



Geometría Computacional

Conceptos básicos

Agenda

- Repaso de Conceptos básicos de geometría
- Problemas geométricos
 - Con puntos
 - Con segmentos de línea
 - Con polígonos



Geometría

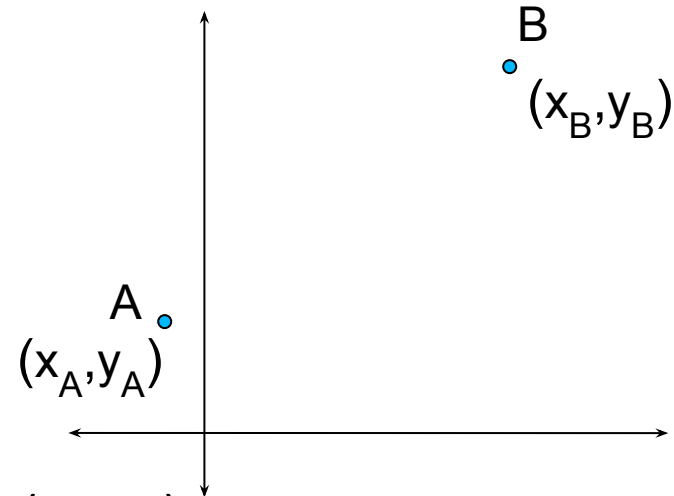
(repaso)

Líneas, Triángulos, Polígonos,
Circunferencia y Círculo

Puntos

Representación : $P (x_p, y_p)$

```
Type def struct {  
    double x;      /* x-coordenada */  
    double y;      /* y-coordenada */  
} point;
```

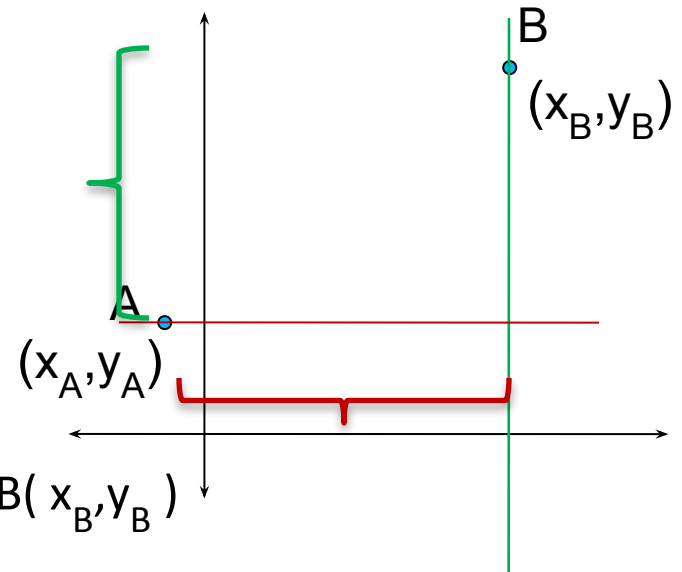


(a) Distancia entre dos puntos $A(x_A, y_A)$ y $B(x_B, y_B)$

Puntos

Representación : $P(x_p, y_p)$

```
Type def struct {  
    double x;      /* x-coordenada */  
    double y;      /* y-coordenada */  
} point;
```

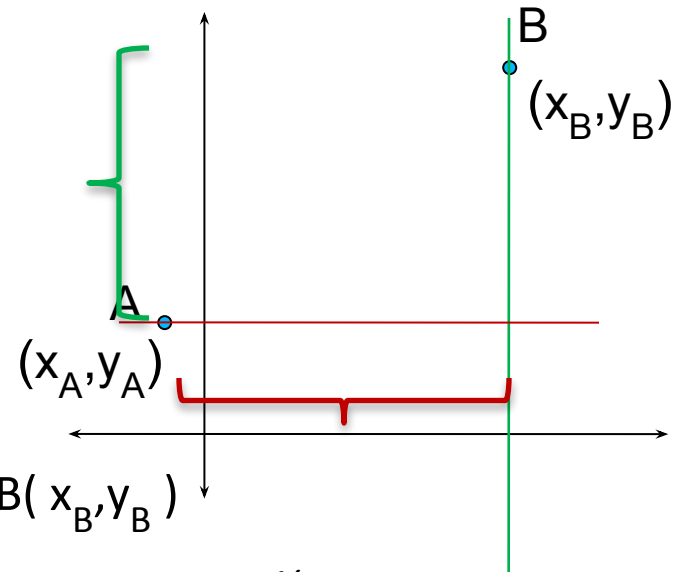


(a) Distancia entre dos puntos $A(x_A, y_A)$ y $B(x_B, y_B)$

Puntos

Representación : $P(x_p, y_p)$

```
Type def struct {  
    double x;      /* x-coordenada */  
    double y;      /* y-coordenada */  
} point;
```



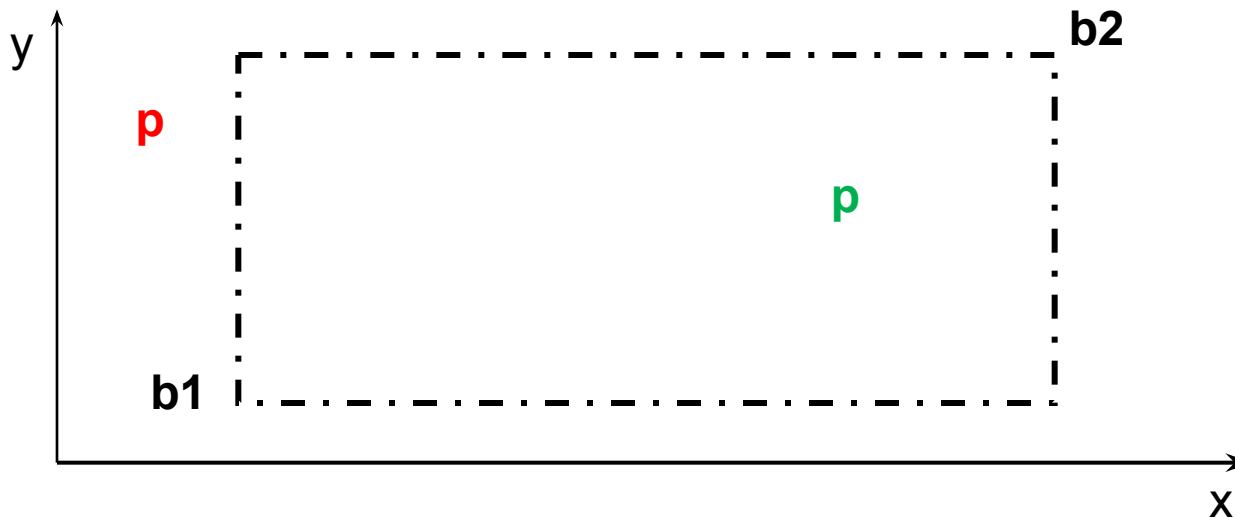
(a) Distancia entre dos puntos $A(x_A, y_A)$ y $B(x_B, y_B)$

$$\text{Dist}(A, B) = \left((x_B - x_A)^2 + (y_B - y_A)^2 \right)^{1/2}$$

Punto en Box

(b) point_in_box (point p, point b1, point b2)

```
{  
  return( (p[X] >= min(b1[X],b2[X])) && (p[X] <= max(b1[X],b2[X]))  
    && (p[Y] >= min(b1[Y],b2[Y])) && (p[Y] <= max(b1[Y],b2[Y])) );  
}
```



Líneas rectas

- Representación

- Ecuación : $ax + by + c = 0$

```
Type def struct {  
    double a;      /* x-coefficient */  
    double b;      /* y-coefficient */  
    double c;      /* constant term */  
} line;
```

- Algunas funciones útiles:

- » Puntos a línea
- » Punto y pendiente a línea
- » Intersección
- » Punto más cercano

Líneas rectas

(a) Ecuación : $ax + by + c = 0$

(b) Ecuación : $y = mx + n$ donde **m** es la pendiente y **n** es la ordenada al origen (donde se intersecta con el eje y)

Relación entre las definiciones (a) y (b)

A partir de (b) tenemos que: $y - mx - n = 0$ entonces $a = -m$, $b = 1$ y $c = -n$

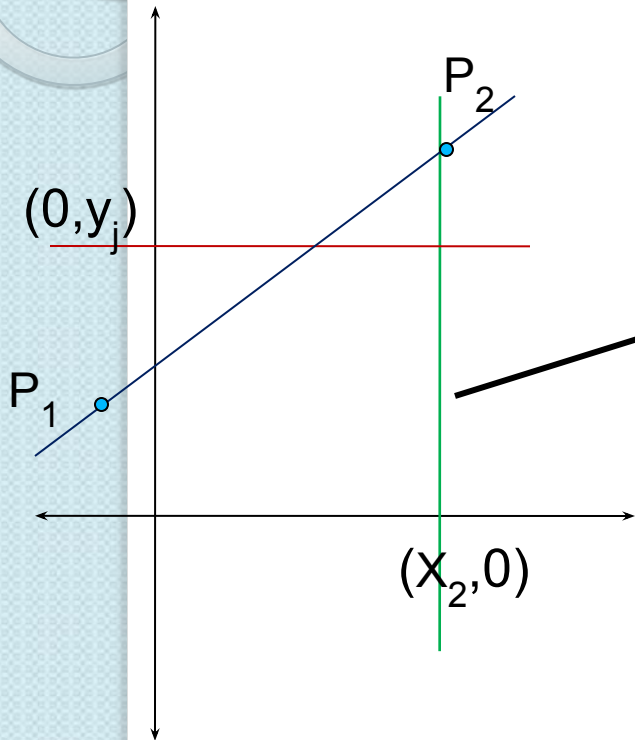
Para la Recta que pasa por los punto $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$ se tiene que la pendiente $m = (y_1 - y_2) / (x_1 - x_2)$ y ordenada al origen $n = y_1 - m x_1$

Atención con las rectas **verticales** !!!

Una recta vertical, se escribe como $x = c$, queda el denominador en 0

Líneas

Puntos a línea



```
1 ) points_to_line(point p1, point p2, line *l)
{
  if (p1[X] == p2[X]) {
    l->a = 1;
    l->b = 0;
    l->c = -p1[X];
  } else {
    l->b = 1;
    l->a = -(p1[Y]-p2[Y])/(p1[X]-p2[X]);
    l->c = -(l->a * p1[X]) - (l->b * p1[Y]);
  }
}
```

Vertical: $(x_2, y) \quad \forall y$
Horizontal: $(x, y_j) \quad \forall x$

Líneas

Punto y pendiente a línea

```
2) point_and_slope_to_line(point p,  
    double m, line *l)  
{  
    l->a = -m;  
    l->b = 1;  
    l->c = -((l->a*p[X]) + (l->b*p[Y]));  
}
```

Líneas

Paralelas – Idénticas

3) bool parallelQ(line l1, line l2)

{

return ((fabs(l1.a-l2.a) <= EPSILON) &&
 (fabs(l1.b-l2.b) <= EPSILON));

}

4) bool same_lineQ(line l1, line l2)

{

return (parallelQ(l1,l2) && (fabs(l1.c-l2.c) <= EPSILON));

}

Líneas : Intersección

5)

Un punto (x, y) cae en una línea $l : y = mx + b$ si al reemplazar x en la formula anterior obtenemos y .

El punto de intersección de las líneas $l_1 : y = m_1x + b_1$ y $l_2 : y = m_2x + b_2$ es el punto en donde ellas son iguales, es decir,

$$m_1x + b_1 = m_2x + b_2$$

$$x = (b_2 - b_1) / (m_1 - m_2), \quad y = m_1 * (b_2 - b_1) / (m_1 - m_2) + b_1$$

Líneas : Intersección

```
5) intersection_point(line l1, line l2, point p)
{
    if (same_lineQ(l1,l2)) {
        printf("Warning: Identical lines, all points intersect.\n");
        p[X] = p[Y] = 0.0;
        return;
    }
    if (parallelQ(l1,l2) == TRUE) {
        printf("Error: Distinct parallel lines do not intersect.\n");
        return;
    }
    p[X] = (l2.b*l1.c - l1.b*l2.c) / (l2.a*l1.b - l1.a*l2.b);
    if (fabs(l1.b) > EPSILON) /* test for vertical line */
        p[Y] = - (l1.a * (p[X]) + l1.c) / l1.b;
    else
        p[Y] = - (l2.a * (p[X]) + l2.c) / l2.b;
}
```

Líneas : Intersección - Ángulos

Dos líneas no paralelas cualesquiera se intersecan formando un ángulo

- Las líneas, escritas en su forma general,

$$l_1 : a_1 x + b_1 y + c_1 = 0 \quad y \quad l_2 : a_2 x + b_2 y + c_2 = 0$$

forman el ángulo θ dado por : $\text{tg } \theta = (a_1 b_2 - a_2 b_1) / (a_1 a_2 + b_1 b_2)$

- Para líneas en formato $l : y = mx + b$, el ángulo está dado por:

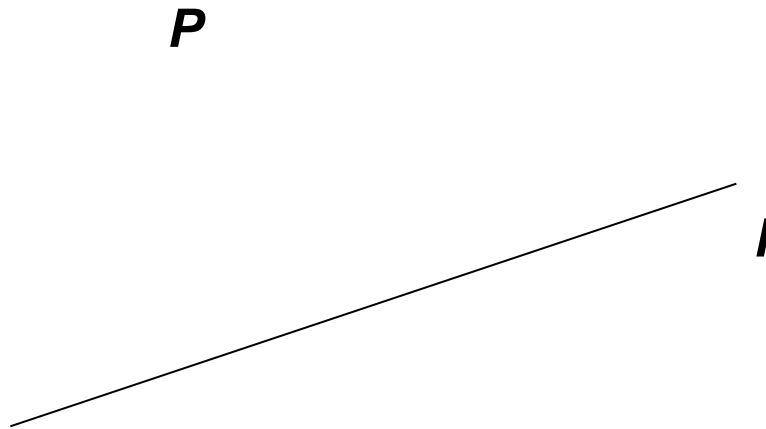
$$\text{tg } \theta = (m_2 - m_1) / (m_1 * m_2 + 1)$$

Dos líneas son **perpendiculares** si ellas se cruzan en ángulos **rectos** (90°). La línea perpendicular a $l : y = mx + b$ es $y = (-1/m)x + b'$ \forall valor de b' .

Líneas: Punto más cercano

6)

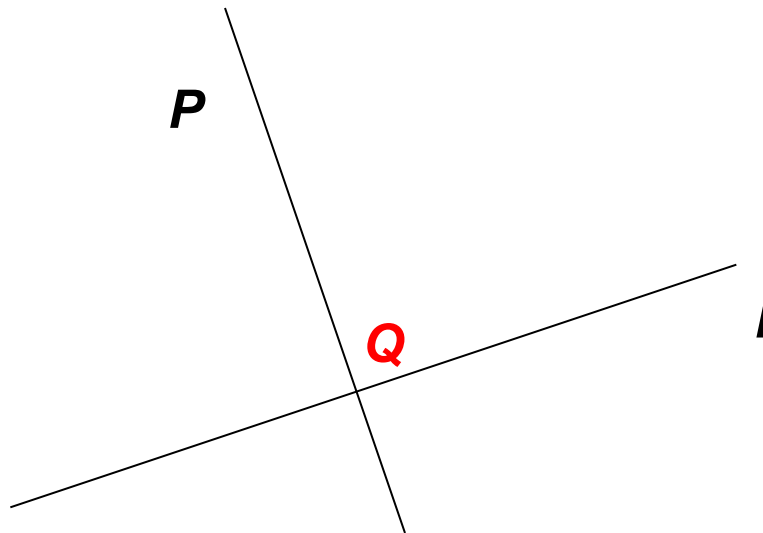
¿Cuál es el punto sobre la línea l más cercano a un punto P ?



Líneas: Punto más cercano

6)

El punto sobre la línea l más cercano a un punto P dado, es el punto Q intersección entre la línea l y la línea perpendicular a l que pasa por el punto P .



Líneas: Punto más cercano

```
6) closest_point(point p_in, line l, point p_c)
{
    line perp;          /* perpendicular to l through (x,y) */
    if (fabs(l.b) <= EPSILON) {          /* vertical line */
        p_c[X] = -(l.c);
        p_c[Y] = p_in[Y];
        return;
    }
    if (fabs(l.a) <= EPSILON) {          /* horizontal line */
        p_c[X] = p_in[X];
        p_c[Y] = -(l.c);
        return;
    }
    point_and_slope_to_line(p_in,1/l.a,&perp); /* normal case */
    intersection_point(l,perp,p_c);
}
```

Triángulos y trigonometría

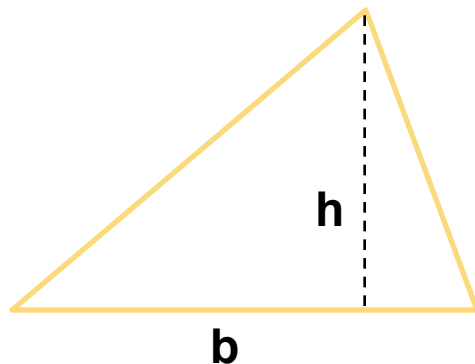
- Perímetro y área
- Triángulos rectángulos
 - Elementos : catetos, hipotenusa y ángulos
 - Teorema de Pitágoras

Triángulos: Perímetro y área

Perímetro : Suma de los lados del triángulo.

Conociendo las coordenadas de cada vértice, se calcula la longitud de cada lado y se suman.

Área: Producto entre la base y la altura, dividido 2.



Triángulos rectángulos

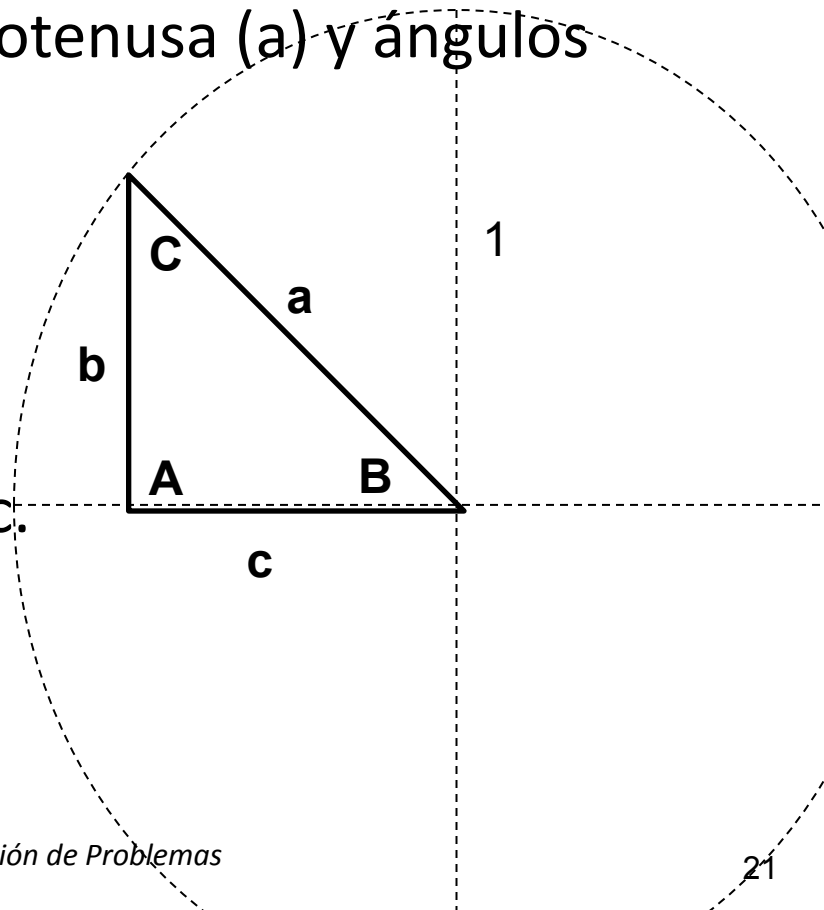
- Trigonometría

- Elementos: catetos (b, c), hipotenusa (a) y ángulos (A, B, C)

- Teorema de Pitágoras

$$a^2 = b^2 + c^2$$

- Funciones : seno, cos, tg, etc.

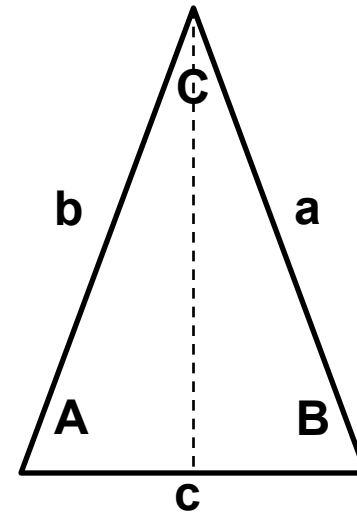


Triángulos no-rectángulos

- Relaciones trigonométricas

- Teorema del seno

$$\frac{a}{\text{sen } A} = \frac{b}{\text{sen } B} = \frac{c}{\text{sen } C}$$



- Teorema del coseno : generalización de Pitágoras

$$a^2 = b^2 + c^2 - 2 b c \cos A$$

Resolviendo Triángulos

Las propiedades trigonométricas nos permiten realizar diferentes cálculos.

Resolver triángulos es el arte de poder obtener ángulos y longitudes de lados de un triángulo a partir de un subconjunto de datos. Los problemas entran en dos categorías:

- *Dados dos ángulos y un lado, encontrar el resto.*
- *Dados dos lados y un ángulo, encontrar el resto.*

Resolviendo Triángulos

Los problemas de resolución de triángulos entran en dos categorías:

- ***Dados dos ángulos y un lado, encontrar el resto.*** Encontrar el tercer ángulo es fácil, ya que la suma de los tres es 180° . El teorema del seno nos permite encontrar el resto de los lados.
- ***Dados dos lados y un ángulo, encontrar el resto.*** Si el ángulo se encuentra entre los dos lados, el teorema del coseno nos permite encontrar el tercer lado. Luego, con el teorema del seno calculamos los restantes ángulos. De lo contrario, podemos usar el teorema del seno y la propiedad de la suma de los ángulos para determinar todos los ángulos, y luego el teorema del seno para obtener el lado restante.

Área de Triángulos

El área de un triángulo T es:

$$A(T) = \frac{1}{2} b * h$$

b es la base, representada por uno de los lados

h es la altura, representada por la distancia entre el tercer vértice y la base. La altura puede ser calculada con alguna de las propiedades vistas.

Otra forma de calcular el área es directamente desde la representación por coordenadas (usando álgebra lineal y determinantes).

$A(T)$ de un triángulo $T=(a, b, c)$ es:

$$2 * A(T) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = a_x b_y - a_y b_x + a_y c_x - a_x c_y + b_x c_y - c_x b_y$$

Área de Triángulos

El área de un triángulo T es:

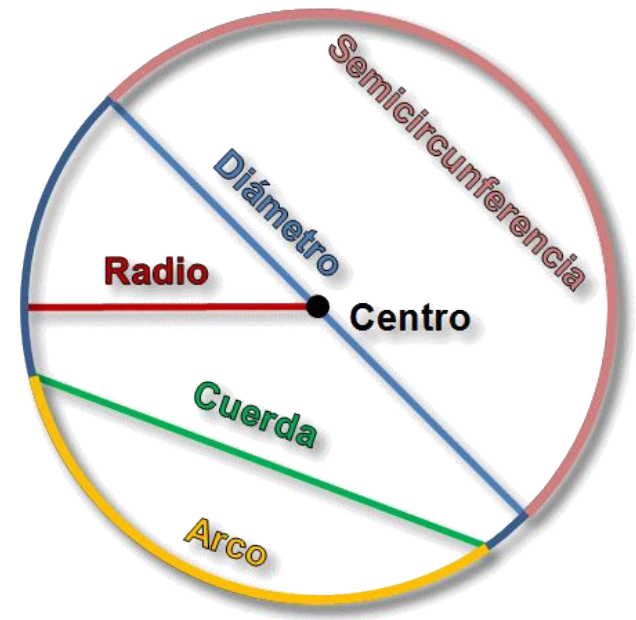
```
double signed_triangle_area(point a, point b, point c) {  
    return( (a[X]*b[Y] - a[Y]*b[X] + a[Y]*c[X]  
            - a[X]*c[Y] + b[X]*c[Y] - c[X]*b[Y]) / 2.0 );  
}
```

```
double triangle_area(point a, point b, point c) {  
    return( fabs(signed_triangle_area(a,b,c)) );  
}
```

Circunferencia y círculo

Elementos:

- Radio: r
- Centro: o
- Cuerda: segmento que une dos puntos de la circunferencia
- Diámetro: es una cuerda que pasa por el centro
- Arco de circunferencia: es cada una de las partes en que la cuerda divide la circunferencia
- Semicircunferencia: mitad de la circunferencia



Circunferencia y círculo

Longitud de la circunferencia:

$$L = 2 \pi r$$

Área del círculo:

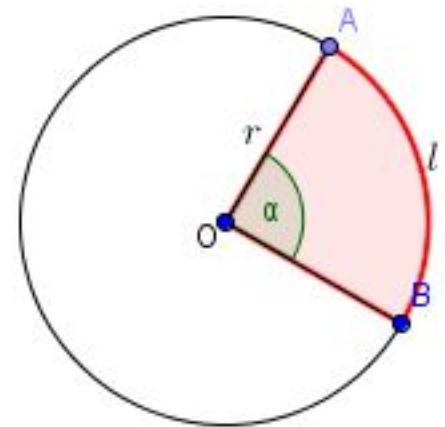
$$A = \pi r^2$$

Longitud del arco de circunferencia:

$$L.\text{arco} = L \alpha^\circ / 360^\circ = 2 \pi r \alpha^\circ / 360^\circ$$

Área del sector circular:

$$A.\text{sector} = A \alpha^\circ / 360^\circ = \pi r^2 \alpha^\circ / 360^\circ$$



Bibliografía

Programming Challenges. The Programming Contest Training Manual.
Steven S. Skiena, Miguel A. Revilla. Springer-Verlag New York, Inc., 2003
ISBN 0-387-00163-8.

Introduction to Algorithms, 2nd Ed –Thomas H. Cormen
ISBN 0-262-03293-7 (hc.: alk. paper, MIT Press).-ISBN 0-07-013151-1
(McGraw-Hill)