



# Algoritmos



## Logro de sesión

- Al finalizar la sesión, el estudiante **utilizar relaciones de generalización entre clases** para la construcción de programas.



# Relación de Generalización

## Contenido:

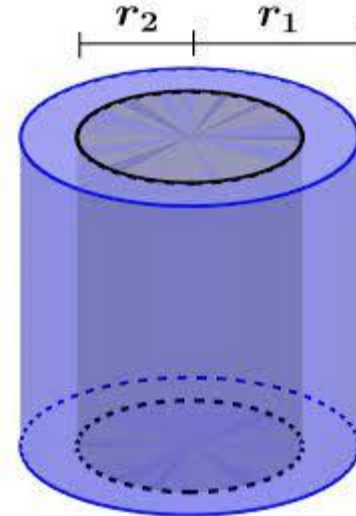
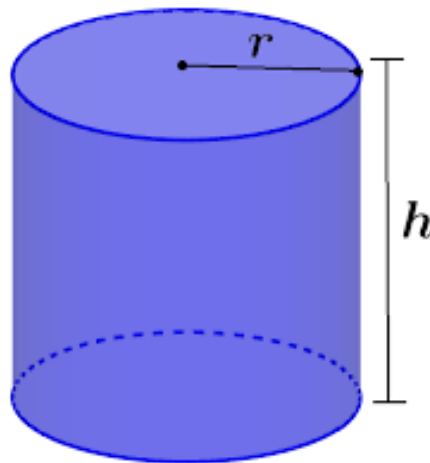
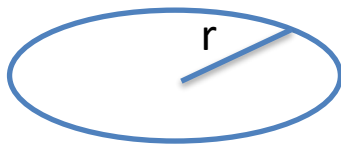
- Generalización [Herencia]
  - Relaciones entre clases de generalización
  - Diagrama relaciones de generalización

# Ejemplo



**Escribir un POO de entorno de consola que permita hallar:**

1. La longitud y área de un círculo
2. El área y volumen de un cilindro
3. El área y el volumen de un cilindro hueco.



# Ejemplo



**Escribir un POO de entorno de consola que permita hallar:**

1. La longitud y área de un círculo
2. El área y volumen de un cilindro
3. El área y el volumen de un cilindro hueco.

**Del Círculo:**

$$\text{Longitud} = 2\pi r$$

$$\text{Area} = \pi r^2$$

**Del Cilindro:**

$$\text{Area} = 2\pi r h + 2\pi r^2$$

$$\text{Volumen} = \pi r^2 h$$

**Cilindro Hueco:**

$$\text{Area} = 2\pi r h + 2\pi h i + 2\pi r^2 - 2\pi i^2$$

$$\text{Volumen} = \pi r^2 h - \pi i^2 h$$

**Donde:**

r es el radio mayor  
i es el radio menor  
h es la altura

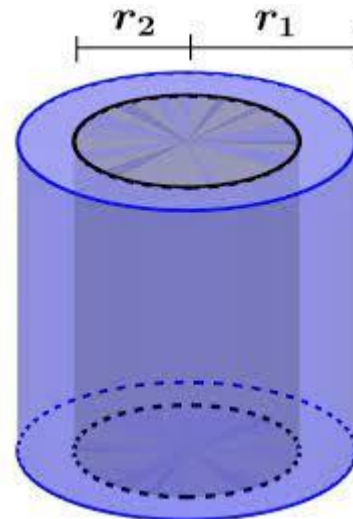
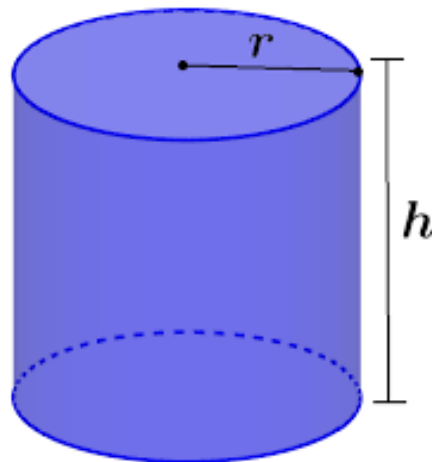
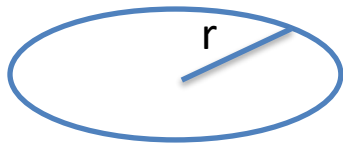
# Ejemplo



En esencia podríamos decir que:

- Un objeto cilindro es un objeto círculo con una altura
- Un objeto cilindro hueco es un cilindro con un espacio hueco.

Aunque semánticamente el ejemplo no es perfecto nos permitirá indicar la sintaxis del lenguaje para los casos de herencia.



# Ejemplo



**1er paso:** Identificamos clases:

- Circulo
- Cilindro
- Cilindro Hueco

**2do paso:** Definimos atributos y métodos para cada clase

# Ejemplo



```
class CCirculo
```

```
{private:  
    double r;  
public:  
    CCirculo(double r);  
    ~CCirculo();  
    double AreadelCirculo();  
    double Longitud();  
};
```

```
class CCilindro
```

```
{ private:  
    double r;  
    double h;  
public:  
    CCilindro(double r, double h);  
    ~CCilindro();  
    double AreadelCilindro();  
    double VolumendelCilindro();  
};
```

```
class CCilindroHueco
```

```
{private:  
    double r; //--- radio mayor  
    double i; //--- radio menor  
    double h;  
public:  
    CCilindroHueco(double r, double i, double h);  
    ~CCilindroHueco();  
    double AreadelCilindroHueco();  
    double VolumendelCilindroHueco();  
};
```



# Ejemplo



```
class CCirculo
{private:
    double r;
public:
    CCirculo(double r);
    ~CCirculo();
    double AreadelCirculo();
    double Longitud();
};
```

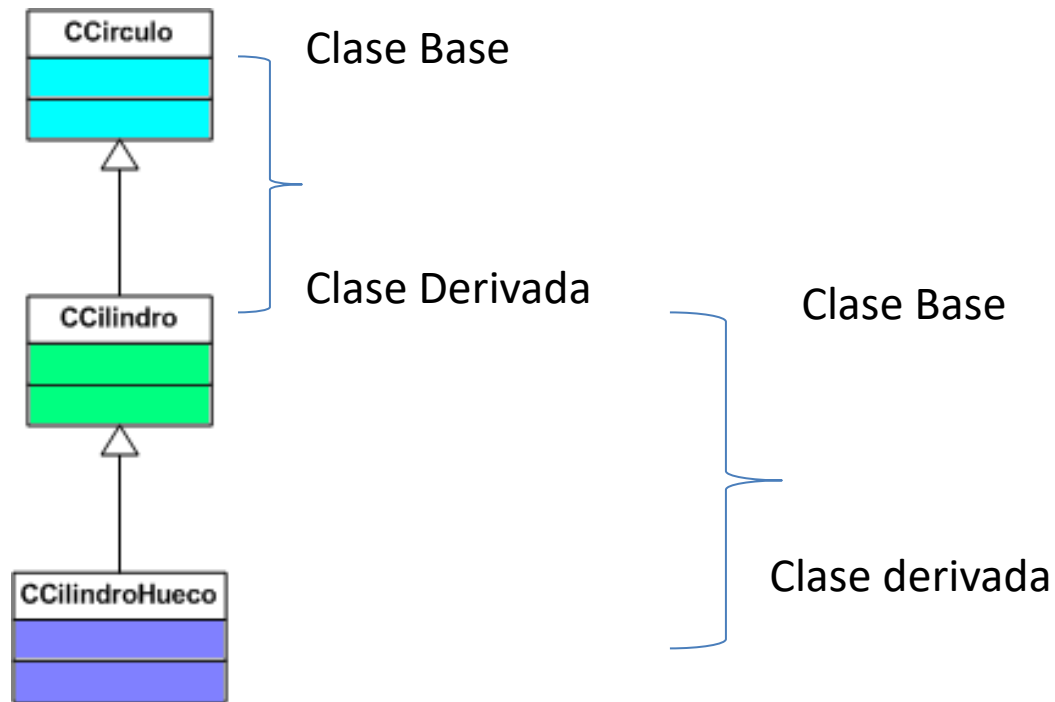
```
class CCilindro
{ private:
    double r;
    double h;
public:
    CCilindro(double r, double h);
    ~CCilindro();
    double AreadelCilindro();
    double VolumendelCilindro();
};
```

```
class CCilindroHueco
{private:
    double r; //--- radio mayor
    double i; //--- radio menor
    double h;
public:
    CCilindroHueco(double r, double i, double h);
    ~CCilindroHueco();
    double AreadelCilindroHueco();
    double VolumendelCilindroHueco();
};
```

**3er paso:** Se observan si hay atributos en común, si los hay se define la clase base.

Si hay métodos comunes se analiza y si los hay se los define en la clase base.

# Ejemplo



# Ejemplo



## 3er paso: Continuando...

```
class CCirculo  
{ protected:  
    double r;  
    public:  
        CCirculo(double r);  
        ~ CCirculo();  
    double AreadelCirculo();  
    double Longitud();  
};
```

```
class CCilindro : public CCirculo  
{ protected:  
    double h;  
    public:  
        CCilindro(double r, double h);  
        ~CCilindro();  
    double AreadelCilindro();  
    double VolumendelCilindro();  
};
```

```
class CCilindroHueco : public CCilindro  
{ protected:  
    double i; //--- radio menor  
    public:  
        CCilindroHueco(double r, double i, double h);  
        ~ CCilindroHueco();  
    double AreadelCilindroHueco();  
    double VolumendelCilindroHueco();  
};
```

```
//-- Figuras.h
```

```
class CCirculo
```

```
{protected:
```

```
    double r;↵
```

```
    public:
```

```
        CCirculo(double r);
```

```
        ~ CCirculo();
```

```
        double AreadelCirculo();
```

```
        double Longitud();
```

```
};
```

```
class CCilindro : public CCirculo
```

```
{ protected:
```

```
    double h;
```

```
    public:
```

```
        CCilindro(double r, double h);
```

```
        ~CCilindro();
```

```
        double AreadelCilindro();
```

```
        double VolumendelCilindro();
```

```
};
```

```
class CCilindroHueco : public CCilindro
```

```
{protected:
```

```
    double i;  //-- radio menor
```

```
    public:
```

```
        CCilindroHueco(double r, double i, double h);
```

```
        ~ CCilindroHueco();
```

```
        double AreadelCilindroHueco();
```

```
        double VolumendelCilindroHueco();
```

```
};
```

Las palabras reservadas: **private**, **protected** y **public** pueden aparecer en cualquier orden y cualquier número de veces en una clase, particionando éstas en múltiples partes, privadas, públicas o protegidas.

**Protected** indica que todo aquello que se encuentre en esta sección puede ser accedida por cualquier clase derivada de la clase base.

**public** en la cabecera de la definición de una clase, define la relación de herencia

# Clase Circulo



```
class CCirculo
{
protected:
    double r;
public:
    CCirculo(double r);
    ~CCirculo();
    double AreadelCirculo();
    double Longitud();
};
```

```
//----- CCirculo -----
```

```
CCirculo::CCirculo(double r)
{
    this->r = r;
}
```

```
CCirculo::~~CCirculo() {}
```

```
double CCirculo::AreadelCirculo()
{
    return(3.1416*r*r);
}
```

```
double CCirculo::Longitud()
{
    return(2*3.1416*r);
}
```

# Clase Cilindro



```
class CCilindro : public CCirculo
{
protected:
    double h;
public:
    CCilindro(double r, double h);
    ~CCilindro();
    double AreadelCilindro();    double VolumendelCilindro();
};

//----- CCilindro -----
CCilindro::CCilindro(double r, double h) :CCirculo(r)//Invoca al constructor del Padre
{
    this->h = h;
}

CCilindro::~~CCilindro() {}

double CCilindro::AreadelCilindro()
{
    return(Longitud() *h + 2 * AreadelCirculo());
}

double CCilindro::VolumendelCilindro()
{
    return(AreadelCirculo() *h);
}
```

# Clase CilindroHueco



```
class CCilindroHueco : public CCilindro
{
protected:
    double i;  //-- radio menor
public:
    CCilindroHueco(double r, double i, double h);
    ~CCilindroHueco();
    double AreadelCilindroHueco();
    double VolumendelCilindroHueco();
};

//----- CCilindroHueco -----

CCilindroHueco::CCilindroHueco(double r, double i, double h) :CCilindro(r, h)//invoca al constructor de la clase Padre
{
    this->i = i;
}







CCilindroHueco::~~CCilindroHueco() {}

double CCilindroHueco::AreadelCilindroHueco()
{
    return(AreadelCilindro() + 2 * 3.1416*h*i - 2 * 3.1416*i*i);
}

double CCilindroHueco::VolumendelCilindroHueco()
{
    return(VolumendelCilindro() - 3.1416*i*i*h);
}
```

# Ejercicio de Aplicación



FORMA	ELEMENTOS	FÓRMULA PERÍMETRO	FÓRMULA ÁREA
<b>TRIÁNGULO</b> 	b: Base h: Altura  l: Lado1 m: Lado2 n: Lado3	$P = l + m + n$	$A = \frac{b \times h}{2}$
<b>CUADRADO</b> 	a: Lado	$P = 4a$	$A = a^2$
<b>RECTÁNGULO</b> 	b: Base h: Altura	$P = 2b + 2h$	$A = b \times h$
<b>ROMBO</b> 	a: Lado  d: Diagonal menor D: Diagonal mayor	$P = 4a$	$A = \frac{D \times d}{2}$
<b>ROMBOIDE</b> 	b: Base h: Altura	$P = 2b + 2h$	$A = b \times h$
<b>TRAPECIO</b> 	l: Lado1 m: Lado2 n: Lado3 o: Lado4  b: Base menor B: Base mayor h: Altura	$P = l + m + n + o$	$A = \frac{h (B + b)}{2}$

Se le solicita que, haciendo uso de conceptos de POO y relaciones de herencia, elabore un programa para calcular el perímetro y área de las figuras mostradas.

Para la solución considere todas las relaciones de herencia que crea conveniente.