STUDENT PRICE BOOK CENTER

DEV-C++ PROGRAMMING: BEGINNER

Topics Include:
Dev-C++ Basics, C++ Basics
Control Structures
Functions in C++
Arrays and Strings

คำนำ

หนังสือ *"การเรียนรู้ภาษา C++ ด้วย Dev-C++"* เล่มนี้ ได้รับการออกแบบมาเพื่อตอบสนองความต้องการของผู้เริ่มต้นที่ไม่มีพื้นฐานการเขียนโปรแกรมมาก่อน รวมถึงผู้ที่ต้องการวางรากฐานที่มั่นคงในการเขียนโปรแกรมด้วยภาษา C++ โดยเน้นการเรียนรู้ผ่านโปรแกรม Dev-C++ ซึ่งใช้งานง่าย เหมาะสำหรับผู้เริ่มต้นทุกระดับ

ในระดับ *ผู้เริ่มต้น (Beginner Level)* ผู้อ่านจะได้รับความรู้และทักษะที่จำเป็นต่อการเริ่มต้นพัฒนาโปรแกรม C++ อย่างเป็นระบบ โดยเนื้อหาครอบคลุม 5 หัวข้อหลัก ดังนี้:

1. **พื้นฐานการใช้งาน Dev-C++**
   เริ่มต้นจากการติดตั้งและตั้งค่าโปรแกรม Dev-C++ การสร้างและคอมไพล์โปรเจกต์แรก พร้อมทำความเข้าใจส่วนประกอบของโปรแกรม C++ เช่น main() และ #include รวมถึงเขียนโปรแกรมง่าย ๆ อย่าง *Hello World!*
2. **พื้นฐานภาษา C++**
   เรียนรู้การประกาศตัวแปรและชนิดข้อมูลพื้นฐาน เช่น int, float, char, bool รวมถึงการรับ-แสดงค่าด้วย cin และ cout และการใช้ตัวดำเนินการทางคณิตศาสตร์และลอจิก เช่น +, -, *, /, ++, --
3. **โครงสร้างควบคุมโปรแกรม**
   ฝึกการเขียนโปรแกรมโดยใช้คำสั่งควบคุมทิศทางของโปรแกรม เช่น if, else if, else, switch-case และการวนลูปด้วย for, while, do-while พร้อมคำสั่งพิเศษอย่าง break และ continue
4. **ฟังก์ชันพื้นฐาน**
   อธิบายการสร้างและเรียกใช้งานฟังก์ชัน ทั้งแบบ void และแบบที่ส่งค่ากลับด้วย return พร้อมการส่งค่าด้วยพารามิเตอร์ เพื่อเสริมสร้างความเข้าใจในการแยกส่วนของโปรแกรม
5. **อาเรย์ (Array) และสตริง (String)**
   แนะนำการใช้อาเรย์ทั้งแบบ 1 มิติและ 2 มิติ การวนลูปผ่านข้อมูลในอาเรย์ และการใช้งานสายอักขระด้วย string จากไลบรารี <string> ซึ่งเป็นพื้นฐานสำคัญสำหรับการจัดการข้อมูลจำนวนมากและข้อความ

เนื้อหาในแต่ละบทถูกออกแบบให้มีตัวอย่างโค้ดพร้อมคำอธิบายโดยละเอียด และแบบฝึกหัดเพื่อเสริมทักษะการคิดวิเคราะห์และการลงมือปฏิบัติจริง ผู้อ่านสามารถเรียนรู้และพัฒนาตนเองไปทีละขั้นได้อย่างมั่นใจ

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1
พื้นฐานการใช้งาน Dev-C++
(Dev-C++ Basic)

เนื้อหา

- Dev-C++ คืออะไร?
- ประวัติของ Dev-C++
- ขีดความสามารถของ Dev-C++
- แนวโน้มในปัจจุบันของ Dev-C++
- ขั้นตอนการติดตั้งและตั้งค่า Dev-C++
- ขั้นตอนการสร้างและคอมไพล์โปรเจกต์แรกใน Dev-C++
- การติดตั้งและตั้งค่า Dev-C++ (2)
- ส่วนประกอบของโปรแกรม C++ เบื้องต้น
- การเขียนโปรแกรม Hello World! ใน Dev-C++

บทนำ

ภาษา C++ เป็นหนึ่งในภาษาการเขียนโปรแกรมระดับสูงที่มีความยืดหยุ่นและทรงพลัง ได้รับความนิยมอย่างแพร่หลายในวงการวิศวกรรมซอฟต์แวร์ การพัฒนาระบบฝังตัว และการสร้างแอปพลิเคชันที่ต้องการประสิทธิภาพสูง หนังสือเล่มนี้มุ่งเน้นการปูพื้นฐานให้ผู้อ่านสามารถเริ่มต้นเรียนรู้ภาษา C++ ได้อย่างเป็นระบบ โดยเริ่มจากการใช้งาน Dev-C++ ซึ่งเป็นโปรแกรมพัฒนา (IDE) ที่เหมาะสมกับผู้เริ่มต้น มีอินเทอร์เฟซที่ใช้งานง่ายและสนับสนุนการเขียนโปรแกรม C++ ได้อย่างครบถ้วน

ในบทแรก ผู้อ่านจะได้เรียนรู้ขั้นตอนการติดตั้งและตั้งค่าโปรแกรม Dev-C++ ตั้งแต่การดาวน์โหลดไฟล์ติดตั้ง การเลือกเวอร์ชันที่เหมาะสม ไปจนถึงการตั้งค่าพื้นฐานที่จำเป็นในการเริ่มเขียนโปรแกรม ทั้งนี้เพื่อให้เครื่องมือที่ใช้งานพร้อมสำหรับการพัฒนาโปรแกรมอย่างราบรื่น โดยไม่ติดขัดทางเทคนิค

ต่อจากนั้น ผู้อ่านจะได้ฝึกการสร้างโปรเจกต์ใหม่ใน Dev-C++ และทดลองคอมไพล์โปรแกรม C++ แรกของตนเอง พร้อมทำความเข้าใจองค์ประกอบหลักของโปรแกรม C++ เช่น ฟังก์ชัน main() ซึ่งเป็นจุดเริ่มต้นของการทำงานของโปรแกรม และไฟล์ส่วนหัว (header files) ที่มีบทบาทในการประกาศฟังก์ชันและการนำเข้าไลบรารีต่าง ๆ ที่จำเป็น

สุดท้าย เพื่อสร้างความมั่นใจและแรงจูงใจในการเรียนรู้ ผู้อ่านจะได้ทดลองเขียนโปรแกรมง่าย ๆ อย่าง “Hello World!” ซึ่งถือเป็นจุดเริ่มต้นสำคัญของการเขียนโปรแกรมในหลายภาษา โค้ดโปรแกรมนี้จะช่วยให้ผู้อ่านเข้าใจรูปแบบพื้นฐานของภาษา C++ และกระบวนการทำงานตั้งแต่เขียนโปรแกรมไปจนถึงการแสดงผลลัพธ์บนหน้าจอ ทั้งนี้เพื่อให้ผู้อ่านพร้อมสำหรับการเรียนรู้ในระดับที่ลึกขึ้นในบทถัดไป.

□ Dev-C++ คืออะไร?

Dev-C++ เป็น **Integrated Development Environment (IDE)** สำหรับภาษา **C และ C++** ที่ได้รับความนิยมมาก โดยเฉพาะในหมู่นักเรียน นักศึกษา และผู้เริ่มต้นเรียนรู้การเขียนโปรแกรมภาษา C/C++ เนื่องจากใช้งานง่ายและไม่ซับซ้อน

□ คุณสมบัติหลักของ Dev-C++

- สนับสนุนภาษา **C และ C++**
- ใช้งานกับ **MinGW (Minimalist GNU for Windows)** compiler
- มี **syntax highlighting** และ **code completion**
- รองรับ **การ debug เบื้องต้น**
- มีระบบสร้างโปรเจกต์ และจัดการไฟล์โค้ดหลายไฟล์ในโครงการเดียว
- ใช้พื้นที่น้อย ติดตั้งง่าย

□ เวอร์ชันที่รู้จักกันมาก:

- **Dev-C++ ของ Bloodshed** (รุ่นดั้งเดิม): หยุดพัฒนาไปนานแล้ว
- **Embarcadero Dev-C++** (รุ่นใหม่): ได้รับการพัฒนาและอัปเดตให้รองรับ Windows 10/11 และมาตรฐาน C++ ล่าสุด (เช่น C++11, C++14, C++17)

□ จุดเด่น:

- เหมาะสำหรับผู้เริ่มต้น
- ติดตั้งง่าย ใช้งานไม่ยุ่งยาก
- ฟรีและโอเพ่นซอร์ส

⚠□ ข้อจำกัด:

- ฟีเจอร์น้อยเมื่อเทียบกับ IDE สมัยใหม่อย่าง Visual Studio, Code::Blocks, หรือ CLion
- ไม่เหมาะกับโปรเจกต์ขนาดใหญ่หรือการพัฒนาเชิงพาณิชย์ในระดับสูง
- ไม่มี IntelliSense หรือระบบ debug ที่ซับซ้อน

หากคุณต้องการตัวอย่างโค้ดใน Dev-C++ หรือคำแนะนำการใช้งาน เช่น การติดตั้ง การเขียนโปรแกรมแรก (Hello World) หรือการใช้ฟังก์ชันต่าง ๆ ก็สามารถขอได้เลยครับ!
ได้เลยครับ! ต่อไปนี้คือ **ประวัติของ Dev-C++** ตั้งแต่เริ่มต้นจนถึงเวอร์ชันล่าสุด:

□ ประวัติของ Dev-C++

□ จุดเริ่มต้น: Bloodshed Dev-C++

- **Dev-C++** ถูกสร้างขึ้นครั้งแรกโดยนักพัฒนาชาวอิตาลีชื่อ **Colin Laplace** ภายใต้ชื่อทีมว่า **Bloodshed Software**
- เริ่มต้นพัฒนาประมาณปี **1998–1999**
- เป็น IDE ที่ออกแบบมาเพื่อภาษา **C/C++** โดยใช้ **MinGW (Minimalist GNU for Windows)** เป็น compiler หลัก
- กลายเป็นที่นิยมอย่างมากในหมู่นักเรียนและผู้เริ่มต้น เพราะเป็นโปรแกรมฟรี ขนาดเล็ก ใช้งานง่าย

□ จุดหยุดพัฒนา

- การพัฒนาโดย Bloodshed หยุดไปช่วงปี **2005–2006**
- รุ่นสุดท้ายของ Bloodshed Dev-C++ คือ **Dev-C++ 4.9.9.2**
- แม้จะหยุดพัฒนา แต่ก็ยังมีผู้ใช้งานจำนวนมากใช้ต่อเนื่อง

□ การคืนชีพ: Orwell Dev-C++

- ในปี **2011** นักพัฒนาชื่อ **Johan Mes (ใช้นามแฝงว่า Orwell)** ได้ทำการ "fork" โครงการและพัฒนา Dev-C++ ขึ้นมาใหม่ในชื่อ **Orwell Dev-C++**
- ปรับปรุง:
  - รองรับมาตรฐาน C++11
  - อัปเดต MinGW compiler
  - ปรับ UI และแก้บั๊กจำนวนมาก
- รุ่นที่รู้จักกันดีจาก Orwell คือเวอร์ชัน **5.x**

□ เวอร์ชันล่าสุด: Embarcadero Dev-C++

- ปี **2020** บริษัท **Embarcadero Technologies** (บริษัทเดียวกับผู้พัฒนา Delphi) ได้เข้ามาสานต่อ Dev-C++ อีกครั้ง
- พัฒนาและรีแบรนด์ในชื่อว่า **Embarcadero Dev-C++**
- จุดเด่นของเวอร์ชันนี้:
  - รองรับ **C++17**

- สนับสนุนการทำงานบน **Windows 10/11**
- รองรับ **DPI scaling** สำหรับจอความละเอียดสูง
- มีระบบการจัดการโปรเจกต์ดีขึ้น และปรับปรุง UI

สรุปเวอร์ชันหลัก

ยุค	ผู้พัฒนา	ชื่อเวอร์ชัน	ปีที่พัฒนา	สถานะ
เริ่มต้น	Bloodshed (Colin Laplace)	Dev-C++ 4.x	~1999–2005	หยุดพัฒนา
กลาง	Orwell (Johan Mes)	Dev-C++ 5.x	2011–2015	หยุดพัฒนา
ปัจจุบัน	Embarcadero	Embarcadero Dev-C++	2020–ปัจจุบัน	ยังพัฒนาอยู่

หมายเหตุเพิ่มเติม

- ปัจจุบันแม้ Dev-C++ จะยังมีผู้ใช้งานอยู่ แต่ IDE สมัยใหม่อย่าง **Visual Studio Code**, **Code::Blocks**, หรือ **CLion** ก็เริ่มมาแทนที่ในหลายภาคการศึกษา
- อย่างไรก็ตาม **Dev-C++** ก็ยังเหมาะมากสำหรับผู้ที่ต้องการ IDE ขนาดเล็ก ทำงานรวดเร็ว และไม่ซับซ้อน

ขีดความสามารถของ Dev-C++

แม้จะเป็น IDE ที่เรียบง่าย แต่ **Dev-C++** ก็มีขีดความสามารถเพียงพอสำหรับการเรียนรู้และพัฒนาโปรแกรมพื้นฐานในภาษา C/C++ ได้อย่างสะดวก โดยมีความสามารถหลักดังนี้:

1. รองรับภาษา C และ C++

- สนับสนุนมาตรฐานภาษา C89, C99 และ C++98/03
- เวอร์ชันใหม่ (Embarcadero) รองรับ **C++11, C++14 และ C++17**

2. การรวมกับ Compiler (MinGW)

- ใช้ **MinGW (Minimalist GNU for Windows)** เป็น Compiler หลัก
- สามารถแยกติดตั้งหรือเลือก compiler เองได้

3. Code Editor ที่ใช้งานง่าย

- Syntax Highlighting (ไฮไลต์คำสั่ง)
- Auto-completion (เติมคำอัตโนมัติระดับเบื้องต้น)
- Bracket matching (ไฮไลต์วงเล็บเปิด-ปิด)
- Multiple tabs (เปิดหลายไฟล์พร้อมกันได้)

4. Project Management

- รองรับการสร้างโปรเจกต์

- เพิ่ม/ลบไฟล์ในโปรเจกต์ได้สะดวก
- แยกไฟล์ header และ source อย่างชัดเจน

☐ **5. การ Debug เบื้องต้น**

- มีระบบ Debug ในตัว (แม้ไม่สมบูรณ์แบบ)
- Breakpoint, Step Over, Watch variables (จำกัด)

☐ **6. เครื่องมือเสริม**

- Makefile Generator (ในบางเวอร์ชัน)
- การจัดการกับ Templates
- คำสั่ง Compile/Build/Run อย่างรวดเร็ว
- รองรับ Command-line Arguments

☐ **7. ฟรีและเบา**

- ทำงานรวดเร็ว ใช้ทรัพยากรน้อย
- เหมาะกับเครื่องสเปกต่ำ/เก่า

☐ แนวโน้มในปัจจุบันของ Dev-C++

☐ **ยังมีการใช้งานอยู่โดยเฉพาะใน:**

1. **การศึกษา**
   - วิทยาลัย มหาวิทยาลัยหลายแห่งในไทยและต่างประเทศยังใช้ Dev-C++ ในการสอนเบื้องต้น
   - เหมาะสำหรับ **Intro to C/C++** courses เพราะใช้งานง่าย ติดตั้งง่าย
2. **ผู้เริ่มต้น (Beginner Level)**
   - ผู้ที่เพิ่งเริ่มเขียนโปรแกรมชื่นชอบเพราะไม่ซับซ้อน

☐ **แต่แนวโน้มทั่วไปคือเริ่ม ถูกแทนที่ด้วยเครื่องมือที่ทันสมัยกว่า เช่น:**

IDE	จุดเด่น
Visual Studio Code (VS Code)	มี Extension รองรับ C/C++, IntelliSense, Debugger เต็มรูปแบบ
Code::Blocks	ใช้งานคล้าย Dev-C++ แต่มีฟีเจอร์ครบกว่า
CLion (JetBrains)	IDE เชิงพาณิชย์ที่ทรงพลังมาก รองรับ CMake, Debug, Refactor อย่างมืออาชีพ
Visual Studio	สำหรับ Windows โดยตรง มี debugger และ GUI แบบครบเครื่อง

☐ ข้อสรุป

จุดแข็งของ Dev-C++	ข้อจำกัด
ใช้งานง่าย เหมาะกับผู้เริ่มต้น	ไม่รองรับมาตรฐาน C++ ล่าสุดทั้งหมด
ใช้ทรัพยากรน้อย	ระบบ Debug ไม่สมบูรณ์แบบ
ฟรีและติดตั้งรวดเร็ว	ไม่มีระบบ IntelliSense ขั้นสูง
รองรับการเขียนโปรแกรมพื้นฐานได้ดี	ไม่เหมาะกับโปรเจกต์ระดับใหญ่หรือซับซ้อน

☐ ขั้นตอนการติดตั้งและตั้งค่า Dev-C++

☐ 1. ดาวน์โหลด Dev-C++

เวอร์ชันแนะนำ: Embarcadero Dev-C++

☐ ดาวน์โหลดได้ที่:

☐ https://www.embarcadero.com/free-tools/dev-cpp

หรือถ้าคุณใช้เวอร์ชัน Orwell Dev-C++ (เบากว่า):

☐ https://sourceforge.net/projects/orwelldevcpp/

☐ 2. ติดตั้งโปรแกรม

1. ดับเบิลคลิกที่ไฟล์ .exe ที่ดาวน์โหลดมา
2. เลือกภาษา (ส่วนใหญ่เลือก English)
3. กด **Next** ไปตามขั้นตอนจนถึงหน้าต่าง License Agreement
4. กด **I Agree**, จากนั้นเลือกโฟลเดอร์สำหรับติดตั้ง (เช่น C:\Dev-Cpp)
5. กด **Install**
6. เมื่อติดตั้งเสร็จ ให้คลิก **Finish**

☐ 3. เปิดใช้งานครั้งแรก และตั้งค่าเบื้องต้น

1. เมื่อเปิดโปรแกรมครั้งแรก จะมีหน้าต่างให้เลือก:
   - **Compiler** ⟶ แนะนำให้ใช้ TDM-GCC 9.x หรือ MinGW64
   - **ภาษา**: เลือก English หรือ Thai ตามสะดวก
2. ตรวจสอบว่าโปรแกรมมองเห็น Compiler:
   - ไปที่ **Tools > Compiler Options**
   - ที่แท็บ **"Compiler Set to configure"** ควรแสดงรายการเช่น TDM-GCC, MinGW, หรือชื่อ compiler อื่น
   - หากไม่มี ให้ดาวน์โหลด MinGW แยกแล้วติดตั้งใหม่ (จะอธิบายเพิ่มถ้าต้องการ)

□ 4. สร้างโปรเจกต์แรก (Hello World)

1. ไปที่ **File > New > Source File**
2. พิมพ์โค้ดต่อไปนี้:

```
#include <iostream>
using namespace std;

int main() {
    cout << "Hello, world!" << endl;
    return 0;
}
```

3. กด **Ctrl + S** เพื่อบันทึกไฟล์ เช่น hello.cpp
4. ไปที่ **Execute > Compile & Run** หรือกดปุ่ม F11
5. หน้าต่าง Console จะเปิดขึ้นพร้อมแสดง:

```
Hello, world!
```

□ 5. การตั้งค่าพื้นฐานอื่น ๆ (แนะนำ)

การตั้งค่า	วิธีการ
ปรับขนาด Font	Tools > Editor Options > Fonts
เปิด Line Number	Tools > Editor Options > General > Show Line Numbers
ตั้งค่า Auto Save	Tools > Environment Options > Files
ตั้งค่าการใช้แท็บแทนช่องว่าง	Tools > Editor Options > Indentation

⚠□ ปัญหาที่พบบ่อย

ปัญหา	วิธีแก้
Compile แล้วมี error ว่า "g++.exe not found"	ไปที่ Tools > Compiler Options ตรวจสอบว่า compiler path ถูกต้อง
Run แล้ว Console ปิดเร็วเกินไป	ใส่ system("pause"); หรือ run จาก Command Prompt แทน
พิมพ์ภาษาไทยแล้วเป็นภาษาต่างดาว	เปลี่ยน Font ใน Editor เป็น Tahoma หรือ Consolas และตั้ง encoding เป็น UTF-8

ขั้นตอนการสร้างและคอมไพล์โปรเจกต์แรกใน Dev-C++

ขั้นตอนที่ 1: เปิด Dev-C++ และสร้างโปรเจกต์ใหม่

1. เปิดโปรแกรม **Dev-C++**
2. ไปที่เมนู **File > New > Project...**
3. จะปรากฏหน้าต่าง “New Project”
4. เลือกประเภทโปรเจกต์:
   - เลือก **"Console Application"**
   - เลือกภาษาเป็น **C++**
5. กด **OK**

ขั้นตอนที่ 2: ตั้งชื่อโปรเจกต์และกำหนดตำแหน่ง

1. ในหน้าต่าง "Create New Project":
   - ใส่ชื่อโปรเจกต์ เช่น HelloProject
   - เลือกโฟลเดอร์สำหรับเก็บโปรเจกต์ เช่น C:\Users\YourName\Documents\DevCpp\HelloProject
2. คลิก **Save**

Dev-C++ จะสร้างไฟล์ .dev และโฟลเดอร์โปรเจกต์ให้โดยอัตโนมัติ

ขั้นตอนที่ 3: เขียนโค้ดโปรแกรมแรก

Dev-C++ จะเปิดไฟล์ main.cpp มาให้โดยอัตโนมัติ คุณสามารถแก้ไขให้เป็นโค้ดตัวอย่างด้านล่าง:

```
#include <iostream>
using namespace std;

int main() {
    cout << "Hello, Dev-C++!" << endl;
    return 0;
}
```

อย่าลืมกด **Ctrl + S** เพื่อบันทึกไฟล์

ขั้นตอนที่ 4: คอมไพล์และรันโปรแกรม

1. กดปุ่ม **F11** หรือไปที่เมนู **Execute > Compile & Run**
2. Dev-C++ จะทำการ Compile โค้ด และเปิดหน้าต่าง Console ขึ้นมา

3. คุณจะเห็นข้อความ:

Hello, Dev-C++!

□ แสดงว่าโปรแกรมทำงานได้เรียบร้อยแล้ว!

□ คำสั่ง Compile ที่ควรรู้

คำสั่ง	ความหมาย	คีย์ลัด
Compile	แปลโปรแกรมเป็นไฟล์ .exe แต่ยังไม่รัน	Ctrl + F9
Run	รันไฟล์ .exe ที่คอมไพล์แล้ว	Ctrl + F10
Compile & Run	แปลและรันในขั้นตอนเดียว	F11

□ เคล็ดลับเพิ่มเติม

- □ อย่าลืมบันทึกไฟล์บ่อยๆ (กด Ctrl + S)
- □ หาก Console ปิดเร็วเกินไป ให้เพิ่มคำสั่ง:
- system("pause");
- □ หากโปรเจกต์มีหลายไฟล์ (เช่น .cpp, .h) สามารถคลิกขวาที่ "Project" > "Add to project" เพื่อเพิ่มไฟล์ได้

□ ตัวอย่างโปรเจกต์ Hello Calculator (เสริม)

```
#include <iostream>
using namespace std;

int main() {
    int a, b;
    cout << "Enter two numbers: ";
    cin >> a >> b;
    cout << "Sum = " << (a + b) << endl;
    return 0;
}
```

การติดตั้งและตั้งค่า Dev-C++ (2)

1. **ดาวน์โหลดโปรแกรม**
   ไปที่เว็บของ Embarcadero Dev-C++ (หรือเลือก Orwell ของ SourceForge ก็ได้) และดาวน์โหลดเวอร์ชันล่าสุด เช่น 6.x (embarcadero.com)
2. **เปิดโปรแกรมติดตั้ง**
   ดับเบิลคลิกไฟล์ติดตั้ง เช่น devcpp-6.3.setup.exe
   เลือกภาษา ⟶ กด “OK” (reddit.com, programmingzona.blogspot.com)
3. **ยอมรับข้อตกลง**
   เลือก “I Agree” เพื่อต่อไป (ultahost.com, programmingzona.blogspot.com)
4. **เลือก Components**
   เลือก “Full installation” เพื่อให้ได้ IDE + MinGW compiler ตามค่าเริ่มต้น (programmingzona.blogspot.com)
5. **กำหนดที่เก็บไฟล์**
   โปรแกรมจะแนะนำโฟลเดอร์เช่น C:\Dev-Cpp หรือ Program Files\Dev-C++ กด Next และ Install (youtube.com, programmingzona.blogspot.com)
6. **ติดตั้งเสร็จสิ้น**
   เลือก “Run Dev-C++” แล้วกด Finish (begincpp.blogspot.com)
7. **ตั้งค่าเบื้องต้น**
   - ตรวจสอบคอมไพเลอร์ที่ Tools ⟶ Compiler Options
   - ตั้งค่าฟอนต์ และเปิดเลขบรรทัดที่ Tools ⟶ Editor Options
   - ตั้ง auto-save และ tab/space ตามชอบ

สร้างและคอมไพล์โปรเจกต์แรก (Hello World)

1. **สร้างโปรเจกต์ใหม่**
   ไปที่ File ⟶ New ⟶ Project...
   เลือก **Console Application > C++ Project** (technologyuk.net, cs.uno.edu)
   ตั้งชื่อ เช่น HelloProject และเลือกโฟลเดอร์เก็บไฟล์
2. **เขียนโค้ดในไฟล์ main.cpp**
   ลบโค้ดเดิม (ถ้ามี) และพิมพ์:
3. #include <iostream>
4. using namespace std;
5.
6. int main() {

```
7.      cout << "Hello, Dev-C++!" << endl;
8.      return 0;
9.  }
```

10. **บันทึกไฟล์**

    กด Ctrl + S (ระบบจะตั้งชื่อไฟล์ main.cpp ให้โดยอัตโนมัติ)

11. **คอมไพล์และรัน**

    กด F11 หรือเลือก Execute ⟶ Compile & Run โปรแกรมจะ compile และเปิดหน้าต่าง console แสดงผล:

12. Hello, Dev-C++!
13. **เรียนรู้คำสั่งเพิ่มเติม**
    - Ctrl + F9: Compile เท่านั้น
    - Ctrl + F10: Run เท่านั้น
    - F11: Compile & Run
14. **กรณี console ปิดเร็วเกินไป**

    เพิ่มบรรทัด system("pause"); ก่อน return 0; หรือรันจาก Command Prompt เพื่อดูผลลัพธ์

สรุปขั้นตอน

ขั้นตอน	รายละเอียด
ดาวน์โหลด	จากเว็บไซต์ Embarcadero หรือ SourceForge
ติดตั้ง	เลือก Full install, ตั้งค่าภาษา, ที่เก็บไฟล์
ตั้งค่า IDE	ตรวจสอบคอมไพเลอร์, ปรับฟอนต์, Auto-save
สร้างโปรเจกต์	เลือก Console/C++ Project, ตั้งชื่อ
เขียนโค้ด	พิมพ์ Hello World บันทึก
คอมไพล์ & รัน	ใช้ F11 หรือเมนู Execute

ส่วนประกอบของโปรแกรม C++ เบื้องต้น

```
#include <iostream>  // (1) Header file

using namespace std; // (2) Namespace

int main() {         // (3) main function
    cout << "Hello, C++!" << endl;  // (4) คำสั่งแสดงผล
```

```
    return 0;        // (5) คืนค่ากลับไปยังระบบปฏิบัติการ
}
```

☐ **(1) Header File**

#include <iostream>

- คือไฟล์ที่ใช้สำหรับนำเข้าฟังก์ชันหรือวัตถุที่จำเป็นจากมาตรฐานภาษา C++
- <iostream> ย่อมาจาก **input-output stream** ใช้สำหรับการแสดงผล (เช่น cout) และรับค่าจากคีย์บอร์ด (เช่น cin)

☐ คำสั่งที่ขึ้นต้นด้วย #include เป็น **Preprocessor Directive** (คำสั่งให้โปรแกรมเตรียมความพร้อมก่อนการคอมไพล์)

☐ **(2) Namespace**

using namespace std;

- std คือชื่อของ **standard namespace** ที่รวบรวมฟังก์ชันและวัตถุจาก C++ มาตรฐาน
- cout, cin, endl อยู่ใน std
- ถ้าไม่ใส่บรรทัดนี้ ต้องเขียนเต็มว่า std::cout, std::endl

☐ เป็นตัวช่วยให้โค้ดกระชับขึ้น แต่ในระบบขนาดใหญ่ มักนิยมไม่ใช้ using namespace std; เพราะอาจชนกับ namespace อื่น

☐ **(3) main() Function**

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

- เป็นจุดเริ่มต้นของโปรแกรม C++
- คอมไพเลอร์จะเริ่มทำงานจากฟังก์ชัน main() ก่อนเสมอ
- ต้องมีอยู่ในทุกโปรแกรม ไม่สามารถละเว้นได้

int main()

- หมายถึง ฟังก์ชันนี้จะ **ส่งค่ากลับเป็นจำนวนเต็ม (int)**
- return 0; บ่งบอกว่าการทำงานสำเร็จ

☐ **(4) คำสั่งภายใน main(): การแสดงผล**

cout << "Hello, C++!" << endl;

- cout = console output = ใช้แสดงผลข้อความออกทางหน้าจอ

- << = operator สำหรับส่งข้อความไปยัง cout
- "Hello, C++!" = string ที่จะแสดง
- endl = ขึ้นบรรทัดใหม่ (เหมือน \n)

☐ (5) คำสั่ง return

return 0;

- ส่งค่ากลับไปยังระบบปฏิบัติการ
- 0 แปลว่าไม่มีข้อผิดพลาด
- ถ้าไม่เขียน return ก็จะถือว่าคืนค่า 0 โดยปริยาย (เฉพาะใน C++ เวอร์ชันใหม่)

☐ ตัวอย่างโปรแกรม C++ ที่สมบูรณ์

```
#include <iostream>      // ใช้งาน cout, cin
using namespace std;     // ใช้งาน std โดยไม่ต้องพิมพ์เต็ม

int main() {
    int a, b;
    cout << "Enter two numbers: ";
    cin >> a >> b;  // รับค่าจากผู้ใช้
    int sum = a + b;
    cout << "Sum = " << sum << endl;
    return 0;
}
```

☐ สรุป: ส่วนประกอบสำคัญของโปรแกรม C++

ส่วน	คำอธิบาย
#include	นำเข้า header files
using namespace std;	เรียกใช้ std namespace
main()	จุดเริ่มต้นของโปรแกรม
cout	แสดงผล
cin	รับค่าจากผู้ใช้
return 0;	ส่งค่าสถานะการทำงานกลับระบบ

□ การเขียนโปรแกรม Hello World! ใน Dev-C++

□ ขั้นตอนที่ 1: เปิดโปรแกรม Dev-C++

1. เปิดโปรแกรม Dev-C++ จาก Start Menu หรือไอคอนบน Desktop
2. คลิก **File ⟶ New ⟶ Source File** เพื่อสร้างไฟล์ใหม่

□ ขั้นตอนที่ 2: พิมพ์โค้ด C++ ดังนี้

```
#include <iostream>      // เรียกใช้ไลบรารีสำหรับแสดงผล
using namespace std;     // ใช้ namespace มาตรฐาน

int main() {
    cout << "Hello World!";   // แสดงผลข้อความ
    return 0;                 // จบโปรแกรม
}
```

□ ขั้นตอนที่ 3: บันทึกไฟล์

1. ไปที่เมนู **File ⟶ Save As**
2. ตั้งชื่อไฟล์ว่า hello.cpp และเลือกไดเรกทอรีที่ต้องการบันทึก
3. คลิก **Save**

▶□ ขั้นตอนที่ 4: คอมไพล์และรัน

- คลิกปุ่ม **Compile & Run** (รูปจรวดสีเขียว) หรือกดปุ่ม **F11**

□ ผลลัพธ์ที่ได้ในหน้าต่าง Console:

```
Hello World!
```

□ คำอธิบายโค้ดแบบง่าย

ส่วนของโค้ด	ความหมาย
#include <iostream>	ใช้สำหรับแสดงผลด้วย cout
using namespace std;	ย่นชื่อฟังก์ชันจาก std::cout ให้เหลือแค่ cout
int main()	ฟังก์ชันหลักของโปรแกรม เป็นจุดเริ่มต้นของการทำงาน
cout << "Hello World!";	แสดงข้อความ "Hello World!" ออกหน้าจอ
return 0;	ส่งค่ากลับ 0 เพื่อบอกว่าโปรแกรมจบอย่างปกติ

โครงสร้างพื้นฐานของโปรแกรม Hello World

```
+---------------------------+
|      #include             |
|      using namespace      |
|      int main()           |
|        cout << ...        |
|        return 0;          |
+---------------------------+
```

สรุป

บทแรกของหนังสือเล่มนี้มุ่งเน้นการปูพื้นฐานการใช้งานโปรแกรม Dev-C++ สำหรับผู้เริ่มต้น โดยครอบคลุมตั้งแต่การติดตั้งและตั้งค่าโปรแกรมอย่างถูกต้อง การสร้างและคอมไพล์โปรเจกต์แรก การทำความเข้าใจส่วนประกอบสำคัญของโปรแกรม C++ เช่น main() และ header files ไปจนถึงการทดลองเขียนโปรแกรม "Hello World!" เพื่อให้ผู้อ่านได้สัมผัสกับกระบวนการพัฒนาโปรแกรมตั้งแต่ต้นจนจบอย่างครบถ้วน เป็นการเตรียมความพร้อมที่สำคัญก่อนเข้าสู่เนื้อหาเชิงลึกในบทต่อไป.

บทที่ 2
พื้นฐานภาษา C++
(C++ Basic)

เนื้อหา

- พื้นฐานภาษา C++
- ข้อมูลเพิ่มเติมพื้นฐานภาษา C++ (Extended)
- ตัวแปรและชนิดข้อมูลพื้นฐานใน C++
- การใช้ cin และ cout ใน C++
- การใช้ cin และ cout ใน C++ — รายละเอียดเพิ่ม
- ตัวดำเนินการ (Operators) ใน C++

บทนำ

ภาษา C++ เป็นภาษาคอมพิวเตอร์เชิงโครงสร้างและเชิงวัตถุที่ได้รับความนิยมสูง เนื่องจากมีความยืดหยุ่นในการเขียนโปรแกรม ทั้งในระดับพื้นฐานและขั้นสูง สำหรับผู้เริ่มต้น การเข้าใจพื้นฐานของภาษาถือเป็นสิ่งสำคัญยิ่งในการพัฒนาไปสู่การสร้างโปรแกรมที่ซับซ้อน โดยบทนี้มุ่งเน้นให้ผู้อ่านได้เรียนรู้เรื่องตัวแปร ชนิดข้อมูล การรับ-ส่งค่าทางหน้าจอ และการใช้ตัวดำเนินการ ซึ่งเป็นโครงสร้างพื้นฐานของทุกโปรแกรม C++

เนื้อหาเริ่มต้นจากการแนะนำตัวแปร (Variables) ซึ่งทำหน้าที่เก็บข้อมูลในหน่วยความจำของคอมพิวเตอร์ และชนิดข้อมูลพื้นฐานที่สำคัญ เช่น int สำหรับจำนวนเต็ม, float สำหรับเลขทศนิยม, char สำหรับตัวอักษร และ bool สำหรับค่าความจริง (true/false) การเลือกใช้ชนิดข้อมูลอย่างเหมาะสมจะช่วยให้โปรแกรมทำงานได้อย่างมีประสิทธิภาพและปลอดภัยจากข้อผิดพลาด

จากนั้นผู้อ่านจะได้เรียนรู้การใช้งานคำสั่ง cin และ cout ซึ่งใช้ในการรับข้อมูลจากผู้ใช้และแสดงผลลัพธ์ออกทางหน้าจอ โดย cin ทำหน้าที่รับค่าจากคีย์บอร์ด และ cout แสดงข้อมูลไปยังหน้าจอ ซึ่งทั้งสองคำสั่งนี้เป็นเครื่องมือหลักในการโต้ตอบกับผู้ใช้งาน

ท้ายที่สุดในบทนี้ ผู้อ่านจะได้ศึกษาการใช้ตัวดำเนินการ (Operators) เช่น เครื่องหมายคำนวณพื้นฐาน +, -, *, /, % รวมถึงการเพิ่มค่าหรือลดค่าด้วย ++ และ -- ซึ่งเป็นสิ่งที่ขาดไม่ได้ในการสร้างโปรแกรมเชิงคำนวณหรือเชิงเงื่อนไข เนื้อหาในบทนี้จะเป็นพื้นฐานสำคัญที่ส่งเสริมความเข้าใจของผู้อ่านในบทถัดไปซึ่งมีความซับซ้อนมากยิ่งขึ้น.

2. พื้นฐานภาษา C++

2.1 ตัวแปรและชนิดข้อมูลพื้นฐาน

- **ตัวแปร (Variable)** คือพื้นที่ในหน่วยความจำที่เก็บข้อมูล
- **ชนิดข้อมูล (Data Types)** ที่ใช้บ่อยใน C++ ได้แก่:

ชนิดข้อมูล	คำอธิบาย	ขนาด (ประมาณ)
int	จำนวนเต็ม เช่น 10, -3, 0	4 bytes
float	จำนวนทศนิยม เช่น 3.14, -0.5	4 bytes
char	ตัวอักษร เช่น 'A', 'b', '9'	1 byte
bool	ค่า True หรือ False	1 byte

2.2 การประกาศตัวแปร

```
int age = 20;           // ตัวแปรเก็บเลขจำนวนเต็ม
float pi = 3.14;        // ตัวแปรเก็บเลขทศนิยม
char grade = 'A';       // ตัวแปรเก็บตัวอักษร
bool isPass = true;     // ตัวแปรเก็บค่าบูลีน (จริง/เท็จ)
```

2.3 การใช้ cin และ cout สำหรับรับและแสดงผลข้อมูล

- cout ใช้แสดงผลข้อความหรือค่าตัวแปรออกหน้าจอ
- cin ใช้รับค่าจากผู้ใช้

ตัวอย่างโปรแกรม: รับข้อมูลและแสดงผล

```
#include <iostream>
using namespace std;

int main() {
    int number;
    cout << "Enter a number: ";    // แสดงข้อความ
    cin >> number;                 // รับค่าจากผู้ใช้เก็บในตัวแปร number

    cout << "You entered: " << number << endl;  // แสดงผลตัวแปร
    return 0;
}
```

2.4 การใช้ตัวดำเนินการ (Operators)

ตัวดำเนินการ	ความหมาย	ตัวอย่าง	ผลลัพธ์
+	บวก	5 + 3	8
-	ลบ	5 - 3	2
*	คูณ	5 * 3	15
/	หาร	10 / 2	5
%	หารเอาเศษ (Modulo)	10 % 3	1
++	เพิ่มค่า 1 (Increment)	int a = 5; a++;	6
--	ลดค่า 1 (Decrement)	int a = 5; a--;	4

2.5 ตัวอย่างโปรแกรมรวมการใช้ตัวแปร, cin, cout และตัวดำเนินการ

```
#include <iostream>
using namespace std;

int main() {
    int a, b;

    cout << "Enter first number: ";
    cin >> a;

    cout << "Enter second number: ";
    cin >> b;

    cout << "Sum: " << (a + b) << endl;
    cout << "Difference: " << (a - b) << endl;
    cout << "Product: " << (a * b) << endl;
    cout << "Quotient: " << (a / b) << endl;
    cout << "Remainder: " << (a % b) << endl;

    a++;  // เพิ่มค่า a ทีละ 1
    cout << "Incremented first number: " << a << endl;
```

```
    b--;  // ลดค่า b ทีละ 1
    cout << "Decremented second number: " << b << endl;

    return 0;
}
```

2.6 คำอธิบายผลการรัน

สมมติผู้ใช้กรอกค่า 10 และ 3

Enter first number: 10

Enter second number: 3

Sum: 13

Difference: 7

Product: 30

Quotient: 3

Remainder: 1

Incremented first number: 11

Decremented second number: 2

ข้อมูลเพิ่มเติมพื้นฐานภาษา C++ (Extended)

---

1. ตัวแปร (Variables) และชนิดข้อมูล (Data Types) เพิ่มเติม

ชนิดข้อมูล	คำอธิบาย	ขนาดโดยประมาณ	ตัวอย่างค่าที่เก็บได้
int	จำนวนเต็ม	4 bytes	0, -1, 123, 10000
float	จำนวนทศนิยมความแม่นยำปานกลาง	4 bytes	3.14159, -0.5
double	จำนวนทศนิยมความแม่นยำสูง	8 bytes	3.1415926535, -0.123456789
char	ตัวอักษร 1 ตัว	1 byte	'A', 'b', '9', '?'
bool	ค่าความจริง (จริง/เท็จ)	1 byte	true, false
long, long long	จำนวนเต็มขนาดใหญ่ (มากกว่า int)	8 bytes (long long)	1234567890123

ชนิดข้อมูล	คำอธิบาย	ขนาดโดยประมาณ	ตัวอย่างค่าที่เก็บได้
unsigned int	จำนวนเต็มบวกเท่านั้น	4 bytes	0, 100, 4294967295 (ค่าบวกเท่านั้น)

---

2. การประกาศตัวแปรและการกำหนดค่า

- การประกาศตัวแปรเปล่า (ยังไม่กำหนดค่าเริ่มต้น)

```
int age;      // ตัวแปรชื่อ age ประเภทจำนวนเต็ม (ค่าเริ่มต้นจะเป็นค่าขยะ)
```

- การประกาศพร้อมกำหนดค่าเริ่มต้น

```
int age = 25;
float pi = 3.14f;   // ตัว f เพื่อบอกว่าเป็น float literal
double e = 2.71828;
char ch = 'Z';
bool isTrue = false;
```

- การประกาศหลายตัวแปรพร้อมกัน

```
int x = 0, y = 1, z = 2;
```

---

3. การใช้ cin รับข้อมูลหลายชนิด

```
#include <iostream>
using namespace std;

int main() {
    int age;
    float height;
    char grade;

    cout << "Enter your age: ";
    cin >> age;

    cout << "Enter your height (in meters): ";
    cin >> height;

    cout << "Enter your grade (A-F): ";
    cin >> grade;
```

```
    cout << "Age: " << age << ", Height: " << height << ", Grade: " << grade << endl;

    return 0;
}
```

หมายเหตุ:

- cin จะรับข้อมูลจนถึงช่องว่างหรือ Enter
- รับ char จะรับแค่ตัวอักษรตัวเดียว
- หากต้องการรับข้อความหลายคำหรือบรรทัดต้องใช้ getline() (ต่อไปสามารถสอนได้)

---

4. ตัวดำเนินการ (Operators) ในเชิงลึก

ประเภทตัวดำเนินการ	ตัวอย่าง	คำอธิบาย
Arithmetic Operators	+, -, *, /, %	บวก, ลบ, คูณ, หาร, หารเอาเศษ
Increment/Decrement	++, --	เพิ่มหรือลดค่าทีละ 1
Assignment Operators	=, +=, -=, *=, /=, %=	กำหนดค่าและบวก-ลบ-คูณ-หารแบบย่อ
Comparison Operators	==, !=, <, >, <=, >=	เปรียบเทียบค่า
Logical Operators	&&, `	

---

5. ตัวอย่างการใช้ตัวดำเนินการเพิ่ม

```
#include <iostream>
using namespace std;

int main() {
    int a = 5, b = 3;

    a += 2;  // เท่ากับ a = a + 2; a = 7
    b *= 4;  // เท่ากับ b = b * 4; b = 12

    cout << "a = " << a << endl;  // 7
    cout << "b = " << b << endl;  // 12

    bool result = (a > b) && (b != 0);
```

```
    cout << "Result of logical expression: " << result << endl;  // false (0)

    return 0;
}
```

---

6. การแสดงผลของ bool ใน C++

- ค่า true จะแสดงเป็น 1
- ค่า false จะแสดงเป็น 0

ถ้าต้องการแสดงเป็นคำว่า true หรือ false แทนตัวเลข ใช้

```
cout << boolalpha;
cout << "Result: " << result << endl;
```

---

7. หมายเหตุเกี่ยวกับการหารจำนวนเต็ม

- ถ้าใช้ / กับตัวแปร int จะทำการหารแบบตัดเศษ (integer division)

```
int x = 7, y = 2;
cout << x / y << endl;  // ผลลัพธ์คือ 3 (ไม่ใช่ 3.5)
```

ถ้าต้องการผลลัพธ์ทศนิยม ต้องใช้ float หรือ double เช่น

```
float x = 7.0f, y = 2.0f;
cout << x / y << endl;  // ผลลัพธ์คือ 3.5
```

ตัวแปรและชนิดข้อมูลพื้นฐานใน C++

1. ตัวแปร (Variable)

- คือชื่อที่เราใช้แทนพื้นที่เก็บข้อมูลในหน่วยความจำ
- ก่อนใช้งานต้องประกาศชนิดข้อมูลและชื่อตัวแปรเสมอ
- ตัวแปรสามารถเก็บค่าต่างๆ ที่กำหนดได้ในโปรแกรม

---

2. ชนิดข้อมูลพื้นฐานหลัก

ชนิดข้อมูล	คำอธิบาย	ขนาด (โดยประมาณ)	ตัวอย่างค่า
int	เก็บเลขจำนวนเต็ม (ไม่มีจุดทศนิยม)	4 bytes	10, -5, 0
float	เก็บเลขทศนิยมแบบความแม่นยำปานกลาง	4 bytes	3.14, -0.5
char	เก็บตัวอักษร 1 ตัว	1 byte	'A', 'z', '9'
bool	เก็บค่า **จริง (true)** หรือ **เท็จ (false)**	1 byte	true, false

---