C++ Programming

Trainer : Rohan Paramane

Email: rohan.paramane@sunbeaminfo.com

Object-oriented software development (OOSD)

- In the past, the problems faced by software development were relatively simple, from task analysis to programming, and then to the debugging of the program, if its not too big it can be done by one person or a group.
- With the rapid increase of software scale, software personnel faces the problem that is very complicated, and there are many factors that need to be considered.
- The errors generated and hidden errors may reach an astonishing degree, this is not something that can be solved in the programming stage.
- Need to standardize the entire software development process and clarify the software
- The tasks of each stage in the development process, while ensuring the correctness of the work of the previous stage, proceed to the next stage work.
- This is the problem that software engineering needs to study and solve. Object-oriented software development and engineering include the following parts:

1.Object oriented analysis (OOA)

- The first step of Object-oriented software development is Object-Oriented Analysis (OOA)
- In the system analysis stage of software engineering, system analysts must integrate with users to make precise Accurate analysis and clear description, summarize what the system should do (not how) from a macro perspective.
- Face right the analysis of the image should be based on object-oriented concepts and methods.
- In the analysis of the task, from the objective existence of things and the relationship between the related objects (including the attributes and behaviors of the objects) and the relationship between the objects are summarized.
- The Objects with the same attributes and behaviors are represented by a class.
- Establish a need to reflect the real work situation model. The model formed at this stage is relatively rough (rather than fine).

2.Object oriented design (OOD)

- The second step of Object-oriented software development is Object-Oriented Design (OOD).
- According to the demand model formed in the object-oriented analysis stage, each part is specifically designed.
- The design of the line class may contain multiple levels (using inheritance).
- Then these classes put forward the ideas and methods of program design, including the design of algorithms.
- In the design stage no specific plan is involved, but a more general description tool (such as pseudo code or flowchart)is used to describe.

3.Object-oriented programming (OOP)

- The third step of Object-oriented software development is Object-oriented Programming (OOP).
- According to the results of object-oriented design, to write it into a program in a computer language, it is obvious that object-oriented Computer language (e.g. C++) needs to be used.
- Otherwise the requirements of object-oriented design cannot be achieved.

OOPS(Object Oriented Programming Language)

- OOPS is not a syntax.
- It is a process / programming methodology which is used to solve real world problems.
- It is a programming methodology to organize complex program in to simple program in terms of classes and object such methodology is called oops.
- It is a programming methodology to organized complex program into simple program by using concept of abstraction , encapsulation , polymorphism and inheritance.
- Languages which support abstraction , encapsulation polymorphism and inheritance are called oop language.

Major pillars of oops

- **Abstraction**
  - getting only essential things and hiding unnecessary details is called as abstraction.
  - Abstraction always describe outer behavior of object.
  - In console application when we give call to function in to the main function , it represents the abstraction
- **Encapsulation**
  - binding of data and code together is called as encapsulation.
  - Implementation of abstraction is called encapsulation.
  - Encapsulation always describe inner behavior of object
  - Function call is abstraction
  - Function definition is encapsulation.
  - Information hiding
    - Data : unprocessed raw material is called as data.
    - Process data is called as information.
    - Hiding information from user is called information hiding.
    - In c++ we used access Specifier to provide information hiding.
- **Modularity**
  - Dividing programs into small modules for the purpose of simplicity is called modularity.
- **Hierarchy (Inheritance [is-a] , Composition [has-a] , Aggregation[has-a], Dependancy)**
  - Hierarchy is ranking or ordering of abstractions.
  - Main purpose of hierarchy is to achieve re-usability.

Minor pillars of oops

- **Polymorphism (Typing)**
  - One interface having multiple forms is called as polymorphism.
  - Polymorphism have two types
    1. **Compile time polymorphism**

       when the call to the function resolved at compile time it is called as compile time polymorphism. And it is achieved by using function overloading and operator overloading
    2. **Runtime polymorphism.**

       when the call to the function resolved at run time it is called as run time polymorphism. And it is achieved by using function overriding.
  - Compile time / Static polymorphism / Static binding / Early binding / Weak typing / False Polymorphism
  - Run time / Dynamic polymorphism / Dynamic binding / Late binding / Strong typing / True polymorphism
- Concurrency
  - The concurrency problem arises when multiple threads simultaneously access same object.
  - You need to take care of object synchronization when concurrency is introduced in the system.
- Persistence
  - It is property by which object maintains its state across time and space.
  - It talks about concept of serialization and also about transferring object across network.

Limitations of C Programming

- C is said to be process oriented, structured programming language.
- When program becomes complex, understating and maintaining such programs is very difficult.
- Language don’t provide security for data.
- Using functions we can achieve code reusability, but reusability is limited. The programs are not extendible.

Classification of high level Languages

- Procedure Oriented Programming language(POP)
  - ALGOL, FORTRAN, PASCAL, BASIC, C etc.
  - "FORTRAN" is considered as first high level POP language.
  - All POP languages follows "TOP Down" approach

- Object oriented programming languages(OOP)
  - Simula, Smalltalk, C++, Java, C#, Python, Go
  - "Simula" is considered as first high level OOP language.
  - more than 2000 lang. are OO.
  - All OOP languages follows "Bottom UP" approach

Few Real Time Applications of C++

- Games
- GUI Based Application (Adobe)
- Database Software (MySQL Server)
- OS (Apple OS)
- Browser(Mozilla)
- Google Applications(Google File System and Chrome browser)
- Banking Applications
- Compilers
- Embedded Systems(smart watches, MP3 players, GPS systems)

Characteristics of Language

1. It has own syntax
2. It has its own rule(semantics)
3. It contain tokens:
   - Identifier
   - Keyword
   - Constant/literal
   - Operator
   - Seperator / punctuators
4. It contains built in features.
5. We use language to develop application(CUI, GUI, Library)

History of C++

- Inventor of C++ is Bjarne Stroustrup.
- C++ is derived from C and simula.
- Its initial name was "C With Classes".
- At is developed in "AT&T Bell Lab" in 1979.
- It is developed on Unix Operating System.
- In 1983 ANSI renamed "C With Classes" to C++.
- C++ is objet oriented programming language

C++ Specifications

- In 1985, the first edition of The C++ Programming Language was released.
- In 1989, C++ 2.0 was released. New features in 2.0 included multiple inheritance, abstract classes, static member functions, const member functions, and protected members.Later feature additions included templates, exceptions, namespaces, new casts, and a Boolean type.
- In 1998, C++98 was released, standardizing the language, and a minor update (C++03) was released in 2003.
- After C++98, C++ evolved relatively slowly until, in 2011, the C++11 standard was released, adding numerous new features,enlarging the standard library further, and providing more facilities to C++ programmers.
- A minor C++14 update was released in December 2014.
- A major revision where various new additions were introduced in C++17.

Main Function

- main should be entry point function of C/C++
- Calling/invoking main function is responsibility of operating system.
- Hence it is also called as Callback function

Execution Flow

- Execution of a C/C++ program involves four stages using different compiling/execution tools
  - Preprocessor
  - Compiler
  - Linker
  - Loader

These tools make the program running.

1) Preprocessor

This is the first stage of any C/C++ program execution process, in this stage Preprocessor processes the program before compilation. Preprocessor include header files, expand the Macros.

2) Complier

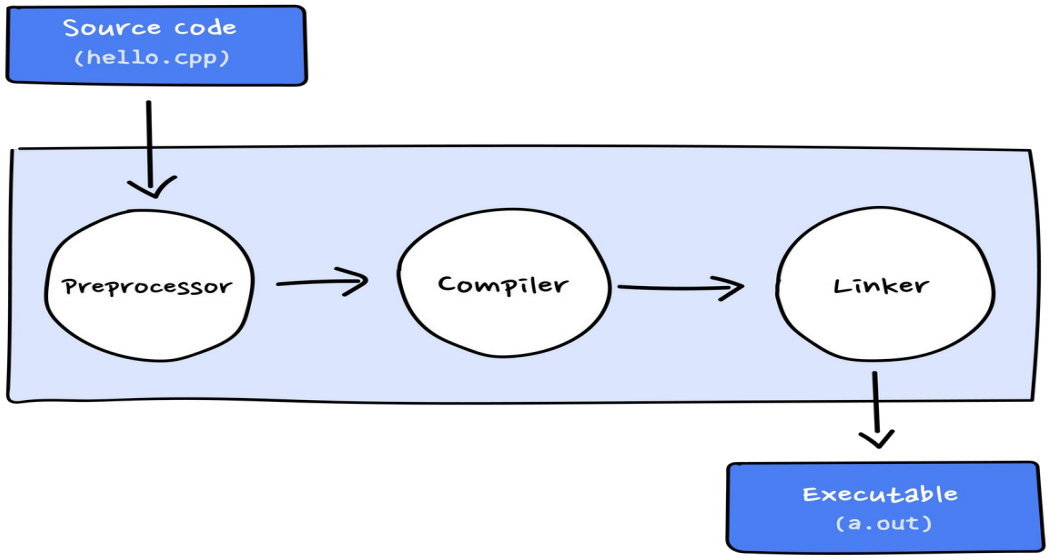
This is the second stage of any C/C++ program execution process, in this stage generated output file after preprocessing (with source code) will be passed to the compiler for compilation. Complier will compile the program, checks the errors and generates the object file (this object file contains assembly code).

3) Linker

This is the third stage of any C/C++ program execution process, in this stage Linker links the more than one object files or libraries and generates the executable file.

4) Loader

This is the fourth or final stage of any C/C++ program execution process, in this stage Loader loads the executable file into the main/primary memory. And program run.



Data Types in C++

- It describes 3 things about variable / object

1. Memory : How much memory is required to store the data.
2. Nature : Which type of data memory can store
3. Operation : Which operations are allowed to perform on data stored inside memory.

- Fundamental Data Types
  - (void, int,char,float,double)
- Derived Data Types
  - (Array, Function, Pointer, Union ,Structure)

Two more additional data types that c++ supports are

1. ***bool* :-** it can take *true* or *false* value. It takes one byte in memory.
2. **wchar_t :-** it can store 16 bit character. It takes 2 bytes in memory.

Bool and wchar_t

- **e.g: bool val=true;**
- **wchar_t**: Wide Character. This should be avoided because its size is implementation defined and not reliable.
- Wide char is similar to char data type, except that wide char take up twice the space and can take on much larger values as a result. char can take 256 values which corresponds to entries in the ASCII table. On the other hand, wide char can take on 65536 values which corresponds to UNICODE values which is a recent international standard which allows for the encoding of characters for virtually all languages and commonly used symbols.

- The type for character constants is char, the type for wide character is wchar_t.
- This data type occupies 2 or 4 bytes depending on the compiler being used.
- Mostly the wchar_t datatype is used when international languages like Japanese are used.
- This data type occupies 2 or 4 bytes depending on the compiler being used.
- L is the prefix for wide character literals and wide-character string literals which tells the compiler that that the char or string is of type wide-char.
- w is prefixed in operations like scanning (**wcin)** or printing (**wcout)** while operating wide-char type.

Type modifiers

- C++ allows the char, int, and double data types to have modifiers preceding them. A modifier is used to alter the meaning of the base type so that it more precisely fits the needs of various situations.
- The data type modifiers are listed here −
  - signed
  - unsigned
  - long
  - short
- The modifiers signed, unsigned, long, and short can be applied to integer base types. In addition, signed and unsigned can be applied to char, and long can be applied to double.
- The modifiers signed and unsigned can also be used as prefix to long or short modifiers. For example, unsigned long int.

- **Type qualifiers**
  - The type qualifiers provide additional information about the variables they precede.
  - const and volatile are two qualifiers

Structure

- Structure is a collection of similar or dissimilar data. It is used to bind logically related data into a single unit.
- This data can be modified b any function to which the structure is passed.
- Thus there is no security provided for the data within a structure.
- This concept is modified by C++ to bind data as well as functions.

Access Specifier

- By default all members in structure are accessible everywhere in the program by dot(.) or arrow(□) operators.
- But such access can be restricted by applying access specifiers
  - private: Accessible only within the struct
  - public: Accessible within & outside struct

Structure in C & C++

struct in c	struct in c ++
we can include only variables into the structure.	we can include the variables as well as the functions in structure.
We need to pass a structure variable by value or by address to the functions.	We don't pass the structure variable to the functions to accept it / display it.The functions inside the struct are called with the variable and DOT operator.
By default all the variables of structure are accessible outside the structure. (using structure variable name)	By default all the members are accessible outside the structure, but we can restrict their access by applying the keywords private /public/ protected.
struct Time t1;	struct Time t1;
AcceptTime(struct Time &t1);	t1.AcceptTime(); //function call

Structure in C & C++

```
struct time {
   int hr, min, sec;
};
void display( struct time *p) {
   printf("%d:%d:%d", p->hr,
   p->min, p->sec);
}
struct time t;
display(&t);
```

```
struct time {
   int hr, min, sec;
void display(){
   printf("%d:%d:%d",
this->hr, this->min, this->sec);
}
};
time t;
t.display();
```

OOP and POP

OOP (Object Oriented Programming)	**POP** (Procedural Oriented Programming)
Emphasis on data of the program	Emphasis on steps or algorithm
OOP follows bottom up approach.	OOP follows top down approach.
A program is divided to objects and their interactions. Programs are divide into small data units i.e. classes	A program is divided into funtions and they interacts. Programs are divided into small code units i.e. functions
Objects communicates with each other by passing messeges.	Functions communicate with each other by passing parameters.
Inheritance is supported.	Inheritance is not supported.
Access control is supported via access modifiers. (private/ public/ protected)	No access modifiers are supported.
Encapsulation is used to hide data.	No data hiding present. Data is globally accessible.
C++, Java	C , Pascal
It overloads functions, constructors, and operators.	Neither it overload functions nor operators
Classes or function can become a friend of another class with the keyword "friend". Note: "**friend**" keyword is used only in c++	No concept of friend function.
Concept of virtual function appear during inheritance.	No concept of virtual classes .

Thank You