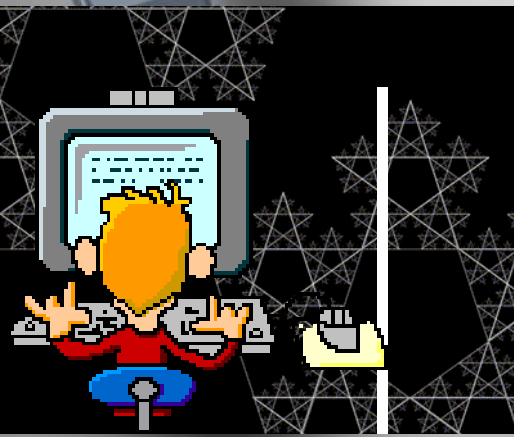




Programación I

Ing. Carlos R. Rodríguez

***Tecnicatura
Universitaria en
Programación***



UNIVERSIDAD TECNOLÒGICA NACIONAL
Facultad Regional Mendoza



Como quizás haya notado, el código que se escribe en el entorno **no es el código que se compila finalmente**.

Antes de ser compilado el código es “**preprocesado**” por un traductor que – en nuestro caso – traduce desde el **pseudocódigo a C/C++**.

Ese programa se llama **cpp**:

```
carlos@Lenovo-G50-80:~/Zinjai/2018/Caballo$ cpp --help
```

```
Usage: cpp [options] file...
```

```
Options:
```

<code>-pass-exit-codes</code>	Exit with highest error code from a phase.
<code>--help</code>	Display this information.

```
Terminal
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda

carlos@lenovo-G50-80:~/Zinjai/2018/Caballo$ cpp --help
Usage: cpp [options] file...
Options:
  -pass-exit-codes      Exit with highest error code from a phase.
  --help                Display this information.
  -target help          Display target specific command line options.
  --help={common|optimizers|param|target|warnings|[^]}([joined|separate|undocumente
d)][,...].
                        Display specific types of command line options.
                        (Use '-v --help' to display command line options of sub-processes).
  --version             Display compiler version information.
  -dumpspecs            Display all of the built in spec strings.
  -dumpversion          Display the version of the compiler.
  -dumpmachine          Display the compiler's target processor.
  -print-search-dirs    Display the directories in the compiler's search path.
  -Xpreprocessor <arg> Pass <arg> on to the preprocessor.
  -Xlinker <arg>       Pass <arg> on to the linker.
  -save-temps           Do not delete intermediate files.
  -save-temps=<arg>    Do not delete intermediate files.
```



Opc. del preprocesador:

- **-I**, que establece el directorio asumido de directivas de inclusión
- **-o**, que establece el archivo de salida.
- **-P**, que anula la salida de los números de línea y del nombre del archivo, dejando el código limpio.

Elegimos un archivo .cpp:

```
carlos@Lenovo-G50-80:~/Zinjai/2018/04 - Subprogramas$ ls -la
total 156
drwxrwxr-x  2 carlos carlos  4096 oct 11 08:55 .
drwxrwxr-x 15 carlos carlos  4096 oct 17 12:10 ..
-rwxrwxr-x  1 carlos carlos 145112 oct 11 08:55 EjInterc.bin
-rw-rw-r--  1 carlos carlos   772 oct 11 08:55 EjInterc.cpp
carlos@Lenovo-G50-80:~/Zinjai/2018/04 - Subprogramas$ cat EjInterc.cpp
// Archivo de traducción de pseudocódigo a C++
#include <program1.h>
/**
 *   Enunciado: Dados 2 valores, intercambiarlos.
 */
procedimiento intercambia(real porRef, real porRef);

principal                                     // Unidad
ama principal
real uno,dos;
limpiar;                                     // Limpia
lla.
leerM(uno,"Valor 1:");
leerM(dos,"Valor 2:");
intercambia(uno,dos);
mostrar << uno << tabulado << dos << salto;
pausa;                                     // Pausa
finalizar.
finPrincipal                                // Fin de
e program principal.

procedimiento intercambia(real porRef x, real porRef y) {
    real z = x;
    x = y;
    y = z;
}
carlos@Lenovo-G50-80:~/Zinjai/2018/04 - Subprogramas$
```



Con “`cpp -I ../ -o salida.i -P EjInterc.cpp`”, el resultado será el código ya traducido a C/C++ y derivado a `salida.i`.

Con `cat salida.i`:

```
if(difLongi > 0) {
    resul = relleno.substr(0,difLongi) + resul;
}
return(resul);
}
inline int aEntero(std::string cad) {
    int resul
        ,longi = cad.length();
    if(longi > 0) {
        if(cad.substr(0,1) == "-") {
            resul = -aEntero(cad.substr(1));
        } else if(longi > 1) {
            resul = 10 * aEntero(cad.substr(0,longi-1)) + aEntero(cad.substr(longi-1,1));
        } else {
            std::string cadDigitos = "0123456789";
            resul = cadDigitos.find(cad);
        }
    } else {
        resul = 0;
    }
}
return(resul);
}
void intercambia(float &,float &);
int main(int argc, char *argv[]) {
    float uno,dos;
    system("clear");;
    std::cout << "Valor 1:";cin >> uno;
    std::cout << "Valor 2:";cin >> dos;
    intercambia(uno,dos);
    std::cout << uno << (char) 9 << dos << endl;
    {std::cout << "En pausa. <Escape> para continuar...";while (leeTec(false) != 27);}
    ;
    return 0;}
void intercambia(float & x, float & y) {
    float z = x;
    x = y;
    y = z;
}
carlos@Lenovo-G50-80:~/Zinjai/2018/04 - Subprogramas$ cpp -I ../ -o salida.i -P EjInterc.cpp
```



Este ejemplo nos muestra cómo traducir desde pseudocódigo a un lenguaje de programación *si le damos las reglas de traducción.*

Se puede editar el archivo program1.h

```
if(difLongi > 0) {
    resul = relleno.substr(0,difLongi) + resul;
}
return(resul);
}
inline int aEntero(std::string cad) {
int resul
    ,longi = cad.length();
if(longi > 0) {
    if(cad.substr(0,1) == "-") {
        resul = -aEntero(cad.substr(1));
    } else if(longi > 1) {
        resul = 10 * aEntero(cad.substr(0,longi-1)) + aEntero(cad.substr(longi-1,1));
    } else {
        std::string cadDigitos = "0123456789";
        resul = cadDigitos.find(cad);
    }
} else {
    resul = 0;
}
return(resul);
}
void intercambia(float &,float &);
int main(int argc, char *argv[]) {
float uno,dos;
system("clear");;
std::cout << "Valor 1:";cin >> uno;
std::cout << "Valor 2:";cin >> dos;
intercambia(uno,dos);
std::cout << uno << (char) 9 << dos << endl;
{std::cout << "En pausa. <Escape> para continuar...";while (leeTec(false) != 27);}
;
return 0;}
void intercambia(float & x, float & y) {
float z = x;
x = y;
y = z;
}
carlos@Lenovo-G50-80:~/Zinjai/2018/04 - Subprogramas$ g++ -I ../ -o salida.i -P Interc.cpp
```