1. Introduction to Programming

Programming ဆိုတာ Code တွေရေးပြီး Computer ကို ခိုင်းစေခြင်းဖြစ်ပါတယ်။ အဲ့ဒီ Code တွေကို ရေးနိုင်ဖို့အတွက် Programming Language တွေကို နားလည် တက်ကျွမ်းဖို့လိုအပ်ပါတယ်။ Computer မှာ Programming Language တွေ များစွာရှိပါတယ်။ သို့သော် Programmer တစ်ယောက် ဖြစ်ဖို့အတွက် Language အကုန်လုံးကိုလေ့လာထားဖို့ မလိုအပ်ပါဘူး။ Programming Language တစ်ခုခုကိုသာလျှင် သေသေချာချာ ဖြစ်ဖြစ်မြောက်မြောက်နားလည် တက်ကျွမ်းဖို့ပဲလိုအပ်ပါတယ်။ Language တစ်ခုကို တတ်မြောက်လျှင် အခြား Language များကို လွယ်လွယ်ကူကူ နားလည် သဘောပေါက်နိုင်ပါတယ်။

Programming နဲ့ ဘာတွေဖန်တီးလို့ရမလဲ?။ Programming ကို အသုံးပြုပြီး ကျွန်တော်တို့ နေ့စဉ် အသုံးပြုနေတဲ့ Mobile App များ၊ Website များ ကိုဖန်တီးနိုင်ပါတယ်။ ဒါအပြင် အခြား Hardware ထိန်းချုပ်မှုများ၊ Robotics System များ၊ AI Development, Game Development စသည့်တို့ကို ဖန်တီးလို့ရပါတယ်။

Programming Language အမျိုးအစားများ

Programming Language တွေရဲ့ Level of Abstraction, Syntax, အသုံးပြုပုံတွေအပေါ်မူတည်ပြီးတော့ Language တွေကို ခွဲခြားထားနိုင်ပါတယ်။

High-Level Programming Languages	Python, Java, Ruby
Low-Level Programming Languages	System-Level Programming
Object-Oriented Programming languages	Java, Python and C++
Functional Programming Languages	Haskell, Lisp
Scripting Languages	Python, PHP, Perl and Ruby
Markup Languages	HTML, XML and CSS
Domain-Specific Languages	SQL, MATLAB, R

Languages တွေကို အထက်ပါအတိုင်း ခွဲခြားထားသော်လည်း အဓိကအားဖြင့် **Complied Language** နဲ့ **Scripting Language** ဆိုပြီး များသောအားဖြင့် Language တွေကို ခွဲခြားနိုင်ပါတယ်။ Scripting Language တွေက များသောအားဖြင့် လွယ်လွယ်ကူကူလေ့လာ တက်မြောက်နိုင်ပါတယ်။

Author : Thura Lin Htut

Complied Languages

Complied Languages ဆိုတာက Code တွေကို ရေးပြီး Run ကြည့်ဖို့အတွက် Compiler တစ်ခု လိုအပ်တဲ့ Languages တွေကိုဆိုလိုတာပါ။ Compiler က ရေးလိုက်တဲ့ Code တွေကို Machine Code (Computer နားလည်တဲ့ Code) အဖြစ်ပြောင်းလဲပေးပါတယ်။ Complied Language တွေက Machine Code အဖြစ် ပြန်ပြောင်းပေးတာကြောင့် Scripting Language တွေထက် ပိုပြီးတော့ မြန်တယ်၊ Efficiency ပိုများတယ်ပေါ့။ Complied Languages များး **– C, C++, Java, Fortran, Swift, Rust, …**

Scripting Languages

Scripting Languages (သို့) Interpreted Languages တွေကို Compiler မပါဘဲ Run နိုင်ပါ တယ်။ Scripting Languages တွေက ရေးလိုက်တဲ့ Code တွေကို တစ်ခုချင်းစီ Line by Line, interpreted လုပ်ပြီးတော့ Code တွေကို Executes လုပ်ပါတယ်။ အဲ့တာကြောင့် Scripting Languages တွေကို Interpreted Languages လို့လဲခေါ်ကြပါတယ်။ Scripting Languages တွေကို အဓိကအားဖြင့် Web Development ပိုင်းနဲ့ Data Analysis ပိုင်းမှာ အသုံးပြုတာများပါတယ်။ Scripting Languages များး – **Python, JavaScript, Ruby, PHP, …**

Program တစ်ခုကို စရေးဖို့အတွက် အနည်းဆုံး ဘာTools တွေလိုအပ်မလဲ?

– Text Editor or IDEs

– Compiler or Interpreter

– Command–line Interface (CLI)

Program တစ်ခုကိုစရေးဖို့အတွက် အထက်ပါ Tools တွေလိုအပ်ပါတယ်။ Code တွေကို Computer မှာ Built In ပါတဲ့ Text Editor တစ်ခုခုနဲ့ အလွယ်တကူရေးလို့ရပါတယ်။ **IDEs (Integrated Development Environments) ဆိုတာ Code ရေးဖို့အတွက် Text Editor တွေကို Compiler/Interpreter, Debugger တွေနဲ့ အထူးပြုလုပ်ထားတဲ့ Software တွေဖြစ်ပါတယ်။** IDE တွေက Code တွေကို လျှင်လျှင်မြန်မြန်ရေးနိုင်အောင်ကူညီပေးပါတယ်။ နောက်တစ်ခုက Code တွေကို Executes လုပ်ဖို့အတွက် Compiler/Interpreter တွေပါ။ **Command–Line Interface ဆို တာ Compiler မှ Executes လုပ်လိုက်တဲ့ Input/Output တွေကို ပြသပေးတဲ့ Tools ဖြစ်ပါတယ်။** ဒီလောက်ဆို Program တစ်ခုကိုစရေးလို့ရပါပြီ။

1.1 Overview of C++

C++ ကို 1980 အစောပိုင်းက Bjarne Stroustrup ဆိုသူက C Languages ကိုအခြေခံပြီးဖန်တီးခဲ့ပါတယ်။ ထို့ကြောင့် C Language မှာရှိတဲ့ Syntax နဲ့ Code Structure တွေက C ++ နဲ့ Similar ဖြစ်ပါတယ်။ System Level Programs , Operating Systems တွေကို ရေးသားရာ၌ C++ ကို ရွေးချယ် ပါတယ်။ C++ ရဲ့ Speed, Flexibility တွေက ကောင်းသောကြောင့် Finance တွေ၊ Game Development တွေနဲ့ High–Performance Computing တွေမှာ အဓိကအသုံးပြုကြပါတယ်။ C++ မှာMemory Management ကို အခြား Language တွေထက် ပိုပြီး Manual လုပ်နိုင်တာပါ။ Memory Manage လုပ်နိုင်တဲ့အတွက်ကြောင့် C++ နဲ့ ရေးတဲ့ Application တွေက ပိုပြီးတော့ Performance ကောင်းတာကို တွေ့နိုင်ပါတယ်။ C++ က Compiled Language တစ်ခုဖြစ်ပါတယ်။ OOP ကို အခြေခံထားတာကြောင့် Complex Program တွေကို ရေးတဲ့အခါမှာ လွယ်ကူစွာ Manage လုပ်နိုင်မှာဖြစ်ပါတယ်။

C++ Compilers

Compiler ဆိုတာ အထက်မှာ ပြောခဲ့သလိုမျိုး၊ Written Code တွေကို Executable Code အဖြစ် ပြောင်းလဲပေးတာဖြစ်ပါတယ်။ C++ မှာ **Open Source Compiler** တွေကော **Commercial Compiler** တွေကော အများကြီးရှိပါတယ်။ Popular ဖြစ်တဲ့ Compiler တွေကို အောက်မှာ ဖော်ပြထားပေးပါတယ်။

Compiler	Description
GCC (GNU Compiler Collection)	Free and open-source compiler that supports multiple programming languages, including C++.
Clang	It is designed to be fast and flexible and is widely used in the development of Apple's operating systems.
Microsoft Visual C++	Commercial compiler, It is widely used in the development of windows application and games
Intel C++	Commercial compiler, It is designed to produce high-performance code and

	used in scientific and engineering communities
Borland C++	Commercial compiler, It is used in the development of Windows application and games.

Compiler တွေကို ရွေးချယ်တဲ့ နေရာမှာ Compiler ရဲ့ Performance, Compatibility, Support Platforms and Technologies စတဲ့ အချက်တွေကို ကြည့်ပြီးရွေးချယ်ရပါမယ်။

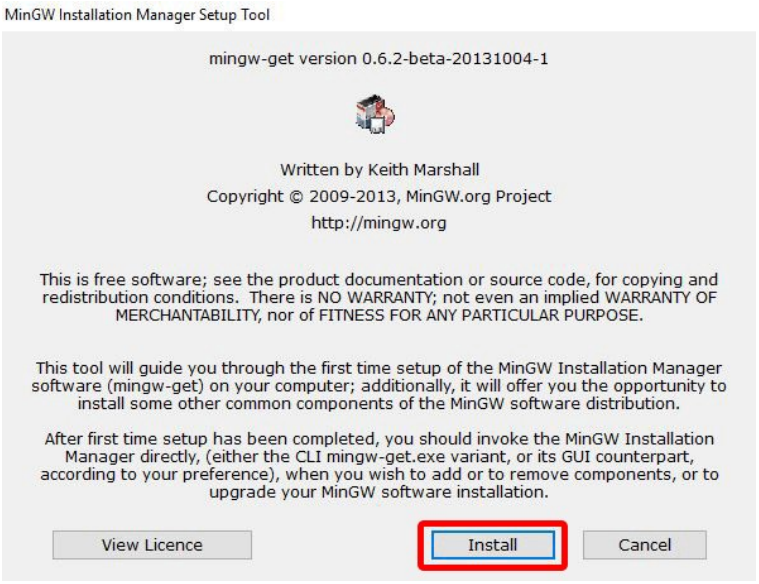
1.2 Environment Setup for C++

C++ Program တစ်ခုကိုမရေးခင် သင်၏ Computer မှာ C++ Compiler တစ်ခုကို Install လုပ်ထားဖို့လိုပါတယ်။ ကျွန်တော်ကတော့ ဒီ စာအုပ်မှာ GNU (GCC) Compiler ကိုပဲ အသုံးပြုသွားမှာ ဖြစ်တာကြောင့် GNU Compiler ကိုပဲ Install လုပ်ပြမှာဖြစ်ပါတယ်။

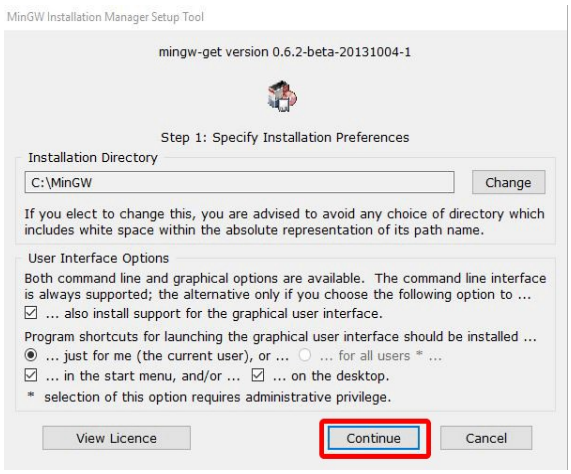
1. GNU (GCC) Compiler ကို Install လုပ်ဖို့ အတွက် အောက်က Link ကို နှိပ်ပြီး ဒေါင်းလုတ်ဆွဲပါ။

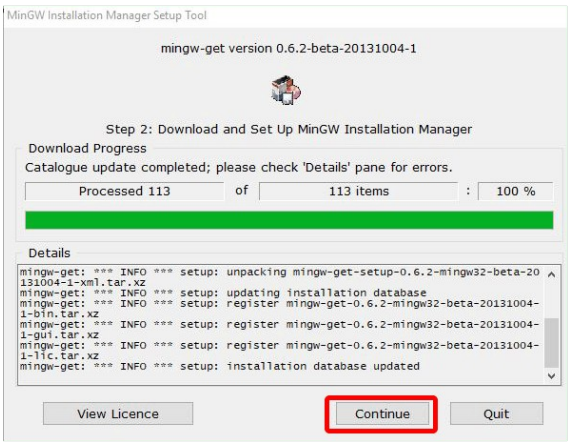
https://sourceforge.net/projects/mingw/files/Installer/mingw-get-setup.exe/download

2. Download ဆွဲပြီးရင် mingw-get-setup.exe file ကို Download Folder ထဲကနေ ဖွင့်ပြီး Install လုပ်ပါ။

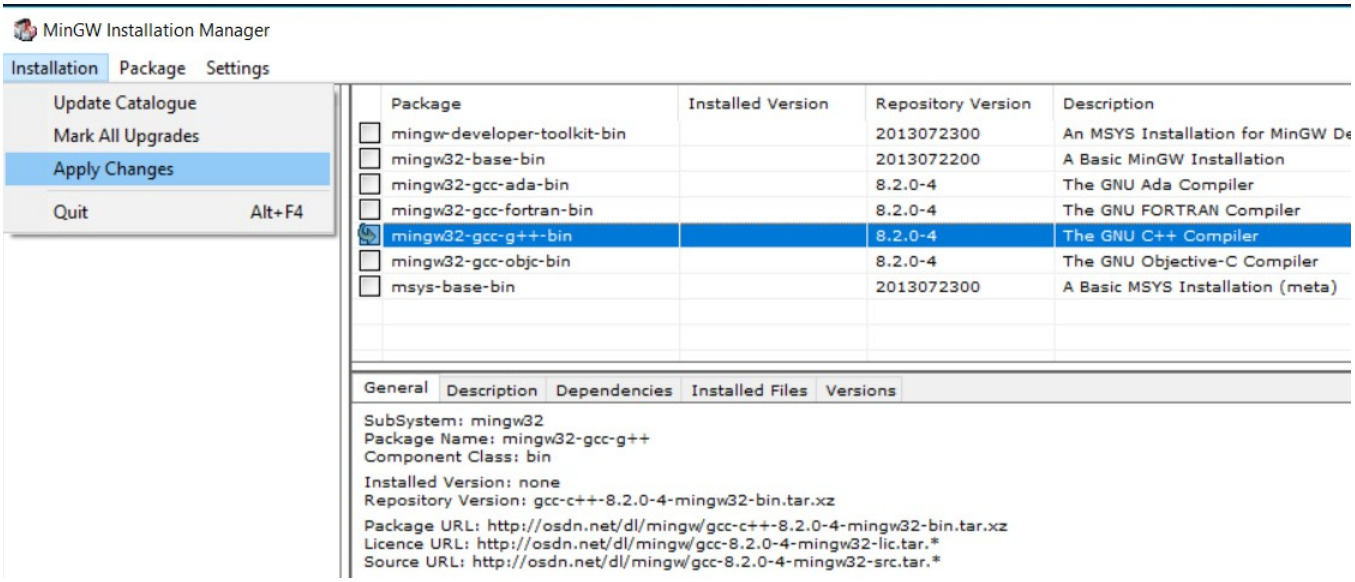


3. Setting တွေကို Default အတိုင်းထားပြီး Continue ကိုသာနှိပ်ပါ။ ပြီးရင် Download ဆွဲနေတာလေးကို ခဏစောင့်လိုက်ပါ။



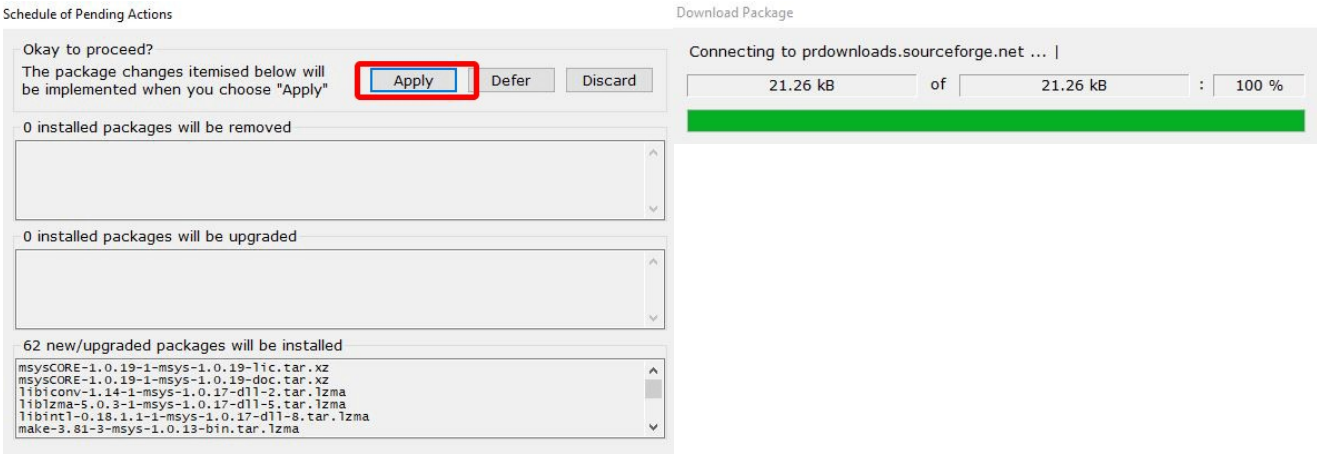


4. ဒီ Screen ကိုမြင်လျှင် mingw32-gcc-g++-bin ဆိုတာ လေးကို ဘေးနားက အစိမ်းမျဉ်းကွေးလေး ပေါ်အောင်နှိပ်ပေးပါ။ ပြီးလျှင်အပေါ်မှာရှိတဲ့ Installation ထဲက Apply Changes ဆိုတာကို နှိပ်လိုက်ပါ။

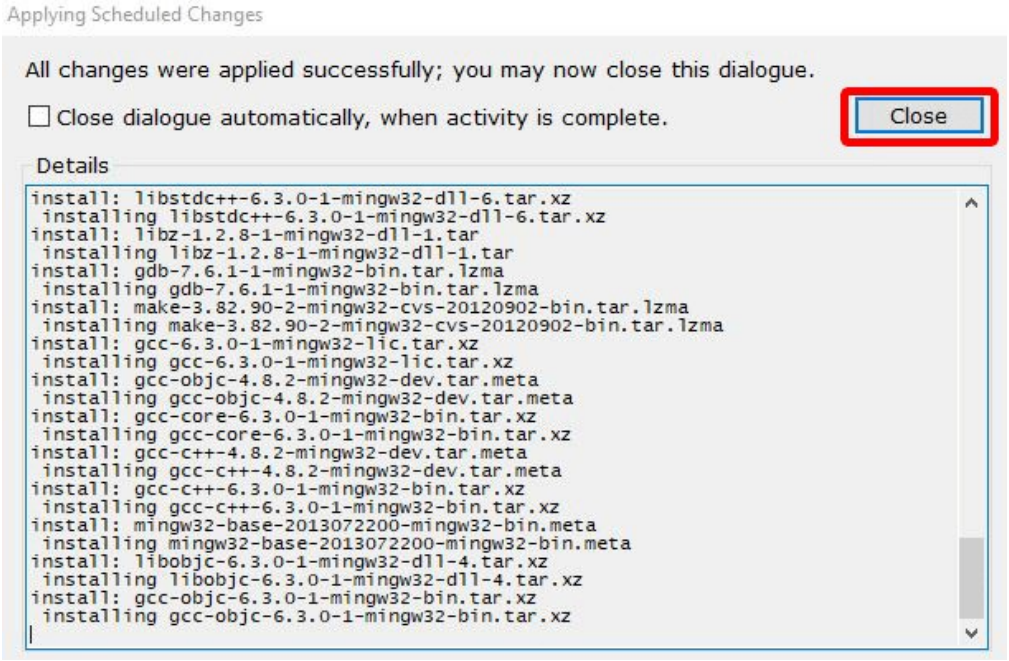


Author : Thura Lin Htut

5. Apply ကိုထပ်နှိပ်ပါ။ Download ဆွဲနေတာလေးကိုခဏစောင့်ပါ။



6. Install ဆွဲနေတာကို ခဏစောင့်ပြီး Close ကိုနှိပ်လိုက်ပါ။ ဒါဆိုရင် GCC compiler ကိုအောင်မြင်စွာ Install လုပ်ပြီးပါပြီခင်ဗျာ။



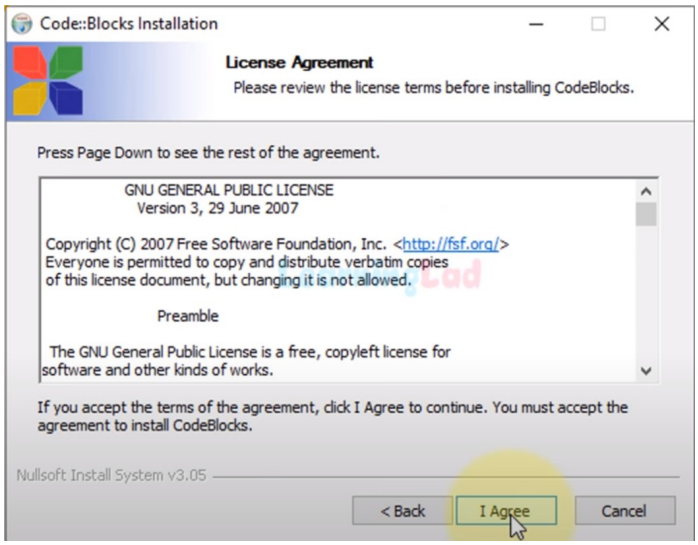
Code::Blocks Installation

Code::Blocks ဆိုတာ C, C++ တို့အတွက် အထူးပြုလုပ်ထားတဲ့ IDE တစ်ခုပါ။ တခြား IDE တွေ သုံးချင်လည်း သုံးလို့ရပါတယ် အရေးကြီးတာက Code ရေးဖို့အတွက်ပါပဲ ကျွန်တော်ကတော့ ဒီစာအုပ်မှာ Code::Blocks နဲ့ပဲ သင်မှာဖြစ်တဲ့အတွက် Code::Blocks ပဲ Install လုပ်ပြပါမယ်။

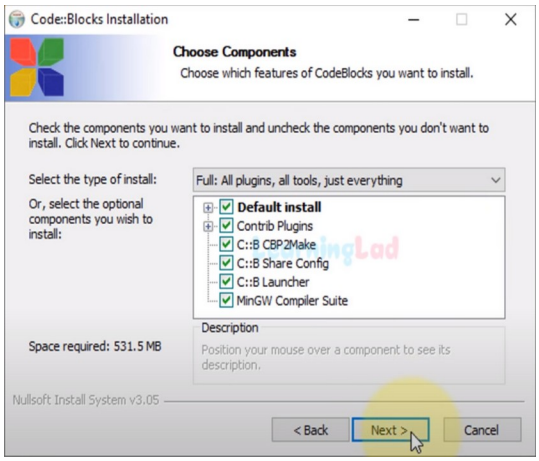
1. Code::Blocks Setup File အားပေးထားသော Link မှ Download ဆွဲပါ။

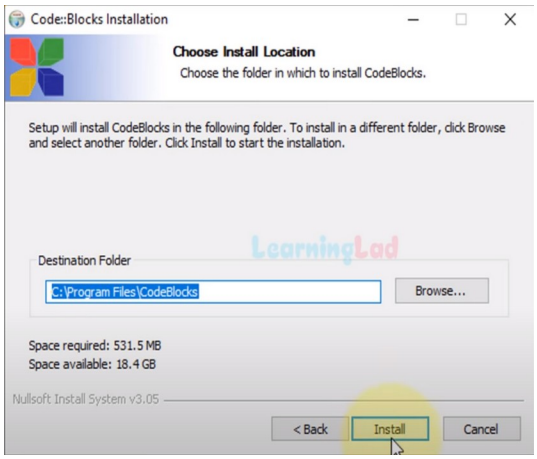
https://sourceforge.net/projects/codeblocks/files/Binaries/20.03/Windows/codeblocks-20.03-setup.exe/download

2. ပြီးလျှင် Code::Blocks Setup File အားဖွင့်၍ Install ဆွဲပါ။

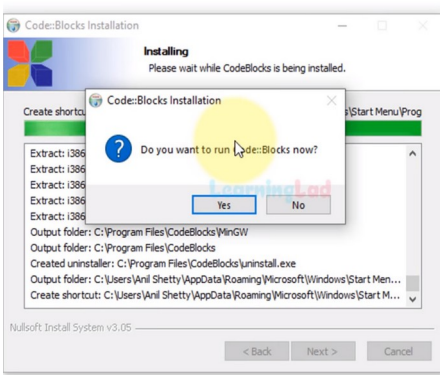


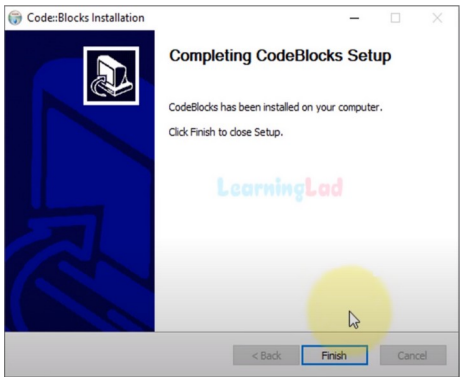
3. Code::Blocks ကို Install ဆွဲတဲ့နေရာမှာ **Next/Install** Button ကိုသာ ဆက်တိုက်နှိပ်ပါ။





4. ဒါဆိုရင် **Install** လုပ်လို့ပြီးပါပြီ။ ပြီးလျှင် **Finish** ကိုနှိပ်ပလိုက်ပါ။





1.3 Basic Structure of C++ Programs

အားလုံး Install လုပ်ပြီးသွားရင် Code::Block ထဲမှာ C++ Programs တစ်ခုကိုစရေးလို့ရပါပြီ။ File > New> Empty File တစ်ခုကိုရွေးလိုက်ပါ။ ပြီးလျှင် File ကို Save ပါ။ Save တဲ့နေရမှာ File Name ရဲ့ နောက်မှာ C++ extension *.cpp ဖြင့် save ပါ။ ဥပမာ – helloworld.cpp ပေါ့။ ဒါဆိုရင် အောက်က Code ကို စရေးလို့ရပါပြီ။

```
#include <iostream>
```

Author : Thura Lin Htut

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

ဒါကတော့ Hello World ဆိုတဲ့ စာသားကို Display ပေါ်မှာမြင်ရအောင် print ထုတ်ပေးတဲ့ Program ပဲဖြစ်ပါတယ်။ #include <iostream> ဆိုတာ Computer ပေါ်မှာ မြင်ရတဲ့ Input/Output တွေကို မြင်ရအောင် ပြုလုပ်ပေးတဲ့ Objects တွေကို Code ထဲမှာ အသုံးပြုလို့ရအောင် Import လုပ်လိုက်တာဖြစ်ပါတယ်။ int main(){... return 0;} ဆိုတဲ့ Functions က C++ မှာ မရှိမဖြစ် ပါဝင်ရမယ့် Functions တစ်ခုဖြစ်ပါတယ်။ C++ Program တစ်ခုကို စ run လိုက်တာနဲ့ int main() ဆိုတဲ့ Function ထဲမှာရှိတဲ့ Code Block တွေကို အရင်စ Run တာဖြစ်ပါတယ်။ std::cout << "Hello, World!"; ဆိုတာကတော့ Hello, World! ဆိုတဲ့ စာသားကို Display ပေါ်မှာ မြင်ရအောင် ထုတ်ပေးတဲ့ Syntax ပဲဖြစ်ပါတယ်။

1. Code တွေရဲ့ syntax တစ်ခုဆုံးတိုင်း Semicolon တွေကို ထည့်ဖို့လိုအပ်ပါတယ်။

```
std::cout << "Hello World" ;
```

2. C++ မှာပါဝင်တဲ့ Built In Functions တွေ, Data Types တွေ, Code Syntax တွေကို မှားလို့မရပါဘူး။ မှားမယ်ဆိုရင် Complier မှာ Errors တွေပြပါတယ်။ Error တွေကို ဖတ်တက်ဖို့လဲ အရေးကြီးပါတယ်။

```
std:: cout < "Hello World";
```

```
error: no match for 'operator<' (operand types are 'std::ostream' {aka 'std::basic_ostream<char>'} and...
note: candidate: 'template<class _T1, class _T2> constexpr bool std::operator<(const std::pair<_T1, _T...
note:   template argument deduction/substitution failed:
note:   'std::ostream' {aka 'std::basic_ostream<char>'} is not derived from 'const std::pair<_T1, _T2>'
note: candidate: 'template<class _Iterator> constexpr bool std::operator<(const std::reverse_iterator<...
```

3. ရေးပြီးသား Code တွေကို မ Run ခင် Compiler မှာ Code ကို Build လုပ်ပါတယ်။ Build လုပ်တယ်ဆိုတာ Readable Code တွေကို Exectuable Code (Machine Code) အဖြစ်ပြောင်းလဲပေးတာဖြစ်ပါတယ်။

4. C++ မှာ Code တွေကို စ Run လိုက်တာနဲ့ main () ဆိုတဲ့ function ထဲမှာရှိတဲ့ Code တွေကို Line by Line စ Run ပါတယ်။ ဒါကြောင့် ကိုယ်လုပ်ချင်တဲ့ Code အစီအစဉ်တွေကို Line by Line ရေးဖို့ အရေးကြီးပါတယ်။ ဥပမာ –

1. အိပ်ယာထမယ်

2. သွားတိုက်မယ်

3. မနက်စာစားမယ်

ကိုယ်ဖြစ်ချင်တဲ့ အကြောင်းအရာတွေကို အစဉ်လိုက်ရေးတက်ဖို့ အရေးကြီးပါတယ်။ ဒါပြီးရင် ဒါလုပ်ဆိုတာ ကို Computer က နားလည်ဖို့အတွက်ပါ။

1.4 C++ Comments

ကျွန်တော်တို့ Program တွေ Projects တွေကို Team တွေနဲ့ အတူတူလုပ်တဲ့ အခါမှာ ကိုယ့်ရဲ့ Code တွေက Easy to Read ဖြစ်ဖို့အရေးကြီးပါတယ်။ ကိုယ်ရေးတဲ့ Code တွေက Complex ဖြစ်နေတဲ့ အခါမှာ ဘယ်လိုအလုပ်လုပ်သွားလဲဆိုတာကို ရှင်းပြတာပိုပြီးတော့ ကောင်းပါတယ်။ ဒီလိုရှင်းပြဖို့ အတွက် Comments တွေကို အသုံးပြုရပါမယ်။ Comments တွေက C++ Complier ထဲမှာ Execute မလုပ်ပါဘူး ဒါကြောင့် ကြိုက်သလိုရေးလို့ရပါတယ် Errors မတက်ပါဘူး။

Single-line Comments

Comment တစ်ကြောင်းထဲရေးတာပါ။ Comments တစ်ကြောင်းထဲ ရေးရင် အရှေ့မှာ Slashes (//) Symbol လေးခံပေးရပါမယ်။

```
//This is Hello World Program

cout << "Hello World!";
```

Multi-line Comments

Comments တွေအများကြီးရေးချင်တယ်ဆိုရင် /* နဲ့စပြီး စာကြောင်းအဆုံးမှာ */ အဆုံးသတ်ပေးရပါတယ်။

```
/*Multil line comments will multi line

Hello World Program

*/

cout << "Hello World!";
```

2. Using Input and Output

Computer Program တွေ Software တွေရဲ့ ရည်ရွယ်ချက်က User Input တွေကို လက်ခံမယ် ပြီးတော့ User Input တွေကနေ Data တွေကို Processing လုပ်ပြီးရင်User ရှိကို Output လုပ်ပေးမယ် ဒါကြောင့် Program တစ်ခုမှာ ပါသင့်တဲ့ အချက်တစ်ခုကတော့ Input နဲ့ Ouput ပါပဲ။ ဒါကို ကျွန်တော်တို့က C++ Language နဲ့ User တွေရှိကနေ Input ယူမယ်၊ ပြီးရင် Output ပြန်ထုတ်ပေးလို့ရပါတယ်။ အဲ့တာဆိုရင် Input/Output Library ကို ကျွန်တော်တို့ ထည့်ရပါမယ်။ C++ ရဲ့ Input/Output Library က <iostream> ပါ။ ဒါကို C++ program ထဲကို ထည့်မည်ဆိုလျှင် #include <iostream> ဆိုပြီး ထည့်ရပါမယ်။ အဲ့ဒီ Input/Output Library ကိုထည့်မှာ program မှာ သူနဲ့ သက်ဆိုင်တဲ့ cin , cout ဆိုတဲ့ syntax တွေကို အသုံးပြုလို့ရမှာဖြစ်ပါတယ်။

2.1 Using cout

cout ဆိုတဲ့ syntax ကို ကျွန်တော်တို့ Hello World Program မှာ အသုံးပြုထားပါသေးတယ်။ အဲ့တော့ cout ဆိုတာက User ရှိကို Ouput ထုတ်ပြပေးတာဖြစ်ပါတယ်။ cout ဆိုတဲ့ syntax က <iostream> ဆိုတဲ့ Standard Libarary ထဲမှာပါပါတယ်။ အဲ့တာကြောင့် cout ကို အသုံးပြုဖို့ဆိုရင် #include <iostream> ဆိုပြီး Code ရဲ့ အပေါ်ပိုင်းမှာ ရေးရပါမယ်။ ဒါဆိုရင် ကျွန်တော်တို့ Simple Program တစ်ခုစရေးကြည့်ပါမယ်။

1. File > New > Empty File ပြီးရင် filename တစ်ခုပေးပြီး save လိုက်ပါ။ ဥပမာ – output.cpp, File Save ပြီဆိုရင် Filename Extensions ကို .cpp ပါအောင် save ရပါမယ်။

2. ဒါဆိုရင် အောက်က Code ကို စရေးလို့ရပါပြီ။

```
#include <iostream>
int main(){
	std::cout << "I am output" << std::endl;
	std::cout << "I am output too" << std::endl;
	return 0;
}
```

3. #include <iostream>

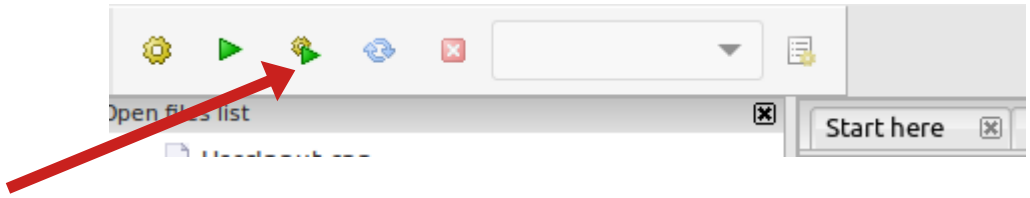
<iostream> ဆိုတဲ့ Library ကို Import လုပ်တယ်။

4. int main(){ .... return 0; } ဆိုတာက Code တွေကို စ Run မဲ့ main function ဖြစ်ပါတယ်။ Code တွေကို main function ရဲ့ { } ထဲမှာ စရေးရပါမယ်။

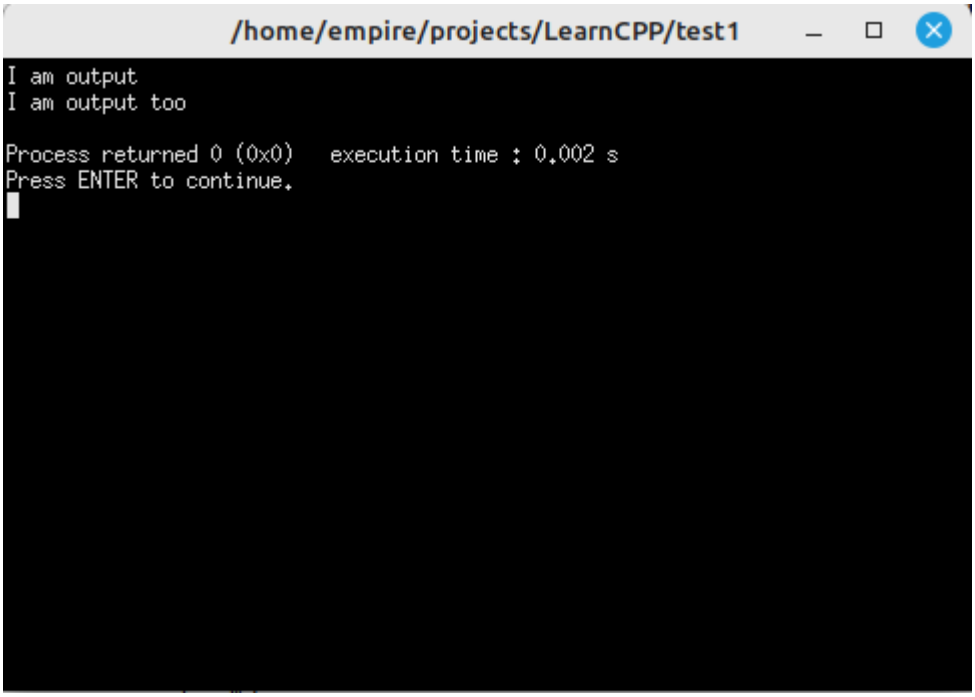
5. std::cout << "I am output" << endl; ဆိုတာက တော့ "I am output" ကို ထုတ်ပေးတာဖြစ်ပါတယ်။ std:: ဆိုတာကတော့ Standard Library တွေကို အသုံးပြုတဲ့ အခါမှာ ရေးပေးရပါတယ်။ cout ဆိုတာက Standard Library ဖြစ်တဲ့အတွက်ကြောင့် cout ကို အသုံးပြုဖို့အတွက် std:: ဆိုတာကို cout ရဲ့ အရှေ့မှာ ရေးပေးဖို့လိုပါတယ်။

cout ဆိုတဲ့ syntax က << နောက်မှာရှိတဲ့ စာသားတွေကို output လုပ်ပေးပါတယ်။ ဒါကြောင့် "I am output" ဆိုတာကို Ouput မှာ ထုတ်ပေးပါတယ်။ std::endl; ဆိုတာကတော့ စာကြောင်း ရဲ့နောက်မှာ Enter ခေါက်တာ ဖြစ်ပါတယ်။

6. ဒါဆိုရင် Code ကို စ Build ပြီးတော့ Run လို့ရပါပြီ။ Code::Blocks မှာဆိုလျှင် F9 ကိုနှိပ်ပြီးတော့ Build and Run လုပ်လို့ရပါတယ်။



7. Ouput ကို ပိတ်မယ်ဆိုလျှင် Enter ကိုခေါက်ပါ။



2.2 Using cin

ဒါဆိုရင် User Input တွေကို ယူမယ်ဆိုရင် cin ဆိုတာ ကို အသုံးပြုရမှာဖြစ်ပါတယ်။ cin ဆိုတာကလည်း Standard Library ဖြစ်တာကြောင့် cin ရဲ့ ရှေ့မှာ std:: ဆိုတာကို ထည့်ပေးရပါမယ်။ cin ကို သုံးမယ် ဆိုရင် >> ဆိုတဲ့ Operator ကို အသုံးပြုရပါမယ်။ << ဆိုတဲ့ Operator က cin ထဲက User

Input Data တွေကို export လုပ်လိုက်တာဖြစ်ပါတယ်။ cin ဆိုတာက User Input လုပ်လိုက်တဲ့ Data ထဲက One Word ကိုသာ export လုပ်ပေးပါတယ်။ ဥပမာ - User က "Hello World" လို့ရိုက်လျှင် "Hello" ဆိုတာကို သာ export လုပ်ပါတယ်။

1. ဒါဆိုရင် File တစ်ခုကို ဆောက်ပြီး Code စရေးကြည့်ကြပါမယ်။

```
#include <iostream>
int main()
{
    std::string text;
    std::cout << "Type Something : ";
    std::cin >> text;
    std::cout << text << std::endl;
    return 0;
}
```

2. ဒါဆိုရင် Code တွေကို ရေးနေကြ Code Structure တိုင်းရေးပါမယ်။ #include , main () တွေရေးမယ်ပေါ့။

3. std::string text; ဆိုတာက String Variable တစ်ခုကို Declare လုပ်လိုက်တာပါ။ Variable အကြောင်းတွေကို နောက်အခန်းတွေမှာရှင်းပြပေးထားပါတယ်။ လောလောဆယ်တော့ Data Store လုပ်ဖို့အတွက်သာမှတ်ထားလိုက်ပါ။

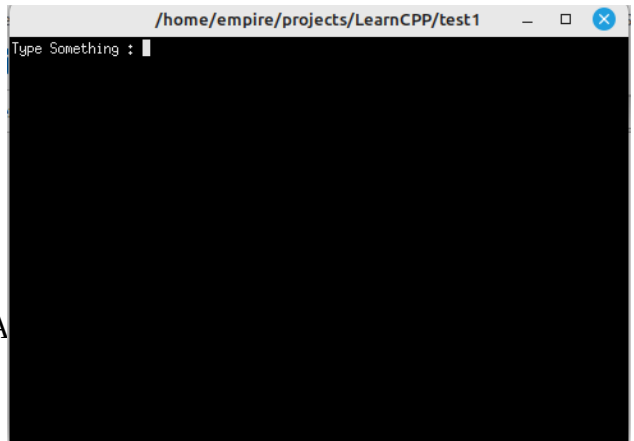
4. std::cout << "Type Something : " , User ကို ဘာလုပ်ရမလဲ ညွှန်ပြဖို့အတွက် ထည့်လိုက်တာပါ။ မထည့်လည်းရပါတယ်။

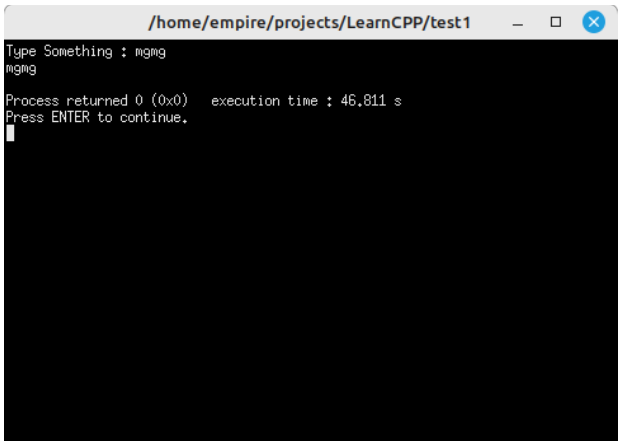
5. std::cin >> text; User က စာလုံး တစ်ခုခုရိုက်ပြီး Enter ခေါက်လိုက်မယ်ဆိုရင် text ဆိုတဲ့ variable ထဲကို cin ကနေ export လုပ်လိုက်တာပါ။ အဲ့တော့ User Input လုပ်လိုက်တဲ့ စာလုံးတွေ အားလုံးက text ဆိုတဲ့ variable ထဲမှာ Store လုပ်သွားပါပြီ။

6. std::cout<< text ; store လုပ်တဲ့ data တွေကို ပြန်ပြီးတော့ Output ထုတ်လိုက်ပါမယ်။

7. အဲ့တော့ Cli မှာ စ Run လို့ရပါပြီ။

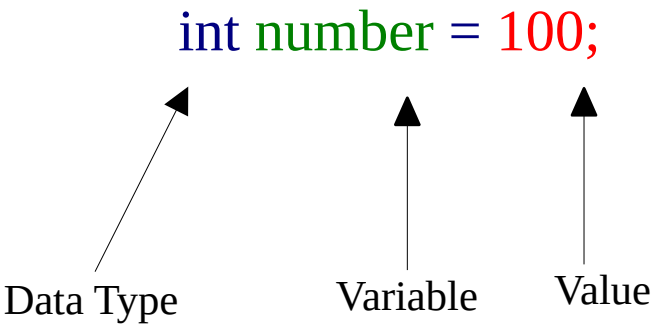
A





3. Data Types and Variables

C++ အပါဝင် အခြား Programming Language တိုင်းမှာ အရေးကြီးတဲ့ အစိတ်အပိုင်းတစ်ခုကတော့ Data Types တွေနဲ့ Variable တွေပဲ ဖြစ်ပါတယ်။ Data Types, Variables တွေက Computer ရဲ့ Memory ထဲမှာ Data တွေကို Store လုပ်တယ် ပြီးတော့ Variables တွေက Manipulation လုပ်ဖို့ အတွက် သုံးပါတယ်။



3.1 Data Types

Computer ရဲ့ Memory ပေါ်မှာ Data တွေ ဘယ်လိုပုံစံနဲ့ Store လုပ်မယ်ဆိုတာကို Data Types တွေနဲ့ ခွဲခြားနိုင်ပါတယ်။ ဥပမာ - စာသားတွေနဲ့ Memory ပေါ်မှာ Store လုပ်မှာလား၊ ဂဏန်းတွေနဲ့ Store လုပ်မှာလားဆိုတာကို ခွဲခြားနိုင်ပါတယ်။.

Example

```
int number = 123;
double DoubleNum = 6.58;
float floatNum = 7.66;
char letter = 'C';
bool boolean = true;
string text = "Hello";
```

အောက်ကဇယားထဲမှာ C++ မှာ built in ပါတဲ့ Data Type တွေပြထားပေးပါတယ်။

Data Type	Size	Description
`int`	2 or 4 bytes	Stores whole numbers, without decimal
`double`	8 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 15 decimal digits
`float`	4 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 6 – 7 decimal digits
`char`	1 byte	Stores a single character/letter/number, or ASCII values
`boolean`	1 byte	Stores true or false values

Author : Thura Lin Htut

အပေါ် Example မှာ ပြထားတဲ့ string Variable က C++ ရဲ့ Built In Data Type မဟုတ်ပါဘူး။ အဲ့တာကြောင့်သူ့ကို Library အနေနဲ့ ထည့်ရပါမယ်။ String ကို Library အနေနဲ့ ထည့်မယ်ဆိုရင် header ပိုင်းမှာ #include <string> ဆိုပြီးရေးရပါမယ်။

Data Type	Size	Description
string	-	Store a sequence of characters, such as letters, digits, and symbols.

3.2 Variables

ကျွန်တော်တို့ Data Types တွေကို Declare လုပ်ယုံနဲ့ Memory ပေါ်မှာ Store လုပ်လို့မရပါဘူး။ Memory ပေါ်မှာ Variables တွေကို အမည်ပေးပြီး Store လုပ်ရပါမယ်။ ဒါမှ Variable Name တွေကို ခေါ်ပြီး Processing ပြန်လုပ်လို့ရမှာဖြစ်ပါတယ်။ **Variables ဆိုတာ Data Types တွေကို အမည်ပေးပြီး Memory ပေါ်မှာ Store လုပ်တာဖြစ်ပါတယ်။** Variables တွေက နာမည်ပေးတဲ့ နေရာမှာ -

1 . ပေးပြီးသား Name တွေပြန်ပေးလို့မရပါဘူး။ (Name တစ်ခုနဲ့ တစ်ခုတူလို့မရပါဘူး)

2. Value နဲ့သက်ဆိုင်တဲ့ နာမည်တွေကိုပေးရပါမယ်။ (Code တွေကို ဖတ်ရလွယ်ရမယ်)

3. C++ ရဲ့ keywords တွေနဲ့ ညှိလို့မရပါဘူး။

4. ဂဏန်းတွေနဲ့ စလို့မရပါဘူး။ စာသားတွေနဲ့ စရေးရပါမယ်။

```
string 1name = “mgmg”;
string name1 = "mgmg";
```

5. white spaces တွေပါလို့မရပါဘူး။ Special Characters တွေပါလို့မရပါဘူး။ !@#$%&...

6. ကျန်တာကတော့ name တွေကို အဆင်ပြေသလို ပေးလို့ရပါတယ်။

3.2.1 Declare Multiple Variables

တူညီတဲ့ Data Types တွေသုံးတဲ့နေရာမှာ Variables တွေကို ကော်မာခံပြီးတော့ ကြောငြာ လို့ရပါတယ်။

```
int x = 10, y = 15, z = 20;
cout << x + y + z ;  // output : 45
```

3.2.2 One Value to Multiple Variables

```
 int x, y, z;
x = y = z = 25;
cout << x + y + z; //output : 50
```

3.3 Declaring Constant Variables

Variables Value တွေကို ပြန်မချိန်းချင်ဘူး ပုံသေ သတ်မှတ်ထားတဲ့ Variables အတိုင်းပဲသုံးချင်တယ်ဆိုရင် **const** keyword ကို variables ရဲ့ ရှေ့မှတ်ထည့်ပေးရပါမယ်။ const keyword ကို variables ရဲ့ ရှေ့မှာ သုံးလိုက်မယ်ဆိုရင် ၎င်း Variable က Read Only ဖြစ်သွားပြီ၊ Value တွေကို ပြန် change လို့မရတော့ပါဘူး။

```
const int myNum = 22;
cout << MyNum // Output : 22
MyNum = 10; // error
```

Before Const

```
int myNum = 22;
MyNum = 10;
```

After Const (Error)

```
const int myNum = 22;
MyNum = 10; // error
```

3.4 C++ Keywords

alignas	asm	auto	bool	break	case	catch
char	char16_t	char32_t	class	concept	const	consteval
constexpr	const_cast	continue	co_await	co_return	co_yield	decltype
default	delete	do	double	dynamic_cast	else	enum
explicit	export	extern	false	float	for	friend
goto	if	inline	int	long	mutable	namespace
new	noexcept	nullptr	operator	private	protected	public
register	reinterpret_cast	requires	return	short	signed	sizeof
static	static_assert	static_cast	struct	switch	template	this
thread_local	throw	true	try	typedef	typeid	typename
union	unsigned	using	virtual	void	volatile	wchar_t
while						