ソフトウェア２
第２回
（2012/12/13）

鶴岡 慶雅

連絡用ページ

- URL

  http://www.logos.t.u-tokyo.ac.jp/~tsuruoka/lecture/software2/

  ユーザ名 :　soft2

  パスワード :　ee2012

- 資料
  - 講義スライド（ppt, pdf）
  - サンプルプログラム

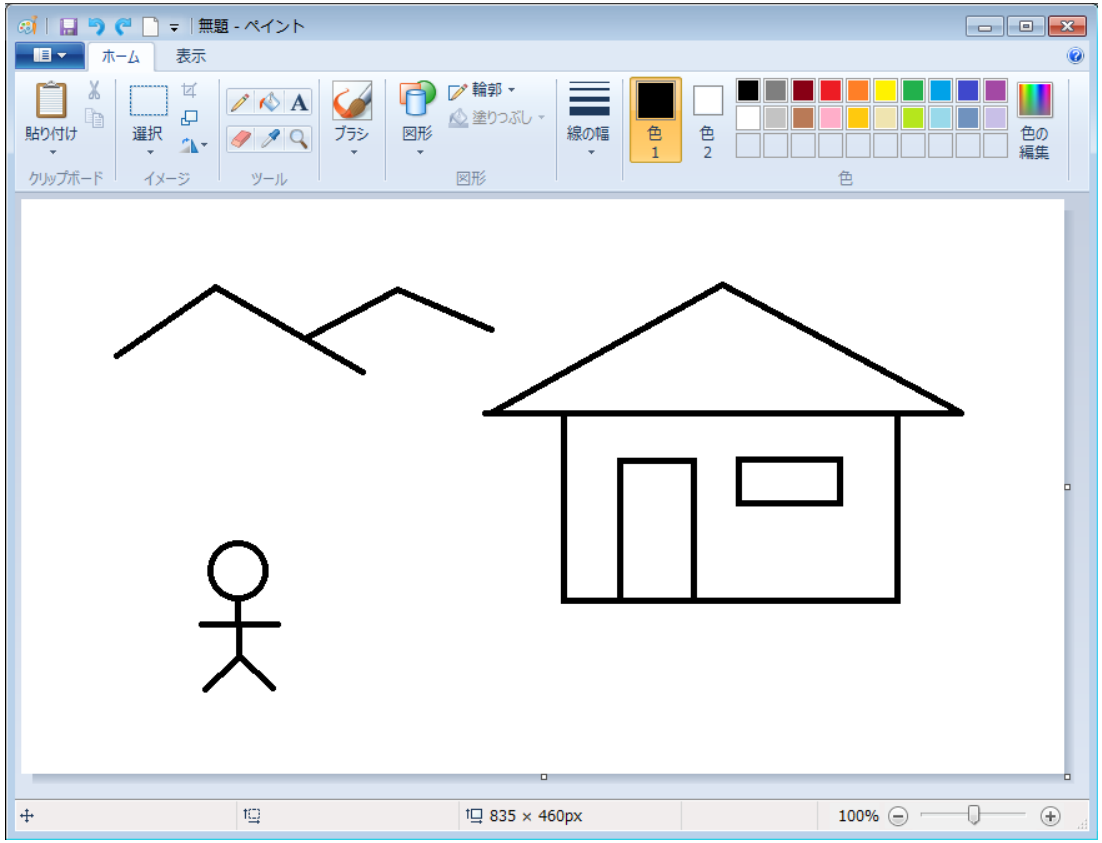
今日の内容

- ペイントソフトの作成

- C言語入門
  - メモリ管理・操作
    - malloc, free, memset, …
  - 文字列操作
    - strcpy, strtok, …
  - 数値計算
    - sin, cos, …

ペイントソフト

- コマンド入力方式で絵を描く
  - 描画機能
    - 線を描く
    - 長方形を描く
    - 円を描く
    - :
  - Undo
    - 直前のコマンドの取り消し
  - 履歴の保存



サンプルプログラム paint.c

- コンパイル＆実行

  % gcc paint.c

  % ./a.out

  0 > line 10 10 20 10 ← (10, 10) から (20, 10) まで線を引く

  1 > quit ← 終了

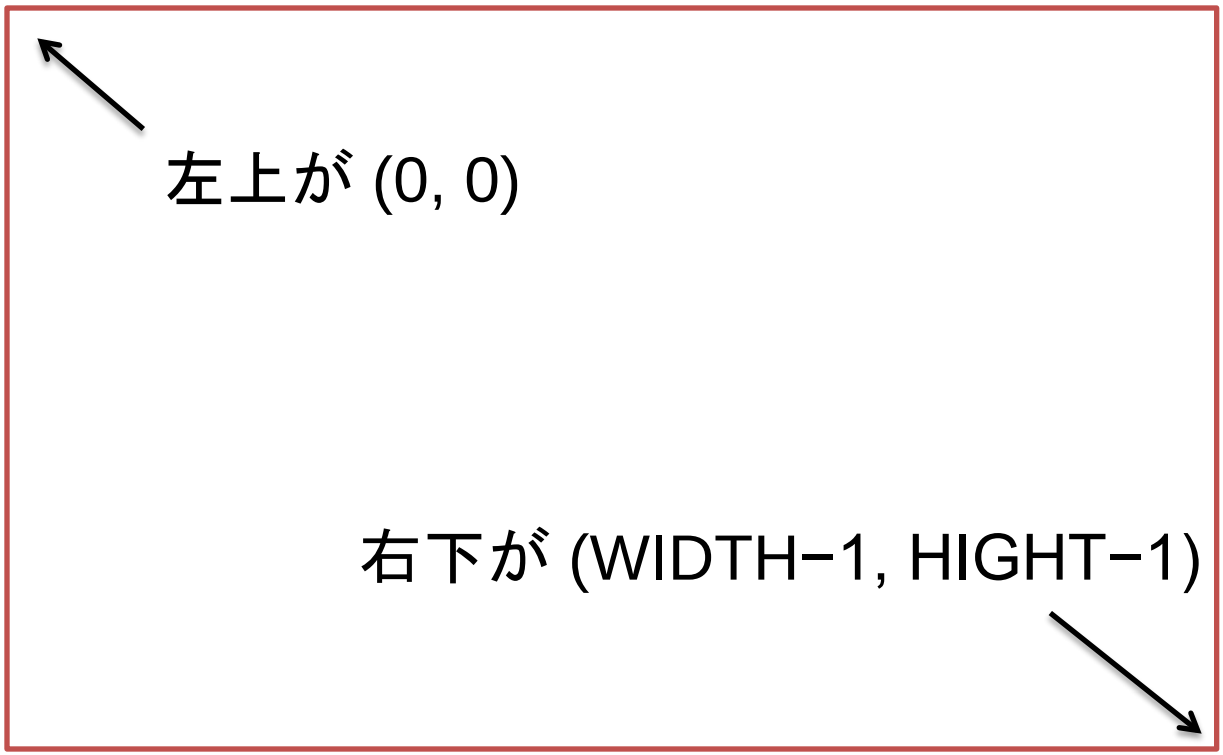
- ターミナルをもうひとつ開く（キャンバス用）

  % tail –f canvas.txt

  ターミナルのサイズをマウスで調整して ---------- が左上にくるように

キャンバス

```
char canvas[WIDTH][HIGHT]
```



※メモリ上では、ドットが上下方向に連続することになるが
canvas[x 座標][y 座標] としたかったので

paint.c 解説

- コマンドの履歴を保存
  - malloc関数でコマンド文字列の長さの分だけメモリを動的に確保
  - 文字列ポインタの配列 history[] にアドレスを保存
  - strcpy関数でコマンドの文字列をコピー

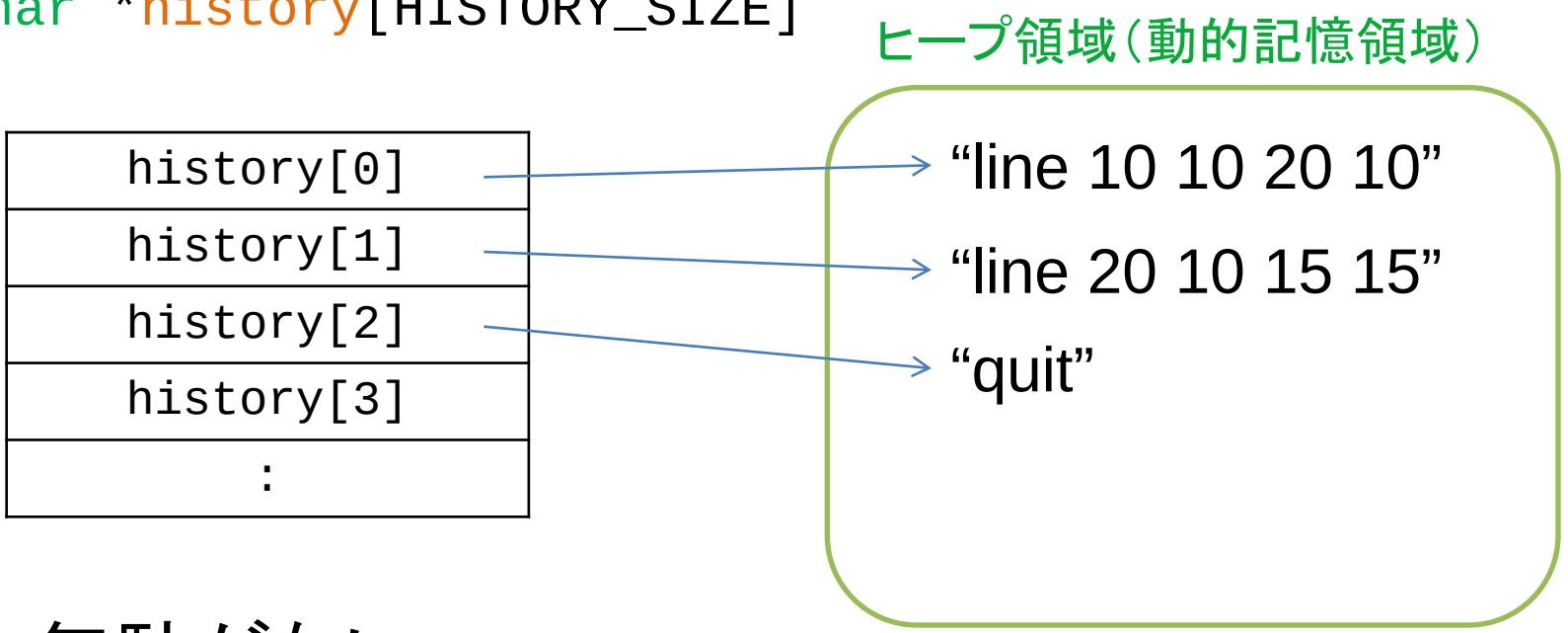
paint.c 94行目～

```
printf("%d > ", n);
fgets(buf, BUFSIZE, stdin);

history[n] = (char*)malloc(sizeof(char) * strlen(buf));
strcpy(history[n], buf);
```

文字列へのポインタの配列

```
char *history[HISTORY_SIZE]
```



- 無駄がない
  - 2次元配列で文字列を格納しようとすると？
- 使わなくなったメモリは解放する
  - free関数を使う
  - 解放し忘れ　→　メモリリーク

paint.c 解説

- コマンド文字列をトークンに分解
  - strtok関数で最初のトークンを取得
    - デリミタ(区切り文字列)は空白のみ
    - 文字列が破壊されるのでコピーしたものを渡す

paint.c 57行目～

```
int interpret_command(const char *command)
{
  char buf[BUFSIZE];
  strcpy(buf, command);

  char *s = strtok(buf, " ");
```

paint.c 解説

- 残りの文字列もトークンに分解
  - strtok関数で2つ目以降のトークンを取得
    - 第1引数をNULLにして呼び出せばよい
  - atoi関数で文字列を整数値に変換

paint.c 27行目～

```
void line_command()
{
  int x0, y0, x1, y1;
  x0 = atoi(strtok(NULL, " "));
  y0 = atoi(strtok(NULL, " "));
  x1 = atoi(strtok(NULL, " "));
  y1 = atoi(strtok(NULL, " "));

  draw_line(x0, y0, x1, y1);
}
```

paint.c 解説

- 線を引く
  - 始点と終点を10等分して点を打つといういい加減なアルゴリズム
  - 始点と終点が遠いと点線に
    - 改良してください（課題1）

paint.c 31行目～

```
void draw_line(int x0, int y0, int x1, int y1)
{
  int i;
  for (i = 0; i <= 10; i++) {
    int x = x0 + 0.1 * i * (x1 - x0);
    int y = y0 + 0.1 * i * (y1 - y0);
    canvas[x][y] = '#';
  }
}
```

paint.c 解説

- キャンバスの初期化
  - memset関数でメモリ領域を「空白」のASCIIコード（0x20）で埋める
    - canvasが配列の先頭アドレスを指すことに注意
    - sizeofで配列のサイズ（バイト数）が得られる
  - もちろんfor文で初期化しても構わない

paint.c 31行目～

```
int init_canvas()
{
  memset(canvas, ' ', sizeof(canvas));
}
```

課題（締め切り12/19）

1. paint.c の線を引くアルゴリズムを改良し、常に実線を引けるようにせよ
   - プログラムを添付すること（ファイル名は “paint1.c”）
   - paint.c を利用せず自分でゼロからプログラムを作成しても構わない

2. 長方形を描くコマンドと円を描くコマンドを追加せよ
   - コマンドの名前、引数の形式は任意
   - プログラムを添付すること（ファイル名は “paint2.c”）

3. Undo 機能を実装せよ
   - プログラムを添付すること（ファイル名は “paint3.c”）
   - ヒント
     - キャンバスを初期化して履歴のコマンドを順に適用すればよい
     - 不要になったコマンド文字列の領域は free する

4. ［発展課題］ 他に有用なコマンド（塗りつぶし、エフェクトをかける、コピー＆ペースト、BMP形式で保存、など）を追加せよ
   - プログラムを添付すること（ファイル名は “paint4.c”）
   - 追加したコマンドについて簡単に説明すること
   - それらを用いて作成した絵も添付すること（ファイル名は “picture.txt”）

課題の提出方法

- 宛先
  - software2@logos.t.u-tokyo.ac.jp
- Subject
  - 形式: SOFT-MM-DD-NNNNNNX
    - MM: 月
    - DD: 日（授業が行われた日）
    - NNNNNNX: 学籍番号
- 本文
  - 冒頭に学籍番号、氏名を明記

三角関数

```
double sin(double x);
double cos(double x);
double tan(double x);
```

- 三角関数の値を計算
  - 引数の単位はラジアン
- #include <math.h> が必要
- 逆三角関数を計算する関数もある
  - atan(),acos(),atan(),…