Algoritmos

Geometría Computacional

MSc Edson Ticona Zegarra

Preparación ICPC Regionales

Contenido

Puntos

Listing 1: Puntos en C++

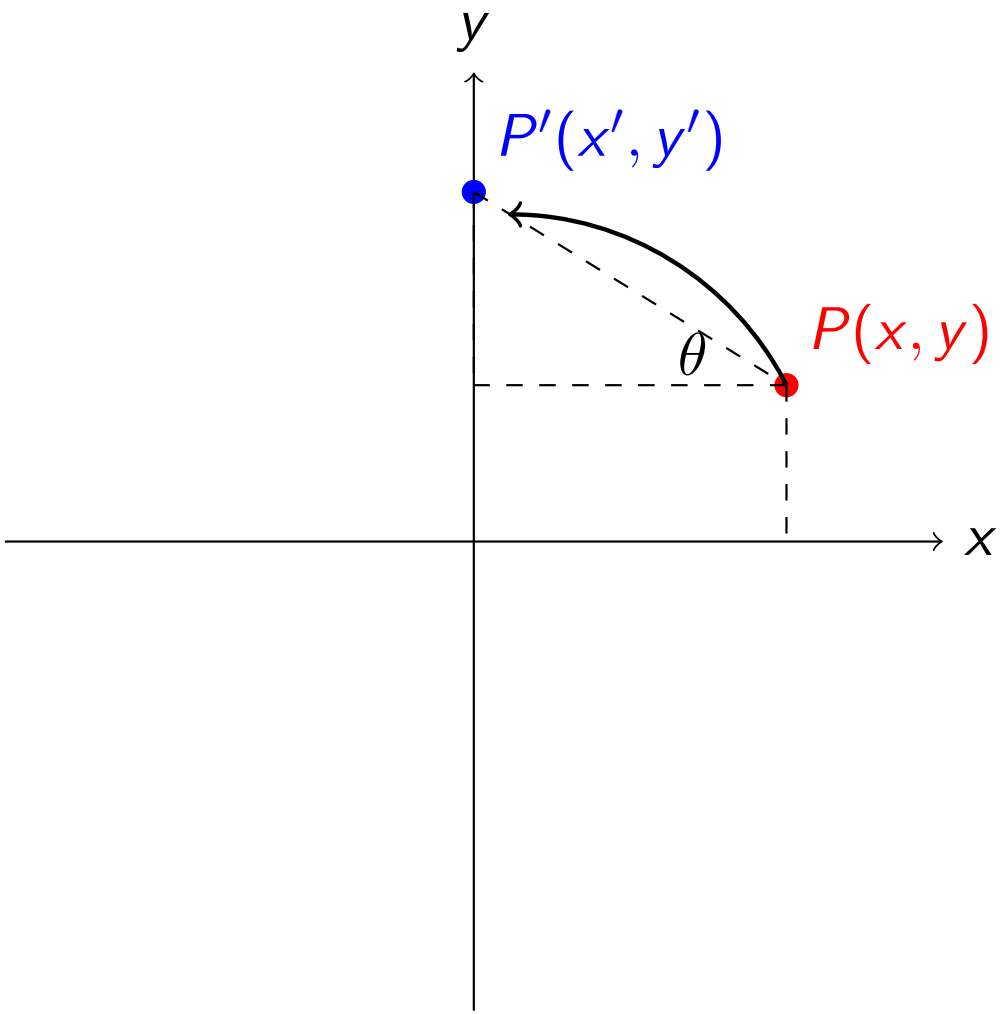
```
#include <iostream>
#include <cmath>
#define EPS 1e-6

using namespace std;

struct point_i {
  int x, y;
  point_i() { x = y = 0; }
  point_i(int _x, int _y) : x(_x), y(_y) {}
};
```

Puntos

Listing 2: Puntos double

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>
#define EPS 1e-6 // 10^-6 => 0.000001

using namespace std;

struct point {
  double x, y;
  point() { x = y = 0; }
  point(double _x, double _y) : x(_x), y(_y) {}

  bool operator < (point p2) const {
    if ( fabs(x-p2.x) > EPS ) // no son "iguales"
      return x < p2.x;
    return y < p2.y;
  }

  // operator overloading: sobrecarga de operadores
  bool operator == (const point &p2) const {
```

Rotación



Matriz de rotación

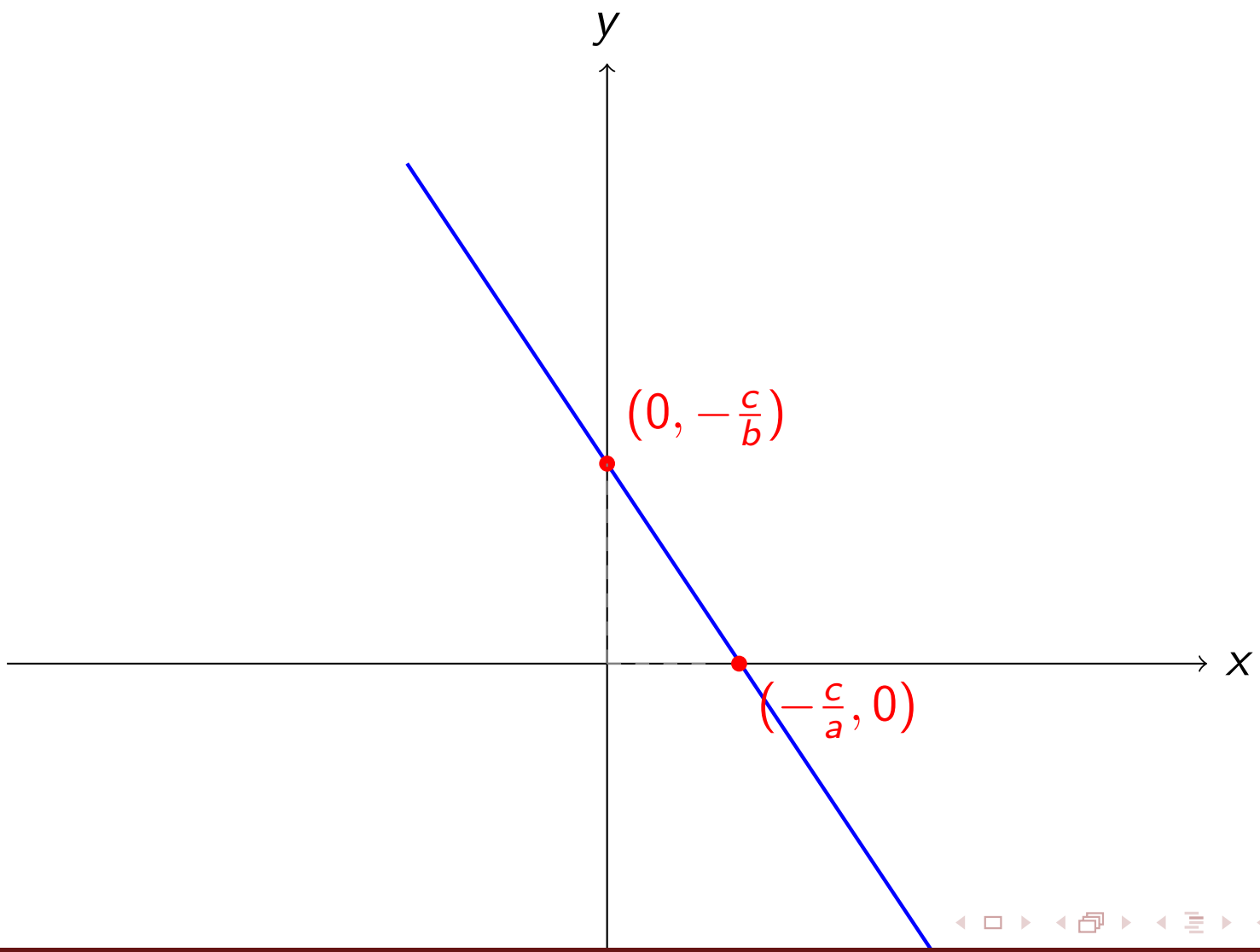
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotación

Listing 3: Rotación de puntos

```
#include <iostream>
#include <cmath>
#define EPS 1e-6 // 10^-6 => 0.000001

// cpp math library uses radians
double degrees2radians(double d) {
  return d*acos(-1.0) / 180;
}

point rotation(const point &p, double theta){
  double rad = degrees2radians(theta);
  return point(p.x*cos(rad) - p.y*sin(rad), p.x*sin(rad) + p.y*cos
}

using namespace std;
struct point {
  double x, y;
  point() { x = y = 0; }
  point(double _x, double _y) : x(_x), y(_y) {}

  bool operator < (point p2) const {
```

Linea: $ax + by + c = 0$



Lineas

Listing 4: Lineas en C++

```
#include <cmath>

struct line {
  double a, b, c; // ax + by + c = 0
}

void pointsToLine(const point &p1, const point &p2, line &l){
  if (fabs(p1.x-p2.x) < EPS)
    l = {1.0, 0.0, -p1.x };
  else
    l = {-(double)(p1.y-p2.y) / (p1.x-p2.x),
         -1.0,
         -(double)(l.a*p1.x) - p1.y };
}

void pointSlopeToLine(const point &p, double m, line &l){
  l.a = -m;
  l.b = 1.0;
  l.c = -((l.a*p.x) + (l.b*p.y));
}
```