What to Add or Edit

- Make all the edits for “C” rather than “C++”
- Add Commons License Information
- Add better “objective” slide
- Add “what is needed” slide

Programing C Chapter 0

CPA: PROGRAMING IN C

Developed As Additional Instructor Led Slides For CPA: Programing Essentials In C

Cisco NetAcademy https://www.netacad.com/

Supplemental Slides Developed by topClown@SteamClown.org

Chapter 1 Objectives

After completing this module, the student will be able to:

- Explain how a sample C++ program works
- Explain the concept of include and using directives
- Explain the concept of integers, floating-point numbers, operators and arithmetic operations in C++ programming
- Discover and fix basic syntax errors
- Modify the structure of a C++ program
- Perform basic calculations
- Understand the precedence and associativity of C++ operators and the proper use of parentheses
- Use the shortcut and pre/post increment/decrement operators
- Build simple expressions
- Translate verbal description into programming language
- Test code using known input and output data
- Compare values using relational operators
- Build Boolean expressions using logical operators

Where to Begin?

- Every Creative Activity Needs Tools
- Many Factors Affecting Programing and Compiling Tools:
  - Hardware platform
  - Operating system
  - Operating system version
- Location Of Tools
  - Locally installed IDE
  - On-line tools

What is an IDE?

- **IDE (Integrated Development Environment)**
- Software application that typically consists of a
  - Code editor
  - Compiler
  - Debugger
  - and a graphical user interface (GUI) builder

eclipse

Visual Studio

Code::Blocks

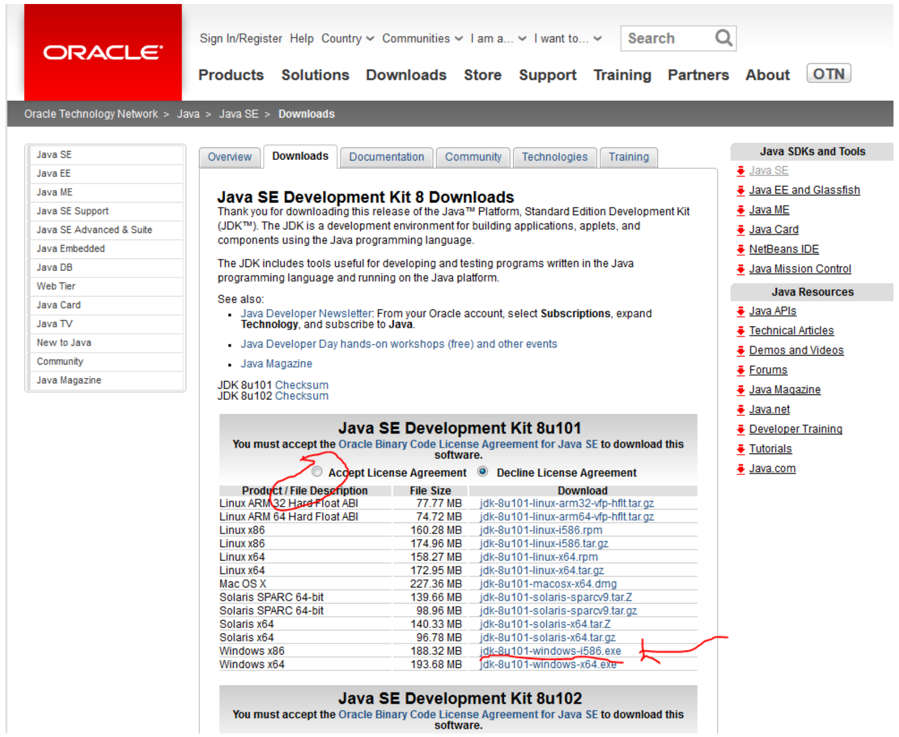
NetBeans

Local or On-Line

- Local IDE has many advantages:
  - toolkit containing everything you may need
  - Real programmers usually use an IDE too
  - An IDE gives tools and apps in one place
- Local IDE disadvantages:
  - May consume a lot of resources
  - Don’t need most of the functions they can perform.
- On-line tools allows
  - Write, store and run your code without installing anything
  - Simplified IDE accessible remotely via the Internet
  - Required: an Internet browser and Internet access.

CHOOSE YOUR IDE

- Oracle JDK
  - http://www.oracle.com/technetwork/java/javase/downloads/index.html
- NetBeans
  - https://netbeans.org/



Java SE Development Kit 8 Downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, applets, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

See also:

- Java Developer Newsletter: From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK 8u101 Checksum
JDK 8u102 Checksum

Java SE Development Kit 8u101

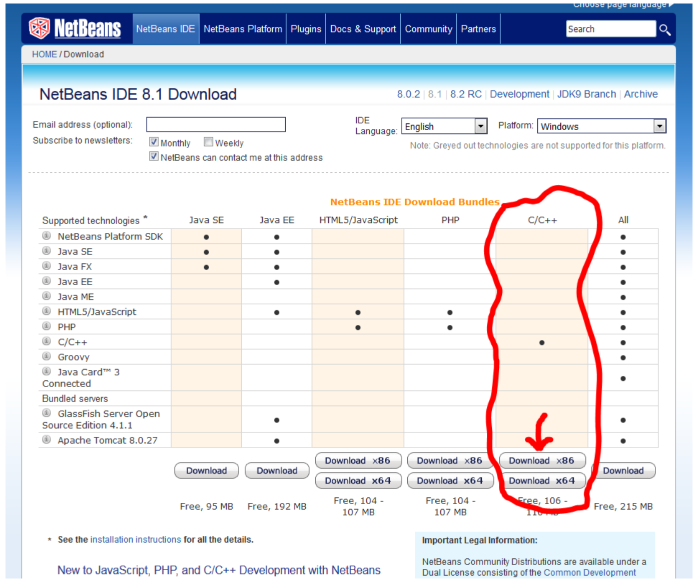
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement [✓ Decline License Agreement]

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.77 MB	jdk-8u101-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.72 MB	jdk-8u101-linux-arm64-vfp-hflt.tar.gz
Linux x86	160.28 MB	jdk-8u101-linux-i586.rpm
Linux x86	174.96 MB	jdk-8u101-linux-i586.tar.gz
Linux x64	158.27 MB	jdk-8u101-linux-x64.rpm
Linux x64	172.95 MB	jdk-8u101-linux-x64.tar.gz
Mac OS X	227.36 MB	jdk-8u101-macosx-x64.dmg
Solaris SPARC 64-bit	139.66 MB	jdk-8u101-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.96 MB	jdk-8u101-solaris-sparcv9.tar.gz
Solaris x64	140.33 MB	jdk-8u101-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u101-solaris-x64.tar.gz
Windows x86	188.32 MB	jdk-8u101-windows-i586.exe
Windows x64	193.68 MB	jdk-8u101-windows-x64.exe

Java SE Development Kit 8u102

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.


Choose page language
NetBeans
NetBeans IDE NetBeans Platform Plugins Docs & Support Community Partners Search
HOME / Download
8.0.2 | 8.1 | 8.2 RC | Development | JDK9 Branch | Archive

NetBeans IDE 8.1 Download

Email address (optional):
Subscribe to newsletters: [✓ Monthly] Weekly
[✓ NetBeans can contact me at this address]

IDE Language: English
Platform: Windows

Note: Greyed out technologies are not supported for this platform.

NetBeans IDE Download Bundles

Supported technologies *	Java SE	Java EE	HTML5/JavaScript	PHP	C/C++	All
NetBeans Platform SDK	●	●				●
Java SE	●	●				●
Java FX	●	●				●
Java EE		●				●
Java ME						●
HTML5/JavaScript		●	●	●		●
PHP			●	●		●
C/C++					●	●
Groovy						●
Java Card™ 3 Connected						●
Bundled servers						
GlassFish Server Open Source Edition 4.1.1		●				●
Apache Tomcat 8.0.27		●				●
	Download	Download	Download x86 Download x64	Download x86 Download x64	Download x86 Download x64	Download
	Free, 95 MB	Free, 192 MB	Free, 104 - 107 MB	Free, 104 - 107 MB	Free, 106 - 110 MB	Free, 215 MB

* **See the installation instructions for all the details.**

New to JavaScript, PHP, and C/C++ Development with NetBeans

Important Legal Information:

NetBeans Community Distributions are available under a Dual License consisting of the Common Development

On-Line IDE → ideone.com

- http://ideone.com/
- At this point it's also good to install adblock+… so you don't have to see all the adds.
- Also it seem to only work in IE and Firefox… 'cause Chrome does not run Java plugins

Natural Language vs. Programming Language

- language is a tool for expressing and recording human thoughts
- programming languages Have Specific Structures
- Lexicon
  - Set of rules determine which symbols (letters, digits, punctuation marks, and so on) could be used in the language
- syntax
  - Set of rules determines the appropriate ways of collating the symbols
- semantics
  - recognize the meaning of every statement expressed in the given language

Error Free…

Any program we write must be error-free in these three ways:

- **Lexically**
- **Syntactically**
- **Semantically**

This is because the message embedded inside a computer program is not intended for a human, but for a machine.

Technically sophisticated, but devoid of even a trace of intelligence

- Computers respond only to a predetermined set of known commands
  - Instruction list
- Machine language
  - tedious, time-consuming to code by hand
  - highly susceptible to a programmer's mistakes
  - difficult to understand for humans
- High-level programming language, Like C++
  - bridge between the people's language (natural language) and computer language (machine language)
  - an intermediate common language for both humans and computers working together
- Portability
  - translated into any number of different machine languages

Compiler

- The translation we are referring to is made by a specialized computer program called a **compiler**. The process of translating from a high-level language into a machine language is called **compilation**.

More Reading

1.2.1 – Your First Program

Add Slides for First Program

```
#include <stdio.h>
int main(void)
{
  puts("Hi, I'm your first Program");
  return(0);
}
```

See GITHUB code_1_2_1

Lab 1.2

- Make some changes to yourFirstProgram

Edit this text

```
#include <stdio.h>
int main(void)
{
  puts("Hi, I'm your first Program");
  return(0);
}
```

1.3.1 - Numbers and How Computers See Them

Numbers & How Computers See Them

- binary system
  - system computers use for storing numbers
  - can perform any operation upon them
- Type: integers vs floating-point
  - the characteristic of a number which determines its kind, range and application
- integers
  - whole numbers or those which are devoid of the fractional part,
- floating-point
  - numbers (or simply floats) that contain (or are able to contain) the fractional part.

Integers

- How does the C & C++ language recognize the integers?
- The same as when you write them on a piece of paper
- They're simply **a string of digits** that make up a number.
- But there's a catch – you can't include any characters that are not digits inside the number.
  - 12,392,267 X
  - 12.393.267 X
  - 12393267 √

More on Integers

- Positive numbers don’t need to be preceded by the plus sign, but you can do it if you want. The following lines describe the same number:
  - +123 √ (though not typical)
  - 123 √

Octal vs Hex

- Octal
  - If an integer number is preceded by the 0 digit, it will be treated as an octal value
  - must contain digits taken from the 0 to 7 range only
  - **0123** This is an octal number with the (decimal) value equal to 83
- Hex
  - Hexadecimal numbers. Such number should be preceded by the prefix written as 0x or 0X
  - **0x123** is a hexadecimal number with the (decimal) value equal to 291

Variables

- Special “containers” for that purpose and these containers are called **variables**
- As the name ***variables*** suggests, the content of a container can be varied in (almost) any way
- What does every variable have?
  - a name
  - a type
  - a value

Variable Names

- the name of the variable can be composed of upper-case or lower-case Latin letters, digits and the character **_** (underscore),
  - ABCdef_ghi
  - abcDEF_GHI
  - A123_456
  - a123_456
- the name of the variable **must begin with a letter**,
- the underline character is a letter (strange but true),
- upper- and lower-case letters are treated as different
  - Alice and ALICE are **<u>different</u>**

Variable Names - Cont

- Programing Style Guide And Conventions
- This is what I do:
    - myVariable ← start lowercaseNoSpaceUpercase
    - myVariable_1 ← start lowercaseNoSpaceUpercase_ThenSomeTimesASpace
- Make Your variableNames mean something!
    - They can be very long…
- Which do you like?
    - i
    - t10
    - Exchange_Rate ← exchangeRate
    - counter
    - DaysToTheEndOfTheWorld ← daysToTheEndOfTheWorld
    - TheNameOfAVariableWhichIsSoLongThatYouWillNotBeAbleToWriteItWithoutMistakes
    - _thisVariableHasA_AtTheFront

Variable Names - cont

- Which do you like?
  - i
  - t10
  - Exchange_Rate ← exchangeRate
  - counter
  - DaysToTheEndOfTheWorld ← daysToTheEndOfTheWorld
  - TheNameOfAVariableWhichIsSoLongThatYouWillNotBeAbleToWriteItWithoutMistakes
  - _thisVariableHasA_AtTheFront

- What's Wrong?
  - 10t
  - Adiós_Señora
  - Exchange Rate

Type Attribute

- **type** is an **attribute** that uniquely defines which values can be stored inside the variable
  - Only an integer value can be assigned to an integer variable (int)
  - The compiler will not allow a floating-point number for type (int)
- Variable exists as a result of a **declaration**
  - syntactic structure that binds a **variable name**, with a specific **type** offered by the C++ language

Declaring a Variable

- Declare a variable of type int named *counter*
  - **int** counter;
- What is declared by the following fragment of a program?
  - **int** variable1, accountBalance, invoices;
  - declares three variables of type *int* named (respectively) *variable1*, *accountBalance* and *invoices*
- You are allowed to use **as many variable declarations as you need**

Assigning a Value to a Variable

- **assignment operator… wait for it…**
  - =
- examples:
  - counter = 1;
- The above statement says: *assign the value of 1 to a variable named Counter or, a bit shorter, assign 1 to Counter*.
- What is the default value of
  - int counter;
  - Go find out…

This is where I still need to update with "C" rather than "C++" syntax…

Default Values

```
// This code is to show that the default assignment of a variable,
// is compiler dependent, and the default could be 0 or something else...
// like 134514688 -- what is that number?
// ------------------------------------------------------------------------
#include <iostream>
using namespace std;
int main(void) {
		int variableWithValueAssigned = 3;
		int variableWithValueNotAssigned;
		cout << "\n this is the value of variableWithValueAssigned \t ";
		cout << variableWithValueAssigned;
		cout << "\n this is the value of variableWithValueNotAssigned \t ";
		cout << variableWithValueNotAssigned;

return 0;
}
```

See GITHUB code_1_3_5

Lab 1.3

Floating Point 1.4.1

Floating Point Numbers

- data type
  - int
  - Float
- Designed to represent and store the numbers that have **a non-empty decimal fraction**
- Can have a fractional part after the decimal point
- “two and a half” or “zero point four”
- 2.5 or 0.4 or just .4

What's the difference? It's just a "."

- 4.0 could be written as 4.
- But 4 and 4.0 are different types
  - **4** is an **int**.
  - **4.0** is a **float**.
- We can say that **the point makes a float**. Don't forget that.

4.

Exponents are "Type" Float Too

- $300000000 = 3 \cdot 10^8$
  - It means: *three times ten to the power of eight*
- In C++ it's represented as **3E8**
- The letter **E** or **e** is the **exponent**
  - *times ten to the power of*
  - the exponent (the value after the "E") **must be** an integer.
  - the base (the value in front of the "E") **may or may not be** an integer.

Big or Small Floats are the Way to Go

- Planck's constant, is a very small number 6.62607 · 10-34
  - float planks = 6.62607E-34;
- $503 billion is the current US deficit, because the government spending of $4.147 trillion is higher than its revenue of $3.644 trillion
  - float deficit = 4.147E12 - 3.644E12;
  - int deficit = 4.147000000000 - 3.644000000000;

int can actually only store a number this big → 2147483647

The Difference is "Significant"

- What is the value of the integer i ?
  - $10 \div 4 = 2$
- What is the value of the integer x ?
  - $10.0 \div 4.0 = 2.5$

```
int i;
float x;
i = 10 / 4;
x = 10.0 / 4.0;
```

Can You Say Rounding Error?

- What happens when we have to convert integer values into float values or vice versa?
- Can lead to **a loss of accuracy**.
- *int* to *float*
- *f* is 100.0, because type *int* (100) is automatically converted into a *float* (100.0).
- **computers store *floats* and *ints* differently in their memory**.

```
int i;
float f;
i = 100;
f = i;
```

See GITHUB code_1_4_11

Can you say Rounding Error?

- Result in a loss of accuracy
- The value of the variable i will be 100
- The .25 gets thrown away when you recast
- Converting a **float** into an **int** is not always feasible

```
int i;
float f;
f = 100.25;
i = f;
```

See GITHUB code_1_4_10

Can You Say More Rounding Error?

- Again, converting a **float** into an **int** is not always feasible
- Integer variables (unlike floats) have a limited capacity
  - They cannot contain arbitrarily large (or arbitrarily small) numbers
- Four bytes (i.e. 32 bits) to store int values
  - numbers from the range of -2147483648 to 2147483647

```
int i;
float f;
f = 1E10;
i = f;
```

See GITHUB code_1_4_11

Lab 1.4.1 (Parentheses)

See GITHUB lab_1_4_1 Parentheses

Fix Github number for the lab

Operators = + - * /

An Assignment Operator

```
int myVariable;
int hitCouonter, lifeForce;

myVariable = 9;
myVariable = 9 * hitCounter;
myVariable = lifeForce - hitCounter;
```

Multiplication

- A asterisk * is **a multiplication operator**
- What is the value of k?
- What is the value of z?

```
int i,j,k;
float x,y,z;
i = 10;
j = 12;
k = i * j;
x = 1.25;
y = 0.5;
z = x * y;
```

Division

- A slash / is **a divisional operator**
- The value in front of the slash is a **dividend**
- The value behind the slash is a **divisor**.

```
int i,j,k;
float x,y,z;
i = 10;
j = 5;
k = i / j;
x = 1.0;
y = 2.0;
z = x / y;
```

Divide by 0?

- Dividing by zero is strictly forbidden
- Dividing by zero will generate
  - a compilation error, runtime error, or some message at runtime
- Run time error: When executing this code, the result of the operation is not a number
  - Special featured value named inf (as in infinitive)
  - This kind of illegal operation is a so-called exception

```
float x;
x = 1.0 / 0.0;
```

```
float x,y;


x = 0.0;
y = 1.0 / x;
```

ADDITION

- The **addition operator** is the **+** (plus) sign
- What is the value of k?
- What is the value of z?

```
int i,j,k;
float x,y,z;

i = 100; j = 2;
k = i + j;
x = 1.0; y = 0.02;
z = x + y;
```

SUBTRACTION

- The **subtraction operator** is obviously the – (minus) sign
- Note that this operator also has another meaning – it can change the sign of a number.

```
int i,j,k;
float x,y,z;

i = 100; j = 200;
k = i - j;
x = 1.0; y = 1.0;
z = x - y;
```

Unary Minus

- In “subtracting” applications, the minus operator expects two arguments:
  - The left (a minuend in arithmetical terms)
  - The right (a subtrahend).
- The subtraction operator is considered to be one of the binary operators, just like the addition, multiplication and division operators
- The minus operator can also be a **unary operator**, as it expects only one argument - the right one

```
int i,j;
 i = -100;
 j = -i;
```

Remainder

- The **remainder operator** is not traditional arithmetic operators.
- Its graphical representation in the C++ language is the **%** (percent) character
  - It's a binary operator (it performs **the modulo operation**) and both arguments cannot be floats
- You cannot compute the remainder with the right argument equal to zero
  - Division by 0 invokes undefined behavior, the modulo operation, is undefined, too.

```
int i,j,k;
i = 13;
j = 5;
k = i % j;
```

Priorities & Binding

- operators of larger (higher) priority perform their operations before the operators with lower priority
- Most operators in the C++ language have the **left-sided binding**
  - which means that the calculation of this sample expression is conducted from left to right
  - 3 will be added to 2, and 5 will be added to the result.

```
x = 2 + 3 * 5;

y = 2 + 3 + 5;
```

List of priorities

- operators in order **from the highest to the lowest priority**
- Unary?
- Binary?

+ -	unary
* / %	
+ -	binary

LIST OF PRIORITIES QUIZ

```
x = 2 * 3 % 5;
```

Parentheses

- Parentheses change the natural order of calculation
- subexpressions in parentheses are always calculated first

```
int i,j,k,l;
i = 100;
j = 25;
k = 13;
l = (5 * ((j % k) + i) / (2 * k)) / 2;
```

Increment a Variable by One

int SheepCounter;

SheepCounter = 0;

SheepCounter = SheepCounter + 1;

++ (plus plus) incrementor operator

SheepCounter++;

-- (minus minus) decrementor operator

SheepCounter = SheepCounter - 1;

SheepCounter--;

Ahhhh I ran out of time…

LAB 1.4.1 (FLOAT)

See GITHUB lab_1_4_1_float

Fix Github number for the lab

Reference Slides

Source Material

- Google search for C++ Sides
  - http://stroustrup.com/Programming/lecture-slides.html

Variable Names - Cont

- Which do you like?
  - i
  - t10
  - Exchange_Rate ← exchangeRate
  - counter
  - DaysToTheEndOfTheWorld ← daysToTheEndOfTheWorld
  - TheNameOfAVariableWhichIsSoLongThatYouWillNotBeAbleToWriteItWithout Mistakes
  - _
- What's Wrong?
  - 10t (does not begin with a letter)
  - Adiós_Señora (contains illegal characters)
  - Exchange Rate (contains a space)