Name: BIT STUDENTS

***MOUNTAINS OF THE MOON UNIVERSITY***

***Faculty of Science, Technology and Innovation***

***DEPARTMENT OF COMPUTER SCIENCE***

Course: BIT

Course unit: STRUCTURED PROGRAMMING

Lecturer: Mr. ANDREW

***GROUP ASSIGNMENT***

***Assignment***

**Bachelor of Information Technology**

1. Explain the any tools used by programmers to develop C++ program.
2. What are the minimum requirement of installing C++Program
3. Demonstrate the procedure/steps of compiling C++Program
4. Write down the procedures of writing and running a C++
5. With aid of illustration describe the Basic structure of C++

Discuss the basic elements of C++ program

***ANSWERS***

**1** Software Development Kits

There are some free C/C++ compilers available, including:

[GNU Compiler Collection](http://gcc.gnu.org/)  
[Minimalist GNU for Windows (MinGW)](http://www.mingw.org/)

2 Graphics/GUI APIs and Libraries

There are some a variety of different C/C++ APIs/libraries available for 2D graphics and/or graphical user interfaces.

3 General APIs and Libraries

There are quite a few C/C++ libraries in the public domain, but you have to be careful. Some sources include:

[Planet Source Code](http://www.planet-source-code.com/)  
[The Free Country](http://www.thefreecountry.com/developercity/freelib.html)

4 Integrated Development Environments

There are a number of commercial Integrated Development Environments (IDEs) that support C/C++. There are also some good free IDEs available, including:

[Bloodshed Dev-C++](http://www.bloodshed.net/devcpp.html)  
[Code::Blocks](http://www.codeblocks.org/)  
[Visual-MinGW](http://visual-mingw.sourceforge.net/)

5 Documentation Tools

There are a number of documentation tools available for C/C++. Some are similar to the javadoc tool that comes with Java, including:

[CppDoc](http://www.cppdoc.com/)  
[Doxygen](http://www.stack.nl/~dimitri/doxygen/)

6 Unit Testing

Several unit testing tools exist for C/C++, including:

[Auto Unit](http://autounit.tigris.org/)  
[Check](https://libcheck.github.io/check/)  
[Google Test](https://github.com/google/googletest)  
[minunit](https://github.com/siu/minunit)

***No2***

## ****System Requirements****

### ****Operating Systems****

Aspose.Page for C++ is a native library. It supports 32 as well as 64-bit operating systems, but not limited to:

* Microsoft Windows desktop (XP, Vista, 7, 8, 10) and server operating systems (2003, 2008, 2012), etc.

### ****Development Environments****

You can use Aspose.Page for C++ to develop applications in any development environment that supports C++, but the following environments are explicitly supported:

* Microsoft Visual C++ version 2017 or higher

***No3***

## The different kinds of files

Compiling C++ programs requires you to work with four kinds of files:

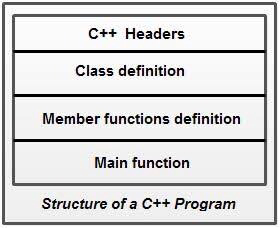
1. Regular **source code** files. These files contain function definitions, and have names which end in ".cc" by convention (although sometimes you will see source code filenames which end in ".cpp" or ".C").
2. **Header** files. These files contain class declarations, function declarations (also known as function prototypes) and various preprocessor statements (see below). They are used to allow source code files to access externally-defined classes and functions. Header files end in ".hh" or ".h" by convention.
3. **Object** files. These files are produced as the output of the compiler. They consist of function definitions in binary form, but they are not executable by themselves. Object files end in ".o" by convention, although on some operating systems (Windows, MS-DOS), they often end in ".obj".
4. **Binary executables**. These are produced as the output of a program called a "linker". The linker links together a number of object files to produce a binary file which can be directly executed. Binary executables have no special suffix on Unix operating systems, although they generally end in ".exe" on Windows.

There are other kinds of files as well, notably libraries (".a" files) and shared libraries (".so" files), but you won't normally need to deal with them directly.

***No5***

# **Structure of a C++ Program**

Programs refer to a sequence of instructions or statements. These statements are what form the structure of a C++ program. Moreover, the C++ program structure divides into several sections which are namely headers, class definition, member functions definitions and main function. So, this article will let us learn about struct C++ in detail.



***No 6***

Function: collection of statements; when executed, accomplishes something o May be predefined or standard

Syntax: rules that specify which statements (instructions) are legal

o Programming language: a set of rules, symbols, and special words

Semantic rule: meaning of the instruction Comments are for the reader, not the compiler

o Two types:

􏰁 Single line

//This is a C++ program. It prints the sentence: // Welcome to C++ Programming.

􏰁 Multiple line

/\*

You can include comments that can occupy several lines.

\*/

Reserved words, keywords, or word symbols

o Include:

􏰁 int

􏰁 float

􏰁 double 􏰁 char

􏰁 const 􏰁 void

􏰁 return

Identifiers and variables:

o Consist of letters, digits, and the underscore character ( \_ ) o Must begin with a letter or underscore

o C++ is case sensitive

􏰁

***REF***

https://w3.cs.jmu.edu/bernstdh/web/common/tools/c.php

* The man page for g++. Type:  **man g++ | more**   at the unix prompt. This is actually the same as the man page for **gcc**. Just look for the material specific to **g++**.

The GNU Info documentation on gcc. This also includes full documentation of g++. **Warning!** This is far more information than most people could possibly absorb in the average millenium.

https://www.instructables.com/How-to-Write-a-Simple-C-Program/?amp\_page=true

https://www.toppr.com/guides/computer-science/introduction-to-c/getting-started-with-c/structure-of-a-c-program/