KATHMANDU UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

An
Assignment on
Object Oriented Programming {COMP116}
Assignment No: 1

Submitted by:
Ashraya Kadel
UNG CE 1/2
Roll No: 25

Submitted to:
Dr. Rajani Chulyadyo
Assistant Professor
Department of Computer science and engineering

Submission date: 25 / 08 / 2023

<Q.1> Explain how C++ programs work.

Ans:

C++ programs works in the following way.

(i): The source code is written using C++ programming language

(ii): Then, the preprocessor replaces the files starting with "#" with the declaration of function included in the file. Source code is expanded.

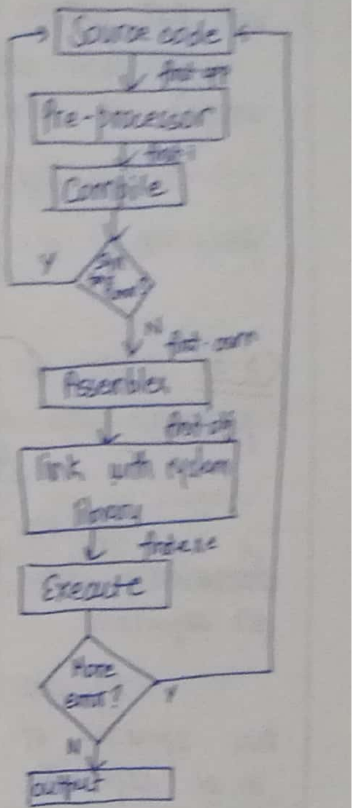
iii) The compiler then compile's the program.

iv) The assembler converts the compiler code into object code with file '.obj'.

v) The object code is then linked with system library and then it creates and executable file '.exe'

vi) The file localized to main memory is in single file and program is executed.

vii) The execution of the program starts from main.



<Q.2> A C++ program that compiles in one compiler may not compile in another compiler. why?

Ans

Compiler translates programs written in high level language to machine code at once.

A C++ program that compiles in one compiler ~~doesn't~~ may not compile in another compile.

This is because if our ~~system~~ code contains references to system files, then it will not compile on a different system.

Eg: Code containing windows dlls doesn't compile in linux system.

But if your code doesn't reference system files, then it will compile on any system.

<Q.3> What do you understand by operator precedence and associativity?

Ans:

Operator precedence is the property determining how different types are executed in an expression.

Operator associativity is the property determining how operators of the same precedence are executed in an expression.

Eg: 5 * (3+1) / ~~8~~2 - 8 % 2

=> 5 * 4/2 - 8%2

=> 5 * 2 - 8%2

=> 5*2 - 0 => 10

<Q.4> What are the differences between pointers and references?

Ans

The differences between pointers and references are as follows:

References	Pointers
Variable cannot be reassigned in reference.	- Variables can be reassigned in pointers
Shares same address as original variable.	- Pointers have their own memory address.
It refers to another variable.	- It stores address of another variable.
It doesn't have NULL value	- NULL value can be assigned.
This variable is referenced by method pass by value.	- The variable does work by method pass by reference.

<Q.5> What are the differences betⁿ pass by value and pass by reference?

Ans:

The differences between pass by value and pass by reference are as follows:

Pass by value	Pass by reference.
It is mechanism of copying the function parameter value to another variable.	- It is mechanism of passing the actual parameters to the function.
Changes made inside function is not reflected upon the original value.	- Changes made inside the function are reflected in the original value.

P.T.O.

Makes copy of actual parameter and execution takes more time.	- Address of the actual parameter is passed and execution ~~takes~~ is ~~more~~ faster.
It requires more memory.	- It requires less memory
Function gets copy of actual content.	- Function accesses original variable content.

<Q·6> Explain the purpose of namespaces.

Ans:

A namespace is a declarative region that provides a scope to the identifiers inside it.

The purpose of namespaces are as follows:

i) It is used to organize code into logical groups.

ii) It is used to prevent name collisions that occurs especially when your codebase includes multiple libraries.

<Q·8> Differentiate bet^{n} pointer & variable reference

Ans:

Pointer variable	Reference variable.
It is variable that ~~points~~ stores address of another variable.	- Reference variable is an alias for another variable
A indirection operator (*) is used to dereference a pointer.	- Reference variable doesn't need dereferencing operator.
The values can be reassigned.	- The values cannot be reassigned.
NULL value can be directly assigned.	- NULL value cannot be directly assigned.

<Q·7>: Compare inline function and normal function on the basis of memory usage, execution time and also explain trade-off between them.

Ans:

Inline function	Normal Function.
It expands the code inline when invoked.	- It provides molecularity to the program.
It is used when small functions called very often	- It is used to improve reusability of code making it maintainable.
Requires 'inline' keyword.	- No keyword declaration.
Execution is generally faster.	- Execution is generally slower.
Compiler pastes code inline.	- compiler doesn't paste code inline.
Functions inside class are implicility inline.	- Functions outside classes are normally normal functions
Use of too many inline function increases the size of executable file.	- Use of normal functions doesn't affect the size of executable file.