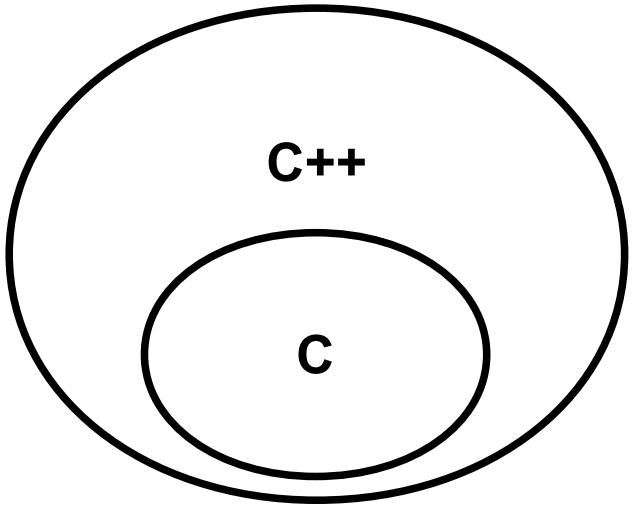
Introducción a C++

- ¿Qué lenguaje vamos utilizar?
- La biblioteca estándar
- E/S por consola, cadenas y archivos
- Funciones
  - Pasaje de parámetros por copia/referencia
- Argumentos de los programas
- Ejemplos tipo práctico 1

¿Qué lenguaje vamos a utilizar?

El lenguaje que utilizaremos en los proyectos es C++.

C++ puede considerarse un superconjunto de C:



En muchas ocasiones, tendremos que elegir entre la “forma de hacer las cosas” (es decir, entre las bibliotecas, mecanismos y estilo) de C o C++.

Introducción al lenguaje C++

- Programas y bloques

```
int main()
{
    return 0;
}
```

- Identificadores:
  - Deben comenzar con una letra o guión bajo.
  - Sólo letras (A-Z, a-z), dígitos (0-9) o el guión bajo (_) pueden seguir al primer símbolo.
- Tipos y variables:
  - Espacio de memoria para guardar algún valor.
  - Tipos básicos: char, int, float, double, bool.
  - Operadores: +, /, *, -, !=, ==, =, <, >, <=, >=, &&, ||.

```
int a, b, c;
char d;
```

Estructuras de control (1)

- Estructuras de selección
  - Sentencia **if**

```
int b;
asignar(b)
if (b > 10) {
    procesar(b)
} else if (b < 0) {
    ...
} else {
    ...
}
```

  - Sentencia **switch**

```
int opcion;
switch (opcion) {
    case 1:
        …
        break;
    case 2:
    case 3:
        …
        break;
    default: {
        …
    }
        break;
}
```

Estructuras de control (2)

- Estructuras de iteración
  - Sentencia **while**

```
int cont = 0;
while (cont <= 20) {
    …
    cont++;
}
```

  - Sentencia **do-while**

```
int cont = 0;
do {
    …
    cont++;
} while (cont <= 20);
```

  - Sentencia **for**

```
for (int cont = 0; cont <= 20; cont++) {
    …

}
```

La biblioteca estándar

C++ provee un mecanismo para hacer uso de bibliotecas en nuestros programas.

Para utilizar una biblioteca tenemos que utilizar la directiva del compilador **`include`**. Por ejemplo:

```
#include  <cstdio>

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

    printf("Hola Mundo!");

    return 0;

}
```

```
#include  <iostream>

using  namespace std;

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

    cout << "Hola Mundo!"; //std::cout << "Hola Mundo!";

    return 0;

}
```

La biblioteca estándar

C++ también incluye la biblioteca estándar de C.

Para varias tareas la biblioteca estándar de C++ introduce nuevos tipos de datos (clases) y funciones:

	C	C++
Entrada/Salida por consola	**cstdio:** `scanf, printf, fgets, ...`	**iostream:** `cin, cout, getline`
Manejo de cadenas	**cstring:** `strcpy, strlen, strcat, ...`	**string:** `string (+,size,c_str,...)`
Manejo de archivos	**cstdio:** `fopen, fprintf, fwrite, ...`	**fstream:** `fstream (open,>>,<<,...)`

Entrada/Salida por consola

iostream (C++)

Salida por consola:

```
cout << var1 << cte1 << var2 << cte2 << ... << varn << cten;
```

Entrada por consola:

```
cin >> var1 >> var2 >> ... >> varn;
```

cstdio (C)

Salida por consola:

```
int printf (const char * format, ...)
```

Entrada por consola:

```
int scanf (const char * format, ...)
```

Cadenas

Las cadenas en C son arreglos (o secuencias en memoria) de caracteres que terminan en '`\0`':

```
char <nombre cadena>[<tamaño>];
```

Las cadenas en C++ son objetos del tipo `string`:

```
string <nombre cadena>;
```

Poseen las siguientes ventajas respecto a las cadenas de C:

- Su tamaño es dinámico.
- Soportan el uso de operadores para la comparación (==,<, >, !=, etc) y la concatenación (+ y +=).

Archivos de texto en C++

fstream (C++)

```
fstream <nombre archivo>;
```

Abrir un archivo:

```
<nombre archivo>.open(<ruta archivo>, <modo>);
```

Escribir en el archivo:

```
<nombre archivo> << var1 << cte1 << ... << varn << cten;
```

Leer del archivo:

```
<nombre archivo> >> var1 >> var2 >> ... >> varn;
```

Cerrar un archivo:

```
<nombre archivo>.close();
```

Archivos en C

cstdio (C)

El tipo **FILE** es una estructura con toda la información del archivo: tamaño, posición que estamos accediendo, estado, etc.

Para abrir archivos utilizamos la función fopen:

```
FILE * fopen (const char * filename, const char * mode);
```

La cadena de modo puede contener los siguientes especificadores:

r : abre un archivo existente para lectura

w : crea el archivo o vacía un archivo existente, dejándo listo para escritura.

Y los modificadores:

+ : agrega escritura al modo r o lectura al modo w.

b : **hace que los archivos sean binarios (por defecto son de texto)**

Para cerrar un archivo utilizamos:

```
int fclose (FILE * stream);
```

Archivos de texto en C

Declaración:

```
FILE * <nombre archivo>;
```

Abrir un archivo (no incluir b en el modo):

```
<nombre archivo> = fopen(<ruta archivo>, <modo sin b>);
```

Escribir en el archivo (para cadenas se pude usar `fputs`):

```
int fprintf (FILE * stream, const char * format, ... );
```

Leer del archivo (para cadenas se pude usar `fgets`):

```
int fscanf (FILE * stream, const char * format, ... );
```

Cerrar un archivo:

```
fclose(<nombre archivo>);
```

Archivos binarios en C

Declaración:

```
FILE * <nombre archivo>;
```

Abrir un archivo (se debe incluir el modificador **b** en el modo):

```
<nombre archivo> = fopen(<ruta archivo>, <modo + b>);
```

Escribir en el archivo:

```
size_t fread (void * ptr, size_t size, size_t count,
  FILE * stream);
```

Leer del archivo:

```
size_t fwrite (const void * ptr, size_t size,
  size_t count, FILE * stream);
```

Cerrar un archivo:

```
fclose(<nombre archivo>);
```

Funciones
Pasaje de parámetros (1)

- Por valor o copia:

```
<tipo> <nombre función> (<tipo> <parámetro1>, ...)
```

- Por referencia:

```
<tipo> <nombre función> (<tipo> & <parámetro1>, ...)
```

- El pasaje de parámetros por referencia es un mecanismo exclusivo de C++.
- En C sólo hay parámetros por copia, por lo que estamos obligados a utilizar punteros.

Funciones
Pasaje de parámetros (2)

```
void modificar(int variable)
{
    variable = 10;
    cout<<"modificar - valor variable:"
          << variable << "\n";
}

int main (int argc, char *argv[])
{
    int variable = 2;
    modificar(variable);
    cout<<"main - valor variable: "
          << variable << "\n";
}
```

> modificar – valor variable: 10
> main – valor variable: 2

```
void modificar(int& variable)
{
    variable = 10;
    cout<<"modificar - valor variable:"
          << variable << "\n";
}

int main (int argc, char *argv[])
{
    int variable = 2;
    modificar(variable);
    cout<<"main - valor variable: "
          << variable << "\n";
}
```

> modificar – valor variable: 10
> main – valor variable: 10

Argumentos de los programas (1)

- Todo programa debe implementar una función principal llamada **`main`.**
- Esta función principal puede recibir dos parámetros, opcionales.
- Permiten acceder a los argumentos que se le pasan al programa, al ejecutarlo desde la línea de comandos.

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

  - **`argc`**: es el contador de argumentos.
  - **`argv`**: es un arreglo de strings con cada uno de los argumentos.
- El primer argumento siempres es el nombre del ejecutable de la aplicación.

Argumentos de los programas (2)

- Un programa simple

```
#include <iostream>
using  namespace std;
int main(int argc, char * argv[]) {
    cout << "Cantidad de argumentos: " << argc << "\n";
    cout << "Argumentos: ";
    for (int i = 0; i < argc; i++)
        cout << argv[i] << " ";
    return 0;
}
```

- Un ejemplo de una ejecución del programa anterior:

```
> ejemplo arg1 arg2 arg3
> Cantidad de argumentos: 4
> Argumentos: ejemplo arg1 arg2 arg3
```

Ejemplo 1

Lea desde un archivo la cantidad de llamadas por hora que se realizaron durante el lapso de 30 días y realice las siguientes funciones:

- imprimir el promedio de llamadas por hora de un día dado;
- calcular e imprimir el día con el mayor promedio de llamadas por hora.

Ejemplo 1

```
int main(int argc, char * argv[]) {
    if (argc != 2) {
        imprimirInstrucciones(argv[0]);
    } else {
        // Abrir el archivo
        char * nombreArchivo = argv[1];
        int llamadas[720];
        fstream archivo;
        archivo.open(nombreArchivo, ios::in);
        if (!archivo.is_open()) {
            cout << "Error al abrir el archivo\n";
        } else {
            // Pasamos las llamadas del archivo a un arreglo
            cargarLlamadas(archivo, llamadas);
            archivo.close();

            // Menú de opciones
            char opcion;
            mostrarOpciones();
            cin >> opcion;
            cin.sync();
```

Ejemplo 1

```
while (opcion != '0') {
                switch (opcion) {
                    case '1': {
                        int dia;
                        cout << "Ingrese el día: ";
                        //printf("Ingrese el día: ");
                        cin >> dia;
                        //scanf("%i", &dia);
                        cin.sync();
                        if ((dia > 0) && (dia < 30)) {
                            cout << "El promedio es: " <<
  promedioDia(llamadas, dia) << "\n";
                        } else
                            cout << "Error al ingresar el día
  (debe ser entre 1 y 30)\n";
                    } break;
```

Ejemplo 1

```
                    case '2': {
                        int dia;
                        float promedio;
                        mayorPromedio(llamadas, dia, promedio);
                        cout << "El mayor promedio es " <<
  promedio << " del día " << dia << "\n";
                        //printf("El mayor promedio es %f del día
  %d\n", promedio, dia);
                    } break;
                    case '0': break;
                    default: {
                        cout <<"Opción incorrecta\n";
                    } break;
                }
                mostrarOpciones();
                cin >> opcion;
                cin.sync();
            }
        }
    }
}
```

Ejemplo 1

```
void cargarLlamadas(fstream & archivo, int llamadas[]) {
    int i = 0;
    while (!archivo.eof()) {
        archivo >> llamadas[i];
        i++;
    }
}


float promedioDia(int llamadas[], int dia) {
    float promedio = 0;
    int horaFin = dia * 24;
    for (int hora = (dia-1) * 24; hora < horaFin; hora++) {
        promedio = promedio + llamadas[hora];
    }
    promedio = promedio / 24;
    return promedio;
}
```

Ejemplo 1

```
void mayorPromedio(int llamadas[], int & mayorDia, float &
   mayorPromedio) {
     float promedio;
     mayorPromedio = 0;
     mayorDia = 1;
     for (int dia = 1; dia <= 30; dia++) {
         promedio = promedioDia(llamadas, dia);
         if (promedio > mayorPromedio) {
             mayorPromedio = promedio;
             mayorDia = dia;
         }
     }
}
```

Ejemplo 2

Dado un archivo de texto, reemplazar todas las ocurrencias de una cadena determinada por otra. Guardar el resultado en un nuevo archivo de texto.

Ejemplo 2

```
int main(int argc, char * argv[]) {
    if (argc != 3) {
        imprimirInstrucciones(argv[0]);
    } else {

        char * nombreArchivoOrigen = argv[1];
        char * nombreArchivoDestino = argv[2];

        fstream archivoOrigen(nombreArchivoOrigen, ios::in);
        if (!archivoOrigen.is_open()) {
            cout << "Error al abrir el archivo origen\n";
        } else {
            fstream archivoDestino(nombreArchivoDestino,
  ios::out);
            if (!archivoDestino.is_open()){
                cout << "Error al abrir el archivo destino\n";
            }else {
```

Ejemplo 2

```
                string palabraR, palabraN;
                 cout << "Ingrese la palabra a reemplazar: \n";
                 cin >> palabraR;
                 cin.sync();
                 cout << "Ingrese la nueva palabra: \n";
                 cin >> palabraN;
                 cin.sync();
                 reemplazarPalabras(archivoOrigen, archivoDestino,
  palabraR, palabraN);
                 archivoDestino.close();
              }
              archivoOrigen.close();
         }
    }
    return 0;
}
```

Ejemplo 2

```
void reemplazarPalabras(fstream & archivoOrigen, fstream &
   archivoDestino, string palabraR, string palabraN) {
     while (!archivoOrigen.eof()) {
          string linea;
          getline(archivoOrigen, linea);
          size_t pos = linea.find(palabraR, 0);
          while (pos != string::npos) {
               linea = linea.replace(pos, palabraR.size(),
   palabraN);
               pos = linea.find(palabraR, pos + palabraN.size());
          }
          archivoDestino << linea << '\n';
     }
}
```

Enlaces útiles

- Referencia de la biblioteca estándar de C++:

  http://www.cplusplus.com/reference

- Pensar en C++ (proyecto de traducción del libro “Thinking in C++” de Bruce Eckel):

  http://arco.inf-cr.uclm.es/~david.villa/pensarC++.html

Federico Améndola

Consultas: laboratorio.ayda@alumnos.exa.unicen.edu.ar