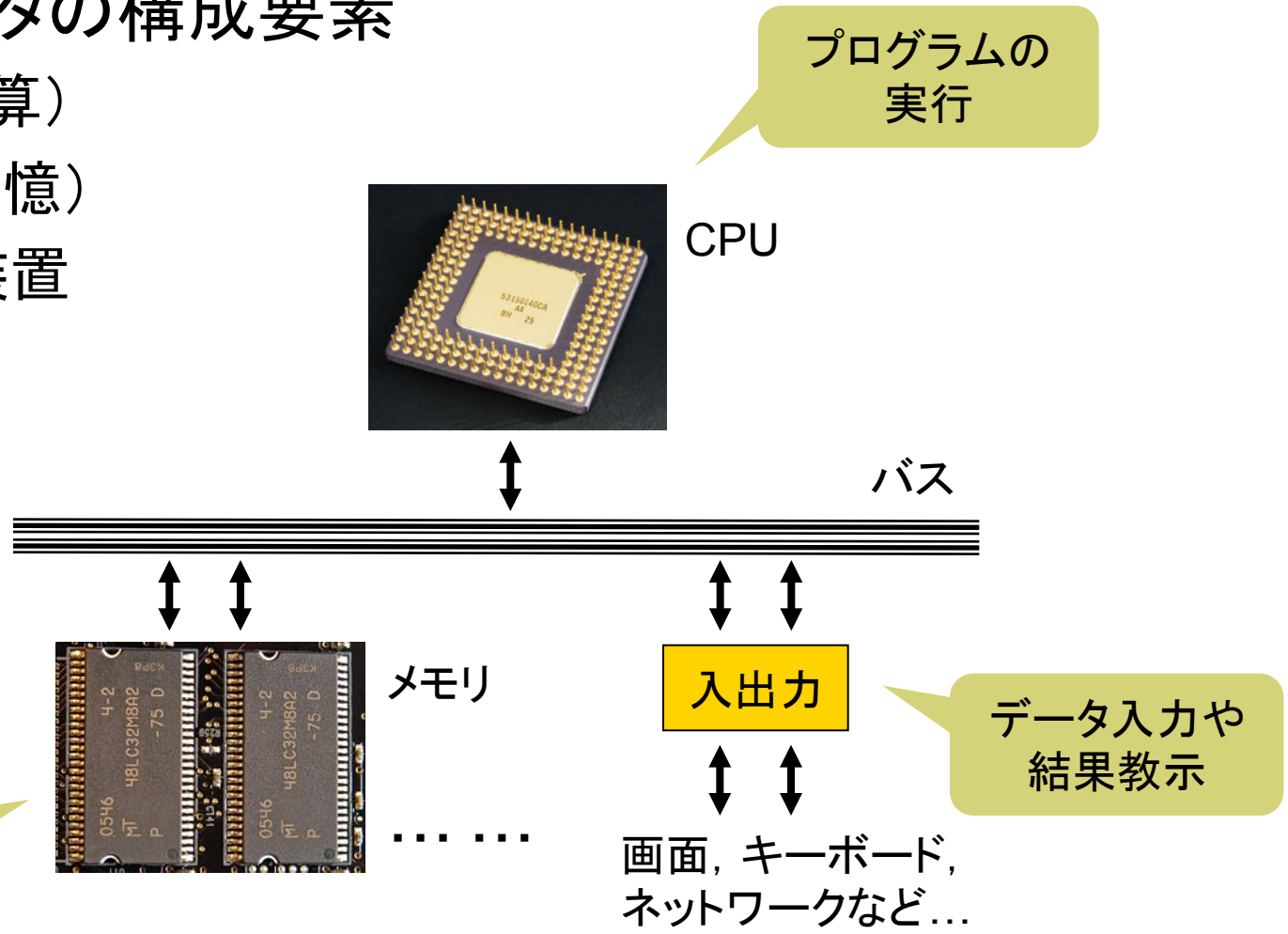
- エラー処理のすすめ
  - エラーを放置すれば、大切なデータが消える可能性も！
  - 単純なエラー処理 ⇒ exit 関数（<stdlib.h>）で強制終了

ファイルの読み書き（復習）

- fprintf関数（p.130）
  - ファイルにデータを書き出す（printfと同じ書式）
  - fprintf(fp, "書式文字列", 変数…)
- fscanf関数（p.131）
  - ファイルからデータを読み込む（scanfと同じ書式）
  - fscanf(fp, "書式文字列", &変数…)
  - 戻り値（int）は、正しく読み込めた項目数
- fclose関数（p.131）
  - 最後にファイルを閉じる（クローズする）
  - fclose(fp); ⇒ 戻り値はない

コンピュータのしくみ

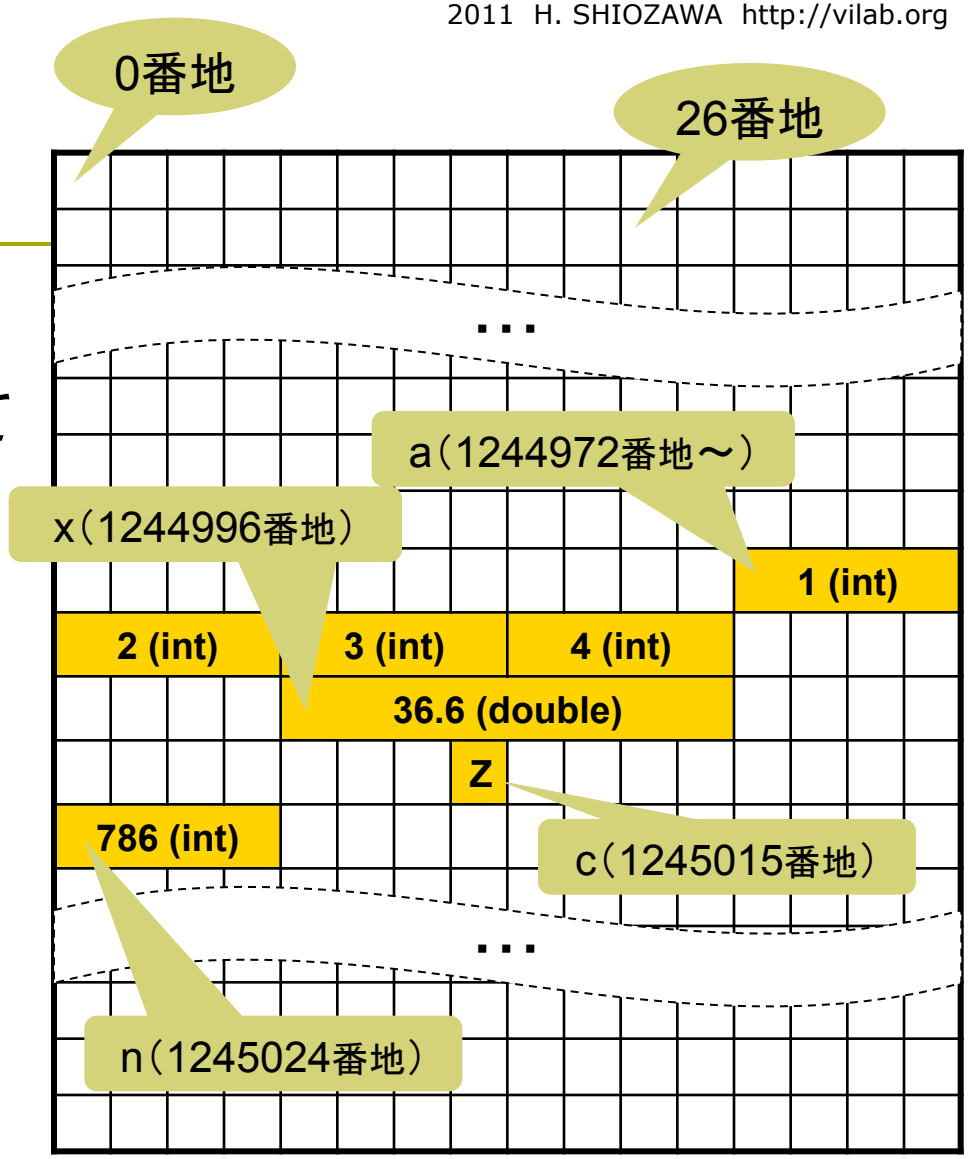
- コンピュータの構成要素
  - CPU（演算）
  - メモリ（記憶）
  - 入出力装置



メモリ空間

- メモリアドレス（番地）
  - コンピュータはどうやって情報を記憶するのか？
  - メモリには通し番号の“番地”がついている

```
int n = 786;
char c = 'Z';
double x = 36.6;
int a[4] = {1,2,3,4};
```



メモリ空間のイメージ

- データのサイズ（p.22）
  - データの種類によってサイズが決まっている
  - char 8ビット＝1バイト　int 32ビット＝4バイト（32ビット機）
  - float 32ビット＝4バイト　double 64ビット＝8バイト

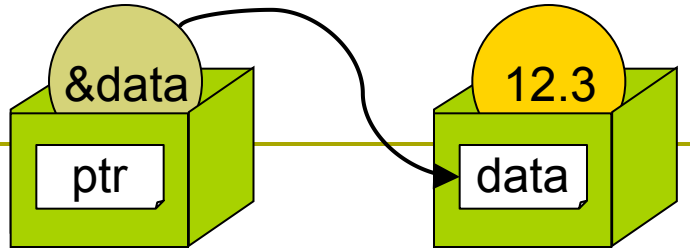
& 演算子

- &演算子（アドレス演算子）（p.147）
  - 変数の格納場所のアドレス（番地）を求める
    【使い方】 &変数名
  - アドレス（ポインタ）を表示する例
    printf("%p", &x); ← %pはポインタのための書式
    printf("%d", (int) &x); ← 無理やり10進数で表示する例

scanf で
おなじみの&

- sizeof 演算子（p.154）
  - 変数やデータ型のサイズ（バイト数）を求める
    sizeof(変数名) 例：sizeof(x)
    sizeof(データ型名) 例：sizeof(int)
  - サイズを表示する例：printf("%d", (int) sizeof(x));

ポインタ変数



- ポインタ(pointer)とは
  - データの格納された場所を示す値＝アドレス
  - または、そのアドレスを格納するための変数

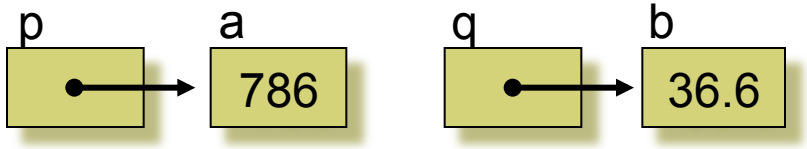
- ポインタ変数の定義
  - 定義(宣言):「型名 ＊変数名;」
  - 必ずメモリの中のデータの型に対応したポインタ型を使うこと

メモリの中のデータの型	対応するポインタの定義例
int	int *ptr;
double	double *ptr;
char	char *ptr;

- 代入と初期化
  - ptr = &data; ← ポインタptrに変数dataのアドレスを代入
  - int *p = &n; ← ポインタpの値をnのアドレスで初期化

変数とポインタ

アドレス（番地）	メモリ内容	変数名
0012FF60 (1245024)	786	a
0012FF50 (1245008)	36.6	b
0012FF44 (1244996)	0012FF60	p
0012FF38 (1244984)	0012FF50	q




```
#include <stdio.h>

int main(void)
{
  int a = 786;
  double b = 36.6;

  int *p;      /* intへのポインタ     */
  double *q;  /* doubleへのポインタ*/

  p = &a;  /* aのアドレスをpに代入 */
  q = &b;  /* bのアドレスをqに代入 */

  printf("アドレス %p (%d): 中身 %d (a)\n", &a, (int)&a, a);
  printf("アドレス %p (%d): 中身 %f (b)\n", &b, (int)&b, b);
  printf("アドレス %p (%d): 中身 %p (p)\n", &p, (int)&p, p);
  printf("アドレス %p (%d): 中身 %p (q)\n", &q, (int)&q, q);
  return 0;
}
```

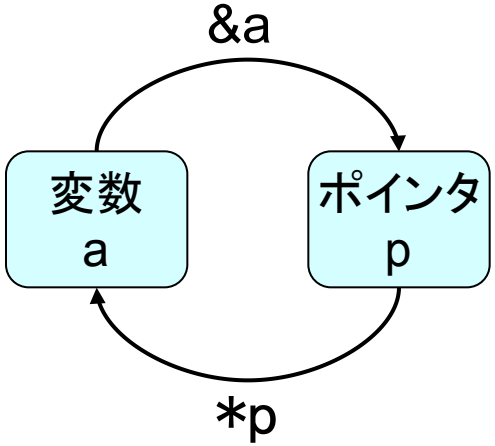
＊演算子

- ＊演算子（間接演算子）（p.147）
  - ポインタが指す先のメモリを参照する

    x = *p; ⇒ メモリのp番地のデータを、変数xに代入する

    *p = 5; ⇒ メモリのp番地に、データ「5」を代入する
  - つまり… p = &a とすると、*p が変数 a の“別名”になる

- 「&」と「＊」は逆の関係
  - &: 番地を調べる　＊: 番地をたどる
  - *(&a) ＝ *p ＝ a　&(*p) ＝ &a ＝ p



- ポインタ使用上の注意
  - **指す先がないポインタを使わないように注意すること！**

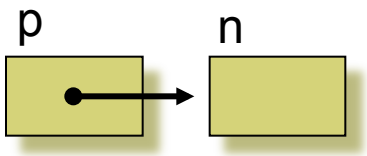
ポインタの使用例

```
#include <stdio.h>

int main(void)
{
  int n, m;
  int *p;

  printf("int? ");
  scanf("%d", &n);

  /* p が n を指すようにする     */
  /* 以後 *p が n の別名になる */
  p = &n;
```



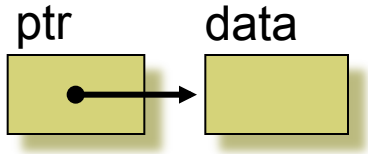

```
  m = *p;
  printf("m=%d\n", m);
  return 0;
}
```

```
#include <stdio.h>

int main(void)
{
  double data, in;
  double *ptr;

  printf("double? ");
  scanf("%lf", &in);

  /* ptr が data を指すようにする     */
  /* 以後 *ptr が data の別名になる */
  ptr = &data;
```




```
  *ptr = in;
  printf("data=%f\n", data);
  return 0;
}
```

演習問題

9a. int型の変数 i とchar型の変数 c を定義し、i と c のアドレスとサイズを表示するプログラムを作成しなさい。

9b. int型の変数 n と、int型へのポインタ p を定義し、n にキーボードから整数を読み込んだ後、printf("%d", *p); という文で n の値を画面に表示するプログラムを作成しなさい。

9c. double型の変数 x, y と、それぞれを指すポインタ xp, yp を定義し、キーボードから x と y に数値を読み込んでから、*xp + *yp と *xp - *yp を表示するプログラムを作成しなさい。

9d. double型の変数 val を定義し、scanf("%lf", ptr); という文で val に値を読み込むプログラムを作成しなさい。

- 次回までの課題: 教科書p.146～163 を予習（入力＆実行）

&演算子と＊演算子

- &演算子（アドレス演算子）
  - 変数のあるアドレス（番地）を調べる
    p = &a; ← 変数aのアドレスをポインタpに代入する
  - アドレスを表示する方法
    printf("%p", &a); ← 標準規格
    printf("%d", (int) &a); ← 無理やり10進数で表示

- ＊演算子（間接演算子）
  - アドレスにあるデータを取り出す
    a = *p; ⇒ ポインタpの指す場所のデータを変数aに代入
  - アドレスにあるデータを書き換える
    *p = 5; ⇒ ポインタpの指す場所に120を書き込む