ビジュアライザの作り方

semiexp

ビジュアライザ

- 幾何系の問題が出た時，それをうまく描画できるとうれしいことがある
- IOI の過去問でビジュアライザ的なことが役に立つかもしれない問題の例：
  - IOI 2006 Joining Points
  - IOI 2007 Flood
  - IOI 2010 Maze

何を使って作るか？

- HTML（canvas）+ JavaScript を使ってつくるとよい
- 線分・長方形・円（楕円）などが色付きで描画できる
- 現代的ブラウザなら基本的に使えるはず

- JavaScript を覚える必要はあるが，canvas を使うだけだったら必要になる知識はほとんどない
  - 文法もかなり C++ に近い

canvas の使い方：おまじない

- ビジュアライザとして使うだけなら，とりあえず下を書く
- c = … の次の行にビジュアライズしたい内容を書いていく

```
<script>function f(){
    c = document.getElementById("id").getContext("2d");
}</script>
<body onload="f()">
<canvas id="id" width="500" height="500" />
</body>
```

座標の注意

- 数学では左下が (0, 0) になるが，コンピュータの世界では左上が (0, 0) になることが多い
- x 座標は右にいくと増え，y 座標は上に行くと増える
- 対処法は次の 3 つ
  1. 気にしない（上から a 行目，左から b 行目みたいな表現をすることも多いので）
  2. y 座標を指定するときに 500 – y などとする
  3. おまじない c.transform(1, 0, 0, -1, 0, 500); を c = … の直後に書く（これは，座標を x’ = x, y’ = 500 – y と変換することを表す）

線分（折れ線）の描画

- 次のように書く

```
c.beginPath();
c.moveTo(100, 100);
c.lineTo(200, 200);
(折れ線にしたい場合は，以降の点も順番に書く)
c.stroke();
```

- これで，(100, 100) と (200, 200) を結ぶ線分が描ける

長方形の描画

- 次のように書く

```
c.beginPath();
c.fillRect(x, y, width, height);
```

- 左上が (x, y), 横 width, 縦 height の長方形が描ける
- fillRect だと内部も塗りつぶされる
- strokeRect を使うと内部は塗りつぶされない

円の描画

- 次のように書く

```
c.beginPath();
c.arc(x, y, r, 0, Math.PI * 2, false);
c.stroke();
```

- 中心が (x, y), 半径 r の円が描ける
- stroke の代わりに fill を使うと，内部が塗りつぶされる
- 0, Math.PI * 2 の部分は，弧の始点，終点を弧度法で指定する
  - (x + r, y) から時計回りに進む向きで指定するので注意

色の指定

- c.beginPath(); の直後に，次のように書く

```
ctx.strokeStyle = ‘rgb(255, 0, 0)’;
```

- strokeStyle では，描画する線の色を指定
- fillStyle を使うと，塗りつぶしの色を指定できる
- 色は赤，緑，青の成分の強さをそれぞれ 0～255 で指定
  - 赤：rgb(255, 0, 0)
  - 青：rgb(0, 0, 255)
  - 緑：rgb(0, 255, 0)
  - 黒：rgb(0, 0, 0)
  - 白：rgb(255, 255, 255)

ファイルのビジュアライズ

- JavaScript でもファイルの読み込みはできるが，また覚えることが増える
- 入力したファイルをどう取り扱うかの問題もある
- 今まで述べた方法を用いて，C++ などで「ファイル → HTML」の変換をするのがよさそう

簡単な JavaScript の文法

- 変数宣言
  - `var hoge = 123;` とか `var hoge = "piyo";` とか
  - `var` を書かず宣言なしでいきなり代入してもよい
  - `int, char*` など，型を書かないことに注意
- 式は C++ とほとんど同じ形で書ける
  - `x = 5; y = x * 2 + 3;` のように書ける
  - 整数と浮動小数点数を区別しないので `5 / 2 = 2.5` であることに注意
- 文字列は *"aaa"*（ダブルクォーテーションで囲う）で表す
- 文字列同士を「足す」(*"a"* + *"b"* など）とくっつけた文字列になる
- 数と文字列を足すと，数を勝手に文字列に変換してからくっつける

if / for / 関数

- if 文は C++ とまったく同じ
- for 文も C++ とまったく同じだが，ループ変数の宣言に注意
  - `for(int i = 0; i < 5; ++i) { … }` ではなく
  - `for(i = 0; i < 5; ++i) { … }` でよい
- 関数は `function` *`hoge`* `(`*`arg1`*`,` *`arg2`*`) { … }` のように書く
  - 戻り値の型も，引数の型も指定しない
  - 値を返すときは普通に `return` *`e`*`;` でよい