การเขียนโปรแกรมภาษา C++
WEEK #1

ดร.นิฏฐิตา เชิดชู โปรแกรมวิชาวิศวกรรมโทรคมนาคม
มหาวิทยาลัยราชภัฏนครปฐม

คำอธิบายรายวิชา

- ขั้นตอนวิธีและผังงาน วิธีการแก้ปัญหาทางวิทยาศาสตร์และอุตสาหกรรมด้วยคอมพิวเตอร์ การออกแบบและพัฒนาโปรแกรม การเขียนโปรแกรมเบื้องต้นด้วยภาษาระดับสูง การฝึกปฏิบัติการโปรแกรมด้วยเครื่องคอมพิวเตอร์

วัตถุประสงค์

- เพื่อพัฒนาความสามารถในการนำคอมพิวเตอร์มาใช้ในงานอุตสาหกรรมในด้านดังต่อไปนี้
  - เพื่อพัฒนาความสามารถในการเขียนโปรแกรมภาษาระดับสูง (C++)
  - เพื่อพัฒนาโปรแกรมคอมพิวเตอร์ในการแก้ปัญหาทางวิศวกรรม

เกณฑ์การวัดผล

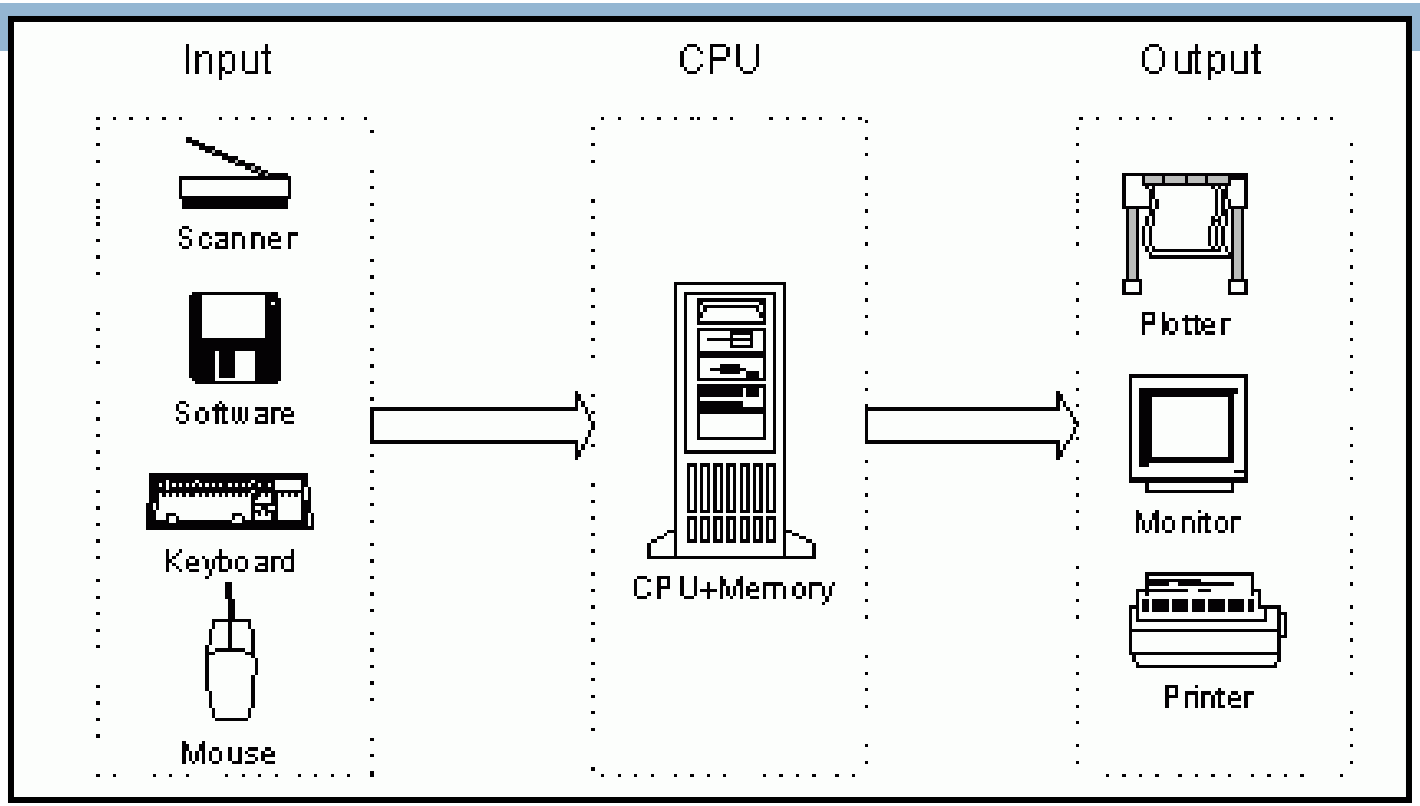
- ระหว่างภาค 80 %
  - เวลาเรียน คิดเป็น 15 %
  - งานทดลองและแบบฝึกหัด คิดเป็น 45%
  - งานศึกษาและค้นคว้าด้วยตนเอง คิดเป็น 10 %
  - การทดสอบระหว่างภาค คิดเป็น 15 % (ข้อเขียน)

- ปลายภาค 20 %
  - ทดสอบปลายภาค คิดเป็น 15 % (ปฏิบัติ)
- รวมเป็น 100 %

เกณฑ์การประเมินผล

A	85-100	คะแนน
B+	80-84	คะแนน
B	75-79	คะแนน
C+	70-74	คะแนน
C	65-69	คะแนน
D+	60-64	คะแนน
D	55-59	คะแนน
E	ต่ำกว่า 50	คะแนน

การโปรแกรมคอมพิวเตอร์



หน่วยรับข้อมูล **หน่วยประมวลผล** **หน่วยแสดงผล**

เซ็นเซอร์ต่าง ๆ เช่นอุณหภูมิ กระแส แรงดัน ความเร็ว

ระบบฝังตัว

สายพานลำเลียง ปิด/เปิดอุปกรณ์ไฟฟ้า

การโปรแกรมคอมพิวเตอร์

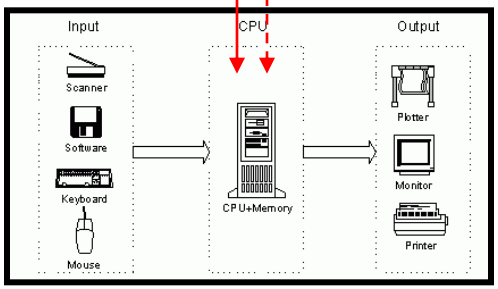
- ภาษาสั่งงานคอมพิวเตอร์ หมายถึง ชุดคำสั่งที่เขียนขึ้นตามรูปแบบและโครงสร้างของภาษาเพื่อสั่งงานให้คอมพิวเตอร์ทำงานตามชุดคำสั่งหรือโปรแกรมซึ่งเขียนถูกขึ้นโดยโปรแกรมเมอร์ (Programmer)ภาษาสั่งงานคอมพิวเตอร์สามารถจำแนกออกได้ 3 ระดับดังนี้

1. ภาษาระดับต่ำ (Low Level Language)

2. ภาษาระดับกลาง (Medium Level Language)

3. ภาษาระดับสูง (High Level Language)

ระบบปฏิบัติการ(OS)

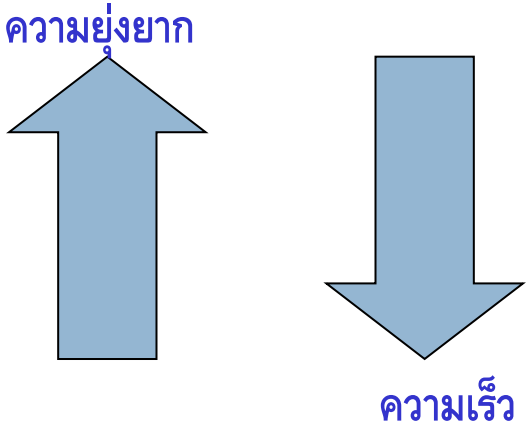


3. ภาษาระดับสูง (High Level Language)

- เป็นภาษาที่สามารถศึกษาและเข้าใจได้ง่าย
- มีลักษณะของคำสั่งคล้ายกับประโยคทางภาษาอังกฤษ
- ง่ายต่อการทำความเข้าใจและใช้เวลาในการเขียนโปรแกรมน้อย
- แต่การสั่งงานให้คอมพิวเตอร์ทำงานได้ช้า
- การสั่งงานให้คอมพิวเตอร์ทำงานต้องมีการแปลความหมายให้เป็นภาษาเครื่องก่อนโดยใช้ตัวแปลภาษาที่เรียกว่า อินเทอร์พรีเทอร์(Interpreter) หรือคอมไพล์เลอร์(Compiler)
- ภาษาเบสิค(BASIC), ปาสคาล(PASCAL), ซี (C), C++ เป็นต้น

ระดับของภาษา

Machine languages
Assembly languages
High-level languages



+1300042774	load basepay	grossPay = basePay + overTimePay;
+1400593419	add overpay	
+1200274027	store grosspay	
Machine languages	Assembly languages	High-level languages

ระดับ	ภาษาระดับต่ำ	ภาษาระดับสูง
การติดต่อกับฮาร์ดแวร์	ติดต่อโดยตรง, เร็ว	ไม่ติดต่อโดยตรง, ช้า
ลักษณะการเขียน	คำสั่งเข้าใจยาก	คำสั่งมีความหมายชัดเจน
การใช้งาน	ยึดติดกับระบบ	สามารถใช้ได้ในทุกระบบ
ตัวอย่างภาษา	Assembly	Pascal, Fortran, etc.

Program Structure in C++

```
// my first program in C++

#include <iostream>
using namespace std;

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

Comment

Directive for Preprocessor

ใช้ standard library

ฟังก์ชันหลัก

```
Hello World!
```

Program Structure in C++

- Comment
  - ใช้เพื่อให้ง่ายต่อการเข้าใจการเขียนโปรแกรม
  - ไม่มีผลต่อการทำงานของ **compiler**
  - // สำหรับ **comment** 1 บรรทัด
  - /* */ สำหรับ **comment** หลายบรรทัด

```
/* my second program in C++
   with more comments */
```

```
// my first program in C++

#include <iostream>
using namespace std;

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

Program Structure in C++

```
// my first program in C++

#include <iostream>
using namespace std;

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

- ขึ้นต้นด้วย #
-ใช้บอกตัว compiler ว่าให้ include iostream standard files เข้าไว้ด้วยเนื่องจากต้องมีการใช้งานในฟังก์ชันหลักต่อไป
-Iostream standard file ประกอบไปด้วยไฟล์ย่อยๆ ที่เกี่ยวข้องกับการ i/o

- Preprocessor Directive เป็นชุดคำสั่งพิเศษแตกต่างจากชุดคำสั่งทั่วไป เมื่อมีการ compile โปรแกรม คำสั่งแบบนี้จะถูกแปลความหมายก่อน
- ชุดคำสั่งแบบ preprocessor directive

#include	#define	#error	#if	#endif
#elif	#else	#indef	#ifndef	#undef
#line	#pragma			

Preprocessor Directive – #include

- คำสั่ง #include เป็นการสั่งให้ compiler นำไฟล์ที่กำหนดหลังคำสั่งมารวมกับโปรแกรมก่อนจะทำการแปลโปรแกรม ดังนั้น #include จะต้องอยู่ที่ส่วนหัวของโปรแกรมเสมอ
- การใช้ #include มีสองแบบ
  - #include <filename>★
    compiler จะค้นหาไฟล์จาก folder TC\INCLUDE ก่อนหากไม่เจอจะกลับมาค้นที่ folder ที่บันทึกโปรแกรมไว้
  - #include “filename”
    compiler จะค้นหาไฟล์จาก folder ที่บันทึกโปรแกรมก่อน จากนั้นจึงจะไปค้นหาที่ TC\INCLUDE

Program Structure in C++

```
// my first program in C++

#include <iostream>
using namespace std;

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

Statement ต้องจบด้วย ; เสมอ

บ่งบอกกับ compiler ว่าจะมีการเรียกใช้ฟังก์ชันที่อยู่ในกลุ่มของ standard library

ฟังก์ชันหลัก

MAIN FUNCTION

- ฟังก์ชันหลัก
- เมื่อโปรแกรมทำงาน (**executed**) โปรแกรมจะเริ่มทำงานจากฟังก์ชันหลักก่อนเป็นอันดับแรกเสมอ
- ภายในโปรแกรมหลัก จะสามารถบรรจุฟังก์ชันย่อยอื่นๆอีกหรือไม่ก็ได้
- แต่ฟังก์ชันหลักจะต้องมีเสมอ !!!

```
// my first program in C++

#include <iostream>
using namespace std;

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

→ ฟังก์ชันที่ใช้เพื่อแสดงผลออกที่หน้าจอ

→ บ่งบอกว่าโปรแกรมสิ้นสุดการทำงานแล้ว

รายละเอียดอื่นๆ เกี่ยวกับ C++

- Case-sensitive – a ≠ A
- Space-insensitive

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

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

- เขียน comment
- เขียนให้อ่านง่าย

  main() {cout<<("Hello World\n");}
- ตั้งชื่อตัวแปลให้มีความหมายสอดคล้องกับตัวแปรนั้นๆ
- ทำการ initial ตัวแปลเสมอ
- ใช้เครื่องหมายวงเล็บเสมอ เพื่อหลีกเลี่ยงการงง
  - a=(10.0 + 2.0) * (5.0 - 6.0) / 2.0

ตัวอย่าง: output ของโปรแกรมออกมาเหมือนกันหรือไม่

- A

```
// my second program in C++

#include <iostream>
using namespace std;

int main ()
{
  cout << "Hello World! ";
  cout << "I'm a C++ program";
  return 0;
}
```

```
Hello World! I'm a C++ program
```

- B

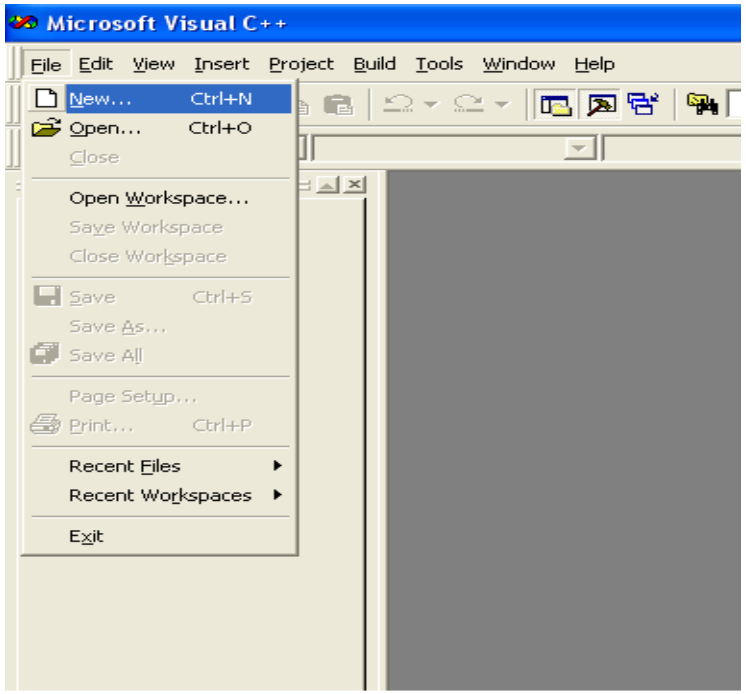
```
int main () { cout << " Hello World! "; cout << " I'm a C++ program "; return 0; }
```

- C

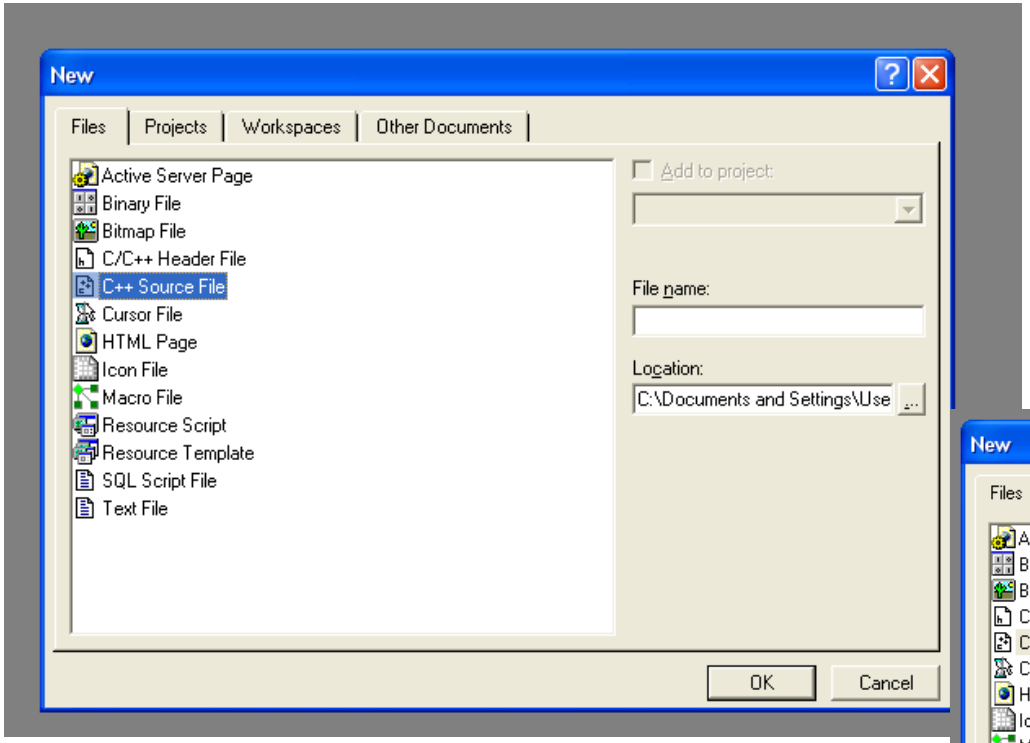
```
int main ()
{
  cout <<
    "Hello World!";
  cout
    << "I'm a C++ program";
  return 0;
}
```

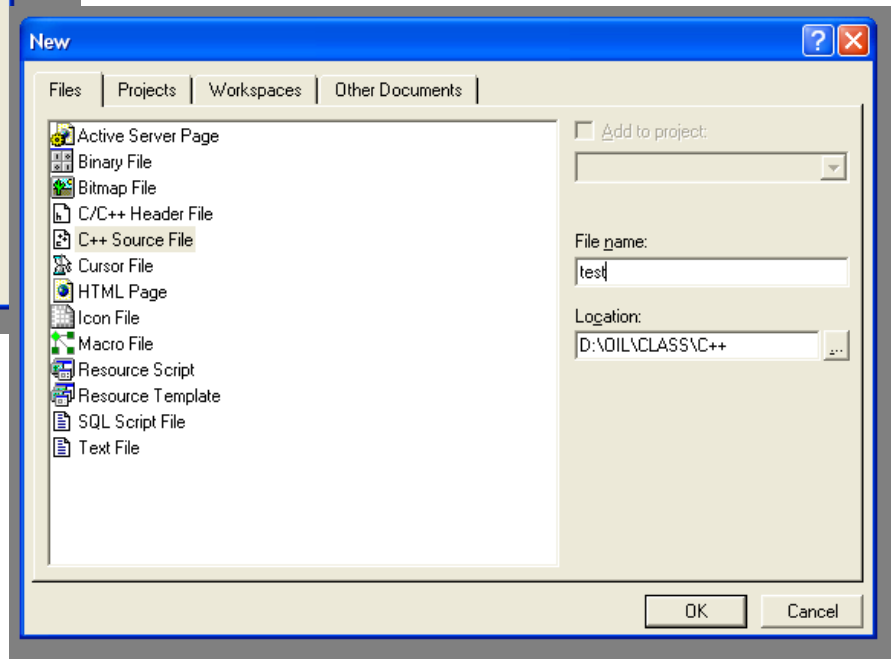
How to run a program in VC++

- เปิด VC++



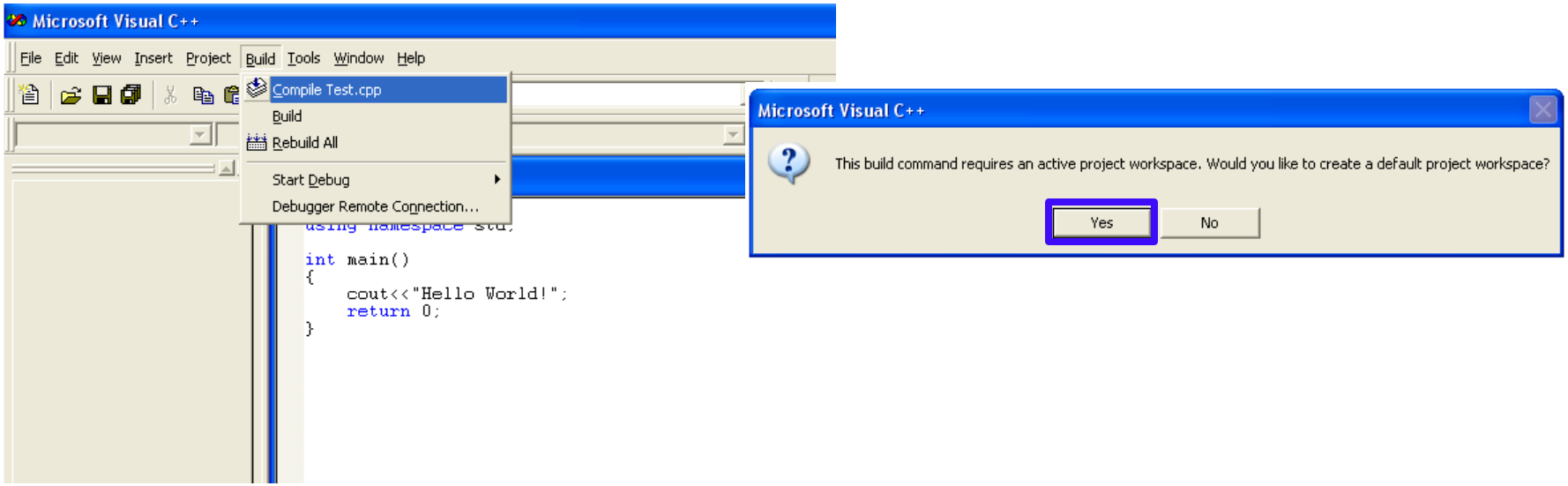
How to run a program in VC++





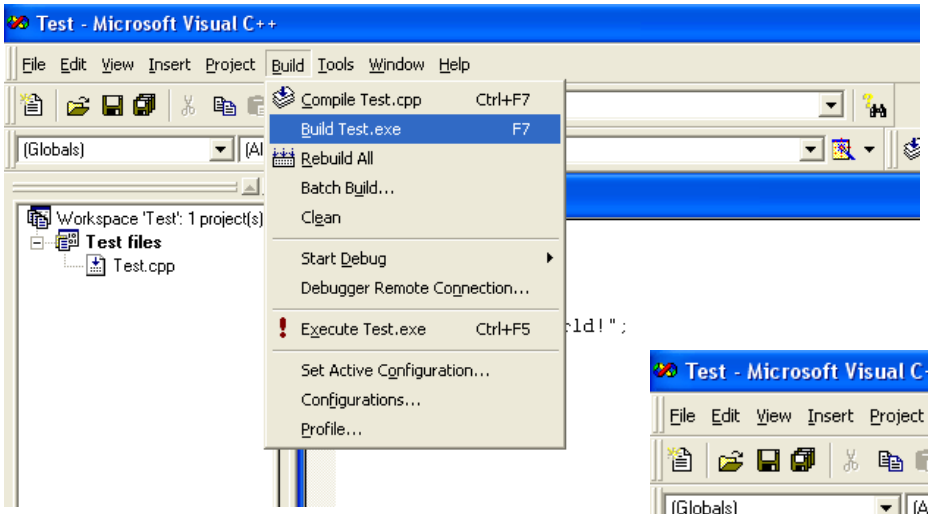
How to run a program in VC++

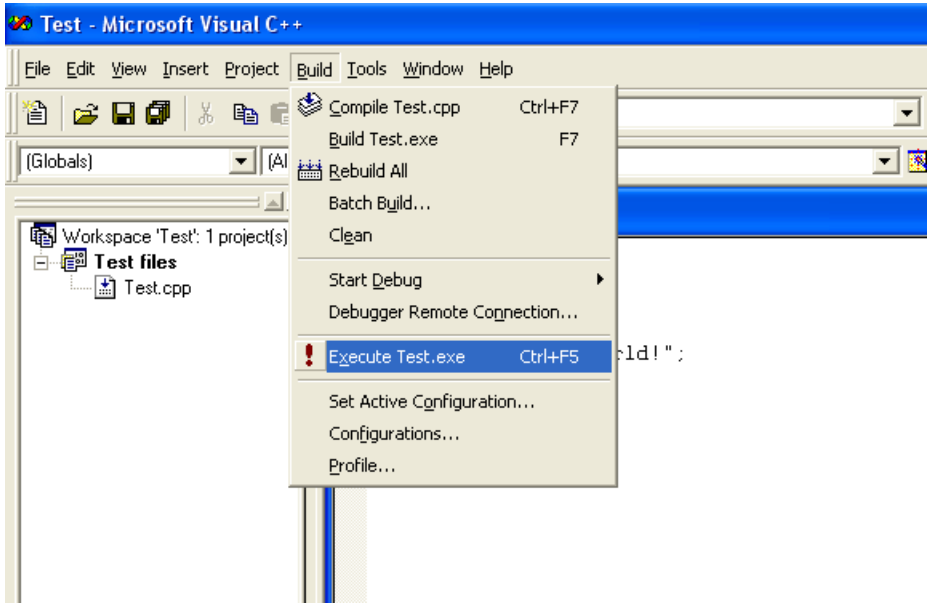
- พิมพ์และเซฟไฟล์เป็น .cpp
- Compile – cpp → exe



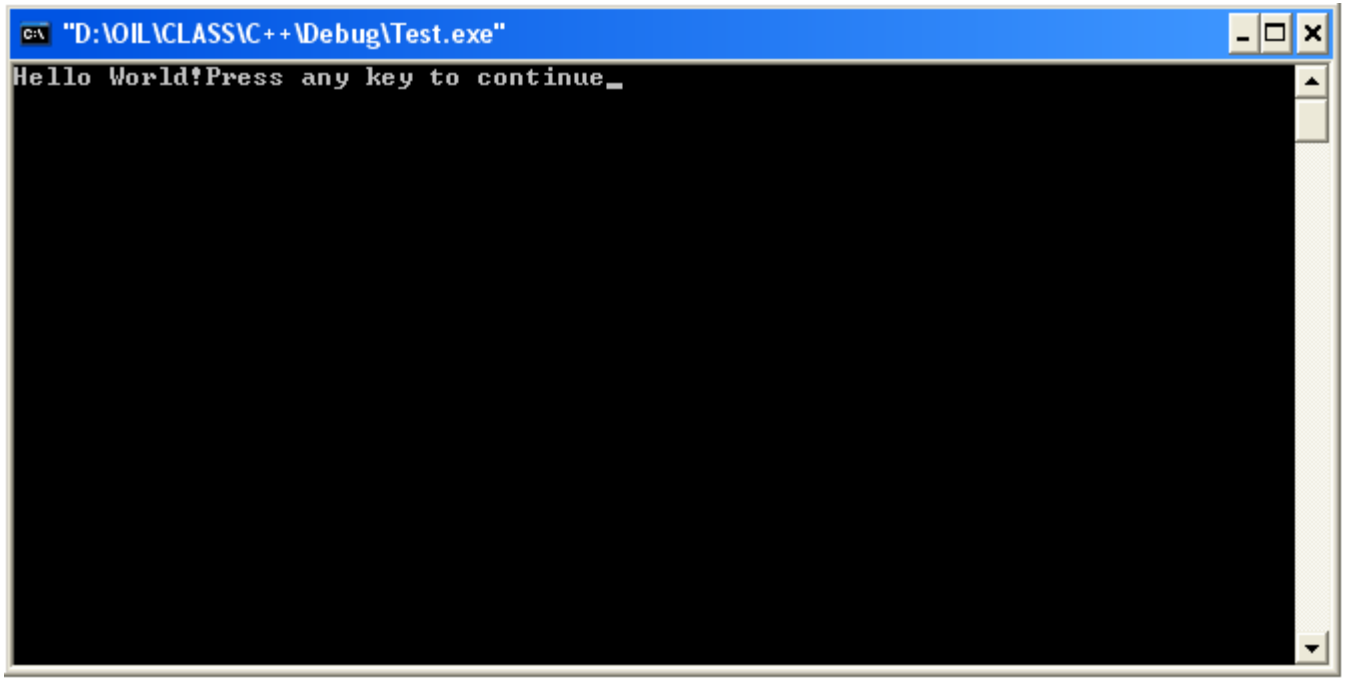
How to run a program in VC++

- Run โปรแกรม





Output


"D:\OIL\CLASS\C++\Debug\Test.exe"
Hello World!Press any key to continue

ตัวอักษรพิเศษ

- ตัวอักษรที่ใช้ช่วยในการจัดหน้ากระดาษ

```
'\n'// new line
'\r'// carriage return
'\t'// horizontal tab
'\v'// vertical tab
'\b'// backspace
'\f'// formfeed
```

- เครื่องหมายคำพูดและ back slash

```
'\''// single quote (')
'\"'// double quote (")
'\\'// backslash (\)
```