About

Name : Kamoga Henry

Position : Full Stack Software Developer

Contacts : 0701 243 139 / 0771 977 854

Website: https://mogahenze.com

Github : https://github.com/HENRY-2016

Languages: Python, C, C++, Javascript, PHP, Web Technologies and more

Worked On Projects:

Mobile Application

dr kg beddings App
https://play.google.com/store/apps/details?id=com.mogahenze.drkgbeddings1

biva organic
https://play.google.com/store/apps/details?id=com.mogahenze.bivaOrganic

WHAT WILL BE COVERED

Link :: https://www.tutorialspoint.com/cplusplus/cpp_operators.htm

1. OVERVIEW
Object-Oriented Programming
Standard Libraries
The ANSI Standard
Learning C++

2. ENVIRONMENT SETUP
Windows

3. BASIC SYNTAX

C++ Program StructureCompile & Execute C++ Program
Semicolons & Blocks in C++
C++ Keywords

4. COMMENTS IN C++

5. DATA TYPES
Primitive Built-in Types
typedef Declarations
Enumerated Types

6. VARIABLE TYPES
Variable Definition in C++
Variable Declaration in C++
Lvalues and Rvalues

7. VARIABLE SCOPE
Local Variables
Global Variables
Initializing Local and Global Variables

8. CONSTANTS/LITERALS
Integer Literals
Floating-point Literals
Boolean Literals
Character Literals
String Literals
Defining Constants

9. MODIFIER TYPES
Type Qualifiers in C++

10. STORAGE CLASSES
The auto Storage Class
The register Storage Class
The static Storage Class
The extern Storage Class
The mutable Storage Class

11. OPERATORS

Arithmetic Operators
Relational Operators
Logical Operators
Bitwise Operators
Assignment Operators
Misc Operators
Operators Precedence in C++

12. LOOP TYPES
While Loop
Loop Control Statements
The Infinite Loop
13. DECISION-MAKING STATEMENTS
If Statement
if...else Statement
if...else if...else Statement
Switch Statement
Nested if Statement
The ? : Operator

14. FUNCTIONS
Defining a Function
Function Declarations
Calling a Function
Function Arguments

15. NUMBERS
Defining Numbers in C++
Math Operations in C++
Random Numbers in C++

16. ARRAYS
Declaring Arrays
Initializing Arrays
Accessing Array Elements
Arrays in C++
Passing Arrays to Functions

17. STRINGS
The C-Style Character String

The String Class in C++

18. POINTERS
What are Pointers?
Using Pointers in C++
Pointers in C++
Null Pointers
Pointer Arithmetic
Pointers vs Arrays
Array of Pointers
Pointer to a Pointer
Passing Pointers to Functions
Return Pointer from Functions

19. REFERENCES
References vs Pointers
Creating References in C++
References as Parameters
Reference as Return Value
20. DATE AND TIME
Current Date and Time
Format Time using struct tm

20. DATE AND TIME
Current Date and Time
Format Time using struct tm

21. BASIC INPUT/OUTPUT
I/O Library Header Files
The Standard Output Stream (cout)
The Standard Input Stream (cin)
The Standard Error Stream (cerr)
The Standard Log Stream (clog)

22. DATA STRUCTURES
Defining a Structure
Accessing Structure Members
Structures as Function Arguments
Pointers to Structures

23. CLASSES AND OBJECTS

C++ Class Definitions
Define C++ Objects
Classes & Objects in Detail
Class Access Modifiers
The public Members
The private Members
The protected Members
Constructor & Destructor
Friend Functions
Inline Functions
this Pointer
Pointer to C++ Classes
Static Members of a Class
Static Function Members

24. INHERITANCE

Base & Derived Classes
Access Control and Inheritance
Type of Inheritance
Multiple Inheritance

25. OVERLOADING (OPERATOR & FUNCTION)

Function Overloading in C++
Overloadable/Non-overloadable Operators
Operator Overloading Examples
Unary Operators Overloading
Increment (++) and Decrement (- -) Operators
Binary Operators Overloading
Relational Operators Overloading
Input/Output Operators Overloading
++ and - - Operators Overloading
Assignment Operators Overloading
Function Call () Operator Overloading
Subscripting [ ] Operator Overloading
Class Member Access Operator - > Overloading

26. POLYMORPHISM

Virtual Function

27. DATA ABSTRACTION
Access Labels Enforce Abstraction
28. DATA ENCAPSULATION
29. INTERFACES
30. FILES AND STREAMS
Opening a File
Closing a File
Writing to a File
File Position Pointers

31. EXCEPTION HANDLING
Throwing Exceptions
Catching Exceptions
C++ Standard Exceptions
Define New Exceptions

32. DYNAMIC MEMORY
The new and delete Operators
Dynamic Memory Allocation for Arrays
Dynamic Memory Allocation for Objects
33. NAMESPACES
Defining a Namespace
The using directive
Discontiguous Namespaces
Nested Namespaces
34. TEMPLATES
Function Template
Class Template
35. PREPROCESSOR
The #define Preprocessor
Function-Like Macros
Conditional Compilation
The # and # # Operators
Predefined C++ Macros
36. SIGNAL HANDLING
The signal() Function
The raise() Function

37. MULTITHREADING
Creating Threads

Terminating Threads
Passing Arguments to Threads
Joining and Detaching Threads

38. WEB PROGRAMMING

What is CGI?
Web Browsing
CGI Architecture Diagram
Web Server Configuration
HTTP Header
CGI Environment Variables
C++ CGI Library
GET and POST Methods
Using Cookies in CGI

39. STL TUTORIAL

40. STANDARD LIBRARY

The Standard Function Library .
The Object Oriented Class Library

OOP Fundamentals

Class :: E.g class {Fruits } Objects { Apple, Banana }
Objects ::
Method :: Methods are functions that belongs to the class
Two definition mthds
Inside class definition
Outside class definition

Constructor || Destructor
Is a special method that is automatically called when an object of a class is created

Access Specifiers
3 access specifiers
public - members are accessible from outside the class
private - members cannot be accessed (or viewed) from outside the class
protected - members cannot be accessed from outside the class, however, they can be accessed in inherited classes

Exceptions

```
try {
  // Block of code to try
  throw exception; // Throw an exception when a problem arise
}
catch () {
  // Block of code to handle errors
}
```

Pillars of OOPS

Inheritance

Inheritance is the procedure in which one class inherits the attributes and methods of another class.

Encapsulation

The word, “encapsulate,” means to enclose something

we encapsulate by binding the data and functions

Polymorphism

It describes the concept that you can access objects of different types through the same interface

Abstraction

Hiding the internal details or implementations of a function and showing its functionalities only