

Introducción a la Programación con C++ Ejercicios

A. Garrido y J. Martínez-Baena

eug EDITORIAL
UNIVERSIDAD
DE GRANADA



© A. GARRIDO Y J. MARTÍNEZ-BAENA
© UNIVERSIDAD DE GRANADA
© Fotografías y cubierta: ANTONIO GARRIDO
INTRODUCCIÓN A LA PROGRAMACIÓN CON C++: Ejercicios
ISBN: 978-84-338-5924-2.
Edita: Editorial Universidad de Granada.
Campus Universitario de Cartuja. Granada.

Printed in Spain

Impreso en España.

Cualquier forma de reproducción, distribución, comunicación pública o transformación de esta obra sólo puede ser realizada con la autorización de sus titulares, salvo excepción prevista por la ley.

Índice general



1	Expresiones en C++	1
1.1	Introducción	1
1.1.1	Tipos de datos y operaciones	1
1.2	Tipos de datos numéricos	1
1.2.1	Entero vs real	2
1.2.2	Variables	2
1.2.3	Constantes	3
1.2.4	Funciones de biblioteca	3
1.2.5	Errores de aproximación de números reales	4
1.3	Tipo carácter	5
1.3.1	E/S de un carácter	5
1.3.2	Funciones de biblioteca para caracteres	5
1.4	Expresiones complejas	6
1.5	Conversiones explícitas	6
1.6	Ejercicios adicionales	6
2	La estructura de selección	9
2.1	Introducción	9
2.2	La instrucción if simple	9
2.3	La instrucción if/else	9
2.3.1	La instrucción if/else anidada	10
2.4	Condiciones compuestas: operadores lógicos	11
2.4.1	Evaluación en corto	12
2.5	La instrucción switch	12
2.6	Booleanos y enteros	13
2.7	Ejercicios adicionales	13
3	La estructura de iteración	17
3.1	Introducción	17

3.2	Patrones de diseño de bucles	17
3.2.1	El bucle do-while	17
3.2.2	El bucle while	18
3.2.3	Soluciones habituales con while	19
3.2.4	¿Repetir mientras o repetir hasta?	20
3.3	El bucle for	21
3.3.1	Inicialización, condición e incremento	21
3.3.2	Ejecutando un cierto número de veces	21
3.3.3	Omitiendo expresiones	22
3.4	Anidamiento de bucles	22
3.5	Variables lógicas y condiciones compuestas	23
3.6	Casos generales y casos límite	24
3.7	Algunos errores comunes	24
3.7.1	Funciona, pero mejor no hacerlo	25
3.8	Ejercicios adicionales	26
4	Vectores y matrices	29
4.1	Introducción	29
4.2	El tipo vector	29
4.2.1	Operaciones con vectores	30
4.2.2	Vectores de tamaño variable	32
4.2.3	Vectores y lectura adelantada	32
4.3	Vectores de vectores	32
4.3.1	Matrices	33
4.4	Ordenación y búsqueda	33
4.5	Ejercicios adicionales	34
5	Cadenas de caracteres	37
5.1	Introducción	37
5.2	El tipo string	37
5.3	Operaciones básicas con string	38
5.3.1	E/S de string	38
5.3.2	Tamaño y acceso a los caracteres	38
5.3.3	E/S combinada con otros tipos	39
5.4	Más operaciones con string	41
5.4.1	Concatenación de cadenas	41
5.4.2	Comparación de cadenas	41
5.4.3	Extracción de subcadenas	42
5.4.4	Borrado de subcadenas	42
5.4.5	Inserción de subcadenas	42
5.4.6	Reemplazo de subcadenas	43
5.4.7	Búsqueda en cadenas	43
5.5	Ejercicios adicionales	44
6	Funciones	47
6.1	Introducción	47
6.1.1	Notación	47

6.2	Parametrización de las funciones	48
6.2.1	El paso por valor	48
6.2.2	Las funciones de tipo void	49
6.2.3	El paso por referencia	50
6.2.4	Conversiones	52
6.3	Diseño de funciones	52
6.3.1	La separación entre la E/S y los cálculos	52
6.3.2	Diseño descendente (top-down)	54
6.4	Gestión de errores	57
6.4.1	Devolución de un valor de error	58
6.4.2	Errores graves irrecuperables	58
6.4.3	Precondiciones y postcondiciones	58
6.5	Valores por defecto y sobrecarga	60
6.5.1	Parámetros con valores por defecto	60
6.5.2	Sobrecarga de funciones	60
6.6	Ejercicios adicionales	61
7	Funciones con cadenas y vectores	65
7.1	Introducción	65
7.2	Ejercicios de funciones y vectores	65
7.3	Ejercicios de funciones y cadenas	67
7.4	Ejercicios adicionales	68
8	Funciones recursivas	73
8.1	Introducción	73
8.2	El caso general y el caso base	73
8.3	Funciones recursivas con varios puntos de salida	74
8.4	Múltiples casos base y/o generales	74
8.5	Ejercicios adicionales	74
9	Estructuras y pares	77
9.1	Introducción	77
9.2	Estructuras en C++	77
9.2.1	Estructuras y funciones	77
9.2.2	Anidamiento de estructuras	78
9.2.3	Estructuras y otros datos compuestos	78
9.3	El tipo par de la STL	79
9.4	Ejercicios adicionales	80
10	Flujos de Entrada/Salida	83
10.1	Introducción	83
10.1.1	Operaciones básicas de transferencia	84
10.2	Ficheros	86
10.2.1	Apertura de ficheros	86
10.2.2	Añadiendo datos a ficheros	87
10.2.3	Cierre de ficheros	88
10.3	Detección del fin del flujo	88
10.4	Control de errores en los flujos	89
10.4.1	Control de errores en ficheros	89

10.5	Reutilización de flujos	90
10.6	Otras operaciones sobre flujos	90
10.7	Flujos y funciones	91
10.7.1	Copia de flujos	92
10.7.2	Uso de flujos "genéricos"	92
10.8	Algunas recomendaciones finales	92
10.8.1	Lectura adelantada y flujos	92
10.8.2	Ventajas e inconvenientes de "no-op"	93
10.8.3	Evitar la lectura parcial de objetos	94
10.9	Ejercicios adicionales	95
A	Generación de números aleatorios	97
A.1	Introducción	97
A.2	Números pseudoaleatorios	97
B	Redirección de Entrada/Salida	101
B.1	Introducción	101
B.2	Ejecuciones con muchos datos	103
B.3	Almacenamiento externo	104
B.3.1	Añadiendo en lugar de sustituyendo	104
B.4	Redirección de E/S simultánea	104
B.5	Redirección de cerr	105
B.6	Encauzamiento	105
C	Tablas	107
C.1	Tabla ASCII	107
C.2	Operadores C++	108
C.3	Palabras reservadas de C89, C99, C11, C++ y C++11	109
	Bibliografía	111
	Índice alfabético	113

Prólogo



La enseñanza de la programación de ordenadores es un tema ampliamente discutido, no sólo por la dificultad de enseñar una serie de conceptos y habilidades que pueden resultar bastante complicados, sino también por la rapidez con que cambia la tecnología, dando lugar no sólo a nuevos lenguajes de programación, sino también a nuevas formas de abordarla.

El rápido desarrollo de las tecnologías disponibles para programar ordenadores ha tenido un efecto directo sobre la enseñanza, ya que ha incrementado el conjunto de habilidades que se supone que debe poseer un programador. Aun así, no podemos olvidar que los fundamentos de la programación son los mismos.

Un curso de fundamentos de programación presenta conceptos relativamente simples. Sin embargo, para un estudiante que comienza en la programación, los conceptos que se discuten pueden resultar bastante abstractos y difíciles de asimilar. Esta dificultad puede verse compensada en gran medida con una metodología orientada a la práctica. Por ello, tan importante como disponer de un buen manual de referencia sobre el lenguaje de estudio —C++ en nuestro caso— lo es disponer de un buen libro de apoyo en la realización de prácticas.

Precisamente, ese es el objetivo de este libro: ofrecer al estudiante un documento de trabajo para facilitar el aprendizaje aplicando de forma práctica los conceptos aprendidos en clase o en otros libros más teóricos sobre C++ y la STL. En concreto, nos permite ofrecer un guión de prácticas relacionadas con el libro *Fundamentos de Programación con la STL* (Garrido[5]).

Organización del cuaderno de prácticas

Para compensar la falta de experiencia del estudiante, los ejercicios del cuaderno se diseñan para que incidan especialmente en situaciones habituales en la práctica de programación. De hecho, un programador con experiencia consideraría simples muchos de los problemas que se proponen. Esta simplicidad es consecuencia de que tiene asimilados muchos esquemas de solución —patrones de diseño— que ha resuelto repetidamente.

Los temas se plantean con una serie de ejercicios que hagan explícitas estas formas de solución. Adicionalmente, en algunos casos se incluye un pequeño resumen de aspectos teóricos importantes o una discusión sobre la forma en que cierto diseño facilita la solución de problemas. Cada sección suele estar diseñada para practicar algún aspecto concreto. Además, parte de los ejercicios permiten profundizar en algunos puntos de especial relevancia y que suelen pasar desapercibidos por los estudiantes.

El libro está diseñado para comenzar desde cero el aprendizaje de la programación. Como se puede comprobar con un simple vistazo al índice de capítulos, se trata de programación estructurada y modular. Aunque C++ es un lenguaje multiparadigma, este cuaderno se ha creado para un curso de fundamentos; el objetivo es que el estudiante acabe con una base sólida que permita el abordaje de curso más avanzados con garantías de éxito. Más concretamente, se llevan a cabo prácticas que incorporan contenidos sobre:

- Programas simples con expresiones en C++.
- Estructuras de control, incluyendo la selección y la iteración.
- Diseño de funciones.
- Tipos compuestos homogéneos: vectores, matrices y cadenas.
- Tipos compuestos heterogéneos: estructuras y pares.
- Problemas simples de E/S con ficheros de texto.

Finalmente, el documento incluye apéndices con información relevante para realizar los guiones. En concreto, podrá usarlos para consultar:

- La *generación de números aleatorios*. Es un tema que generalmente no se aborda directamente en las clases de teoría, sino que se supone se practicará cuando se desarrollen programas que generan valores aleatorios. Sin embargo, es un tema cuyo contenido teórico es muy relevante para poderlo usar adecuadamente. En lugar de dar una breve especificación de las funciones que ofrece el lenguaje, se incluye una exposición más detallada con el fin de que el estudiante no sólo lo use, sino de que entienda por qué funciona.
- La *redirección de E/S*. Aunque inicialmente se plantea como un tema relacionado con la E/S, se ha dejado para un apéndice puesto que es un contenido transversal para todo el curso. Desde los primeros ejercicios ya podemos aprovechar este redireccionamiento para ejecuciones rápidas desde la línea de órdenes.
- *Tablas* relacionadas con el lenguaje. En la práctica, es muy útil disponer de tablas que incluyen detalles sobre palabras reservadas, operadores del lenguaje, código *ASCII*, etc.

Entorno de programación

El curso está basado en el estándar más extendido de C++ —el del 98/03— aunque sigue siendo eficaz si está interesado en programar con el último estándar, ya que los ejercicios y las discusiones siguen siendo igualmente válidos. Desde este punto de vista, puede usar cualquier versión, aunque se recomienda al menos la del 98. En la fecha que estamos, seguro que cualquier compilador cumple con esta condición.

No es necesario usar ningún compilador o entorno de programación concreto. Se ha desarrollado para que el estudiante sea libre de optar por las herramientas que le sean más cómodas, siendo casi cualquier compilador estándar una buena opción para practicar. Sólo es necesario un editor de texto y un compilador.

Sin embargo, siendo un primer contacto con la programación, es recomendable que el estudiante pueda centrarse en los detalles del lenguaje y deje la dificultad de la gestión de proyectos para más adelante. Por ello, se recomienda algún entorno que facilite la compilación y ejecución. En nuestro caso, hemos usado *Code::Blocks*, que se ajusta perfectamente a los objetivos del curso.

Agradecimientos

Los autores desean agradecer a todos sus alumnos por haber contribuido —aun sin saberlo— a la creación de este cuaderno, especialmente a los que “*se equivocan*” porque precisamente ellos permiten encontrar los puntos débiles en el proceso de enseñar a programar.

Esta sección no puede terminar sin un agradecimiento a todos los que con su trabajo desinteresado han contribuido al software libre, creando sistemas y herramientas gratuitas que nos permiten por un lado crear este documento —íntegramente desarrollado con software libre— y por otro, mucho más interesante, ofrecer una solución totalmente gratuita para realizar este curso.

A. Garrido y J. Martínez-Baena
Mayo de 2016.

1

Expresiones en C++

Introducción	1
Tipos de datos y operaciones	
Tipos de datos numéricos	1
Entero vs real	
Variables	
Constantes	
Funciones de biblioteca	
Errores de aproximación de números reales	
Tipo carácter	5
E/S de un carácter	
Funciones de biblioteca para caracteres	
Expresiones complejas	6
Conversiones explícitas	6
Ejercicios adicionales	6

1.1 Introducción

En este capítulo se presenta el estudio de las expresiones básicas en C++. En concreto se estudiarán los siguientes aspectos:

- Tipos de datos básicos y sus operaciones.
- Expresiones.
- Funciones de biblioteca.

A pesar de la amplia gama de tipos de datos que ofrece el lenguaje, no se estudiarán en profundidad, ya que por un lado prácticamente todos los programas que se van a crear pueden resolverse con los tipos básicos, y por otro no aportan un valor añadido en un curso introductorio. Más adelante, cuando ya se posea una base de programación y se requieran soluciones más específicas y optimizadas, será fácil incorporar todos esos conocimientos.

1.1.1 Tipos de datos y operaciones

Antes de empezar, es importante que el alumno entienda que un programa puede manejar una amplia variedad de información y, para ello, incluye la posibilidad de almacenar y operar con distintos tipos de datos. Un tipo de dato implica:

- Una representación. La información la almacena y maneja un ordenador que sólo sabe trabajar con ceros y unos. Cualquier dato que quiera introducir en un ordenador deberá transformarse a ceros y unos. Es inevitable que los datos se representen siempre en ese lenguaje binario.
Como consecuencia de esa representación, los tipos de datos que maneja el ordenador tendrán ciertas características muy concretas. Por ejemplo, debido a la limitación en la cantidad de memoria que posee el ordenador, se puede limitar el tamaño de los datos. Un caso concreto sería guardar un dato de tipo entero, que sabemos que tiene un número infinito de posibilidades, pero que en nuestros programas estará limitado a un rango finito de valores.
- Un conjunto de operaciones. El lenguaje nos ofrece, para cada tipo, un conjunto de operaciones asociado. Estas operaciones nos permiten obtener nuevos datos como resultado de operar con uno o más operandos. Por ejemplo, podemos obtener un nuevo entero como resultado de la suma de dos datos del mismo tipo.

En este capítulo vamos a introducir la creación de expresiones donde aparecen datos de distintos tipos relacionados mediante un conjunto de operadores. Para escribir nuestros programas, es fundamental entender perfectamente cómo se evalúan estas expresiones y qué resultados se obtienen.

1.2 Tipos de datos numéricos

En esta sección vamos a introducir algunos ejemplos de programas simples en C++ con la intención de que el lector asimile más fácilmente las consecuencias prácticas de los conceptos y reglas que se estudian de forma más detallada en clase de teoría.

Aunque durante la exposición se incluirán algunos contenidos para recordar y enfatizar algunos detalles clave, es recomendable que revise los contenidos teóricos antes trabajar este tema. Los ejercicios se han diseñado para hacer que el lector infiera los conceptos fundamentales del tema, evitando memorizar reglas e invitando a razonar sobre los contenidos que se presentan.