

# Entrada / Salida

# Streams

# Archivos

Algoritmos y Estructuras de Datos I

Departamento de Computación,  
Facultad de Ciencias Exactas y Naturales,  
Universidad de Buenos Aires

# Agenda

- **Tipos básicos**
- **Repaso de Clases/Objetos**
- **Entrada / Salida por pantalla (cin / cout)**
- **Entrada / Salida por archivo (fstream)**

# Tipos Básicos

- **int:** número entero
- **double:** número real
- **char:** caracter
- **String:** secuencia de caracteres.

# Salida

```
#include <iostream> // Biblioteca Entrada/Salida
```

```
using namespace std;
```

```
int main(int argc, char *argv[])  
{  
    cout << "¡Hola Mundo!";  
    return 0;  
}
```



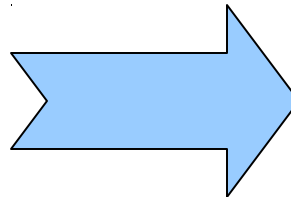
*Los invitamos a implementarlo*

# Salida

```
#include <iostream> // Biblioteca Entrada/Salida
```

```
using namespace std;
```

```
int main(int argc, char *argv[])  
{  
    cout << "¡Hola Mundo!";  
    return 0;  
}
```



# Salida

```
#include <iostream> // Biblioteca Entrada/Salida

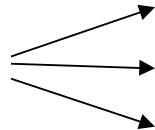
using namespace std;

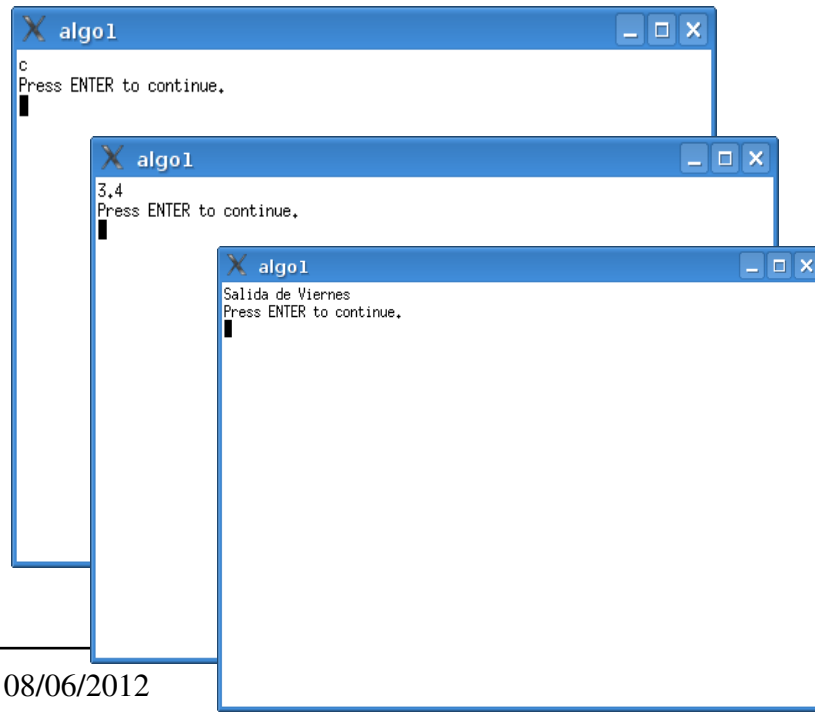
int main(int argc, char *argv[])
{
    int i = 4;
    cout << i;
    return 0;
}
```

# Salida

```
#include <iostream> // Biblioteca Entrada/Salida
```

```
using namespace std;
```

```
int main(int argc, char *argv[])  
{  
    int i = 4;  char i = 'c';  
    double i = 3.4;  
    string i = "Salida de Viernes";  
    cout << i;  
    return 0;  
}
```



# Salida

```
#include <iostream>    // Biblioteca Entrada/Salida

using namespace std;

int main(int argc, char *argv[])
{
    int Argentina = 1;
    int Nigeria = 4;

    cout << "Nigeria gano por " << (Nigeria - Argentina) << " goles";
    return 0;
}
```



*Los invitamos a implemententarlo*



# Salida

```
#include <iostream> // Biblioteca Entrada/Salida
```

```
using namespace std;
```

```
int main(int argc, char *argv[])  
{
```

```
    int Argentina = 1;
```

```
    int Nigeria = 4;
```

```
    cout << "Argentina: " << Argentina << " goles" << endl;
```

```
    cout << "Nigeria : " << Nigeria << " goles";
```

```
    return 0;
```

```
}
```



*Los invitamos a implementarlo*

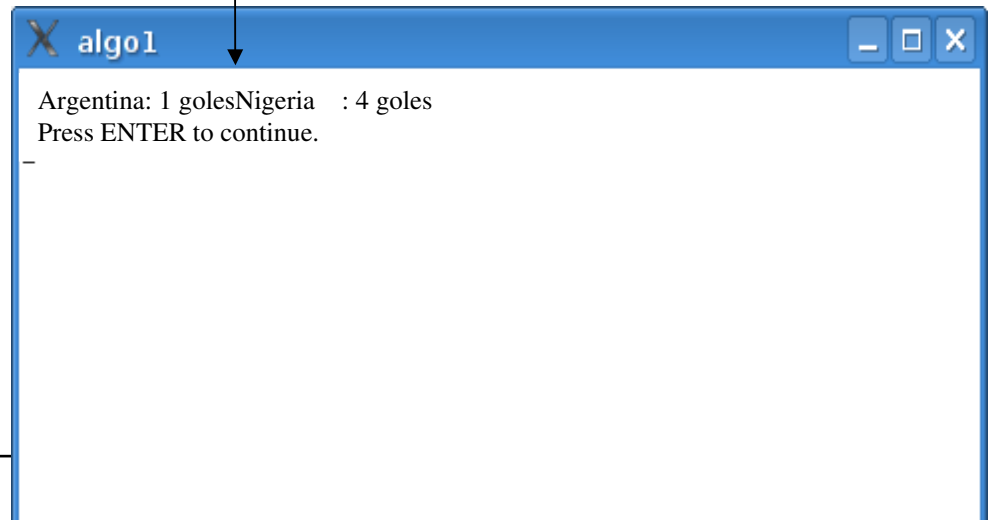
# Salida

```
#include <iostream> // Biblioteca Entrada/Salida

using namespace std;

int main(int argc, char *argv[])
{
    int Argentina = 1;
    int Nigeria = 4;

    cout << "Argentina " << Argentina << " goles" << endl;
    cout << "Nigeria " << Nigeria << " goles";
    return 0;
}
```



```
algo1
Argentina: 1 golesNigeria : 4 goles
Press ENTER to continue.
```

# Salida

Un gran poder conlleva una gran responsabilidad ...



# Salida

**Ejercicio: Hacer un programa que muestre por pantalla los primeros mil números naturales.**

**... tienen 5 min. por reloj.**



*Tipear los 1000 número me va a llevar más de 5 min. ...*



*... qué era eso de los ciclos?*



# Objetos



# Clases



# Objetos



## Atributos

- Modelo
- Color
- Rodado

## Métodos

- Incrementar rodado
- Perímetro de rueda
- Iguales

# bicicleta.h

```
1  #ifndef BICICLETA_INCLUDED
2  #define BICICLETA_INCLUDED
3
4  #include <iostream>
5
6  using namespace std;
7
8  class Bicicleta{
9
10     public:
11         Bicicleta();
12         Bicicleta(const string m, const string c, const int r);
13
14
15         void incrementarRodado(const int c);
16         double perimetroRueda() const;
17         bool operator==(const Bicicleta b2);
18
19     private:
20         string _modelo;
21         string _color;
22         int _rodado;
23
24 }; // <-- Muy importante este punto y coma
25
26 #endif //BICICLETA_INCLUDED
27
```



## Atributos

- Modelo
- Color
- Rodado

## Métodos

- Incrementar Rodado
- Perimetro de rueda
- Iguales



# bicicleta.cpp

```
1  #include "bicicleta.h"
2
3  Bicicleta::Bicicleta(){
4      _modelo="paseo";
5      _color="amarillo";
6      _rodado=20;
7  };
8
9  Bicicleta::Bicicleta(const string m, const string c, const int r){
10     _modelo=m;
11     _color=c;
12     _rodado=r;
13 };
14
15 void Bicicleta::incrementarRodado(const int c){
16     _rodado=_rodado+c;
17 };
18
19 double Bicicleta::perimetroRueda() const{
20     double ret;
21     ret=3.14 * _rodado;
22     return ret;
23 };
24
25 bool Bicicleta::operator==(const Bicicleta b2){
26     return ( (_modelo==b2._modelo) && (_color==b2._color) );
27 };
```



## Atributos

- Modelo
- Color
- Rodado

## Métodos

- Incrementar Rodado
- Perimetro de rueda
- Iguales

# main.cpp

```
1  #include <iostream>
2  #include "bicicleta.h"
3
4  using namespace std;
5
6  int main(int argc, char *argv[]){
7      Bicicleta b1("cross", "rojo", 20);
8      Bicicleta b2("cross", "rojo", 26);
9      Bicicleta b3;
10
11      cout << "Perimetro b1: " << b1.perimetroRueda() << endl;
12      cout << "b1 == b2 : " << (b1==b2) << endl;
13      cout << "b1 == b3 : " << (b1==b3) << endl;
14
15      return 0;
16  }
```



## Atributos

- Modelo
- Color
- Rodado

## Métodos

- Incrementar Rodado
- Perimetro de rueda
- Iguales

# Salida

**Ejercicio: Agregar al tipo Bicicleta los métodos públicos**

```
string modelo() const;  
string color() const;  
int rodado() const;  
void mostrarsePorPantalla() const;
```



**... tienen 7 min. por reloj.**

*Métodos Públicos ...  
Métodos para tod@s*

# Salida

**¿Qué pasa si utilizo ...**

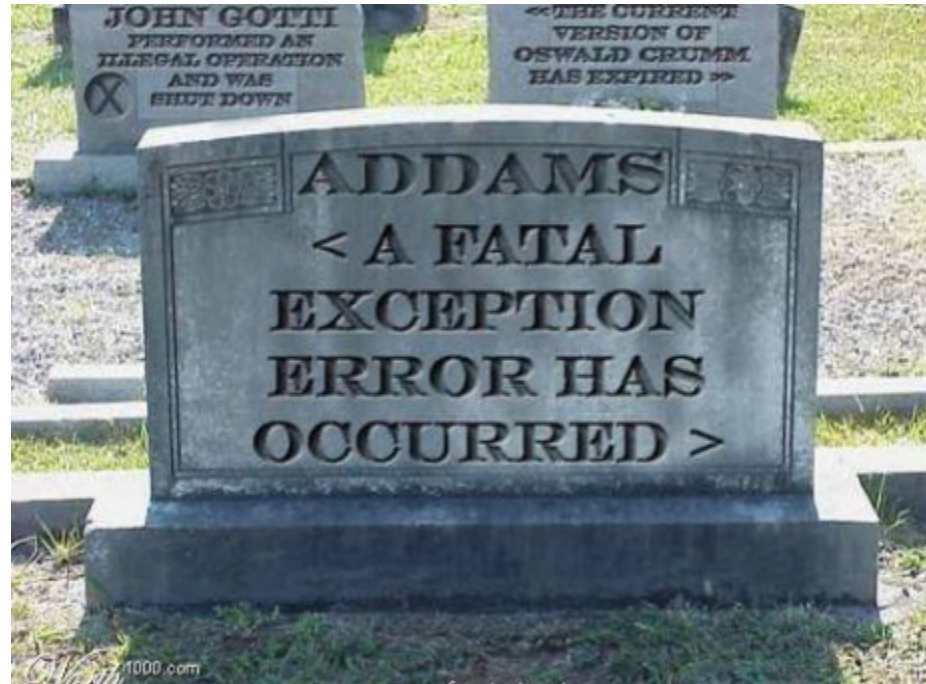
```
Bicicleta b1("paseo","rojo",20);  
cout << b1;
```

# Salida

¿Qué pasa si utilizo ...

```
Bicicleta b1("paseo","rojo",20);  
cout << b1;
```

**error: no match for 'operator<<' in .....**



# Salida

**¿Solución: Implementar el operador << para el tipo Bicicleta ...**

bicicleta.h

**cout**

**b1**



```
std::ostream & operator<<(std::ostream & os,const Bicicleta & b);
```

bicicleta.cpp

```
std::ostream & operator<<(std::ostream & os,const Bicicleta & b){  
    os << "Modelo: " << b.modelo() << " Color: " << b.color()  
    << " Rodado: " << b.rodado();  
    return os;  
}
```

# Salida

**Ejercicio: implementar operator << en Bicicleta**

**... tienen 3 min. por reloj.**



# Salida



bicicleta.h

cout/**archivo**

**b1**

std::ostream & operator<<(std::ostream & os,const Bicicleta & b);





# Salida

```
1  #include <iostream>
2  #include "bicicleta.h"
3  #include <fstream>
4
5  using namespace std;
6
7  int main(int argc, char *argv[]){
8      Bicicleta b1("cross", "rojo", 20);
9      Bicicleta b2("cross", "rojo", 26);
10     Bicicleta b3;
11
12     cout << "Bicicletas " << endl;
13     cout << "=====" << endl;
14     cout << "b1 --> " << b1 << endl;
15     cout << "b2 --> " << b2 << endl;
16     cout << "b3 --> " << b3 << endl;
17
18     ofstream miSalida("test.txt");
19     miSalida << b1;
20     miSalida.close();
21
22     return 0;
23 }
```

} NUEVO

} NUEVO



Salida





# Entrada

bicicleta4.txt

```
( cross amarillo 20 )
```

```
1  #include <iostream>
2  #include "bicicleta.h"
3  #include <fstream>
4  using namespace std;
5
6  int main(int argc, char *argv[]){
7      //...//
8      ifstream miArchivoBicicleta("bicicleta4.txt");
9
10     char c;
11     string modelob4;
12     string colorb4;
13     int rodadob4;
14
15     miArchivoBicicleta >> c; // descarto el parentesis (
16     miArchivoBicicleta >> modelob4; // leo el modelo
17     miArchivoBicicleta >> colorb4; // leo el color
18     miArchivoBicicleta >> rodadob4; // leo el rodado
19     miArchivoBicicleta >> c; // descarto el parentesis )
20
21     miArchivoBicicleta.close();
22     // armo la bicicleta
23     Bicicleta b4(modelob4, colorb4, rodadob4);
24     // muestro por pantalla
25     cout << "Bicicletas " << endl;
26     cout << "=====" << endl;
27     cout << "b4 --> " << b4 << endl;
28
29     return 0;
30 }
```

Por qué levanta “cross” y no levanta “cross amarillo” ?  
Y si el archivo tiene:

```
( cross racer amarillo 20 )
```

o tiene:

```
( cross racer amarillo patito 20 )
```

# Entrada

**bicicleta4.txt**

```
( |cross racer| |amarillo patito| 2 )
```



+ ciclos!!!

# Entrada

bicicleta4.txt

```
( |cross racer| |amarillo patito| 2 )
```

```
2  #include "bicicleta.h"
3  #include <fstream>
4  using namespace std;
5
6  int main(int argc, char *argv[]){
7      //...//
8      ifstream miArchivoBicicleta("bicicleta4.txt");
9
10     char c;
11     string modelob4;
12     string colorb4;
13     int rodadob4;
14
15     miArchivoBicicleta >> c; // descarto el parentesis (
16     miArchivoBicicleta >> c; // descarto el pipe |
17     getline(miArchivoBicicleta,modelob4,'|'); // leo el modelo
18     miArchivoBicicleta >> c; // descarto el pipe |
19     getline(miArchivoBicicleta,colorb4,'|'); // leo el color
20     miArchivoBicicleta >> rodadob4; // leo el rodado
21     miArchivoBicicleta >> c; // descarto el parentesis )
22
23     miArchivoBicicleta.close();
24     // armo la bicicleta
25     Bicicleta b4(modelob4, colorb4, rodadob4);
26     // muestro por pantalla
27     cout << "Bicicletas " << endl;
28     cout << "===== " << endl;
29     cout << "b4 --> " << b4 << endl;
30
31     return 0;
32 }
```

Si no ponemos  
separador, toma  
por defecto fin  
de línea

# Y nos faltó ...

Y no vimos ... pero pueden investigar:

**`cin`**

**`archivo.good()`**

**`archivo.peek()`**

**`archivo.MASOPCIONES`**



**¿Preguntas?**