Programming Refresher

Programming Basics

Learning Objectives

By the end of this lesson, you will be able to:

- Classify different types of software and explain their distribution methods to understand the various ways software is distributed
- Differentiate between procedural, object-oriented, and functional programming paradigms to select the best one for your use case
- Construct a well-defined program structure to write syntactically correct code snippets
- Utilize various IDEs and programming tools to write and debug code effectively

Business Scenario

ABC is a supply-chain management company. The development team at ABC is creating software to manage their clients' product supply chains. This software can be either a web or a desktop application installed in all warehouses.

The company needs to decide which type of software will best meet its needs and how to use programming techniques to implement it.

Introduction to Software

What Is Software?

- Software is a set of instructions written for the computer to understand, interpret, and execute.
- Software programs are structured, organized, and formally written in a computer language. Hardware runs software programs.
- Without software support, hardware cannot increase its capabilities. The network and computer systems cannot function without software programs.

Software: Examples

Operating system

Internet browser

Adobe

Spreadsheet

Office

Hardware: Examples

Hardware is a physical electronic device used in or with a machine. Some examples of hardware in a computer are:

Mouse

Keyboard

Ethernet cords

Hard drive

Why Do We Need Software?

1. The software makes human tasks easier and provides a consistent way of executing instructions with minimal to zero error, unlike execution done by human beings.

2. Software, such as commercial off-the-shelf software, can be customized for any organization or user and designed for the mass market.

3. For instance, a calculator comes with software that makes regular calculations simpler and faster.

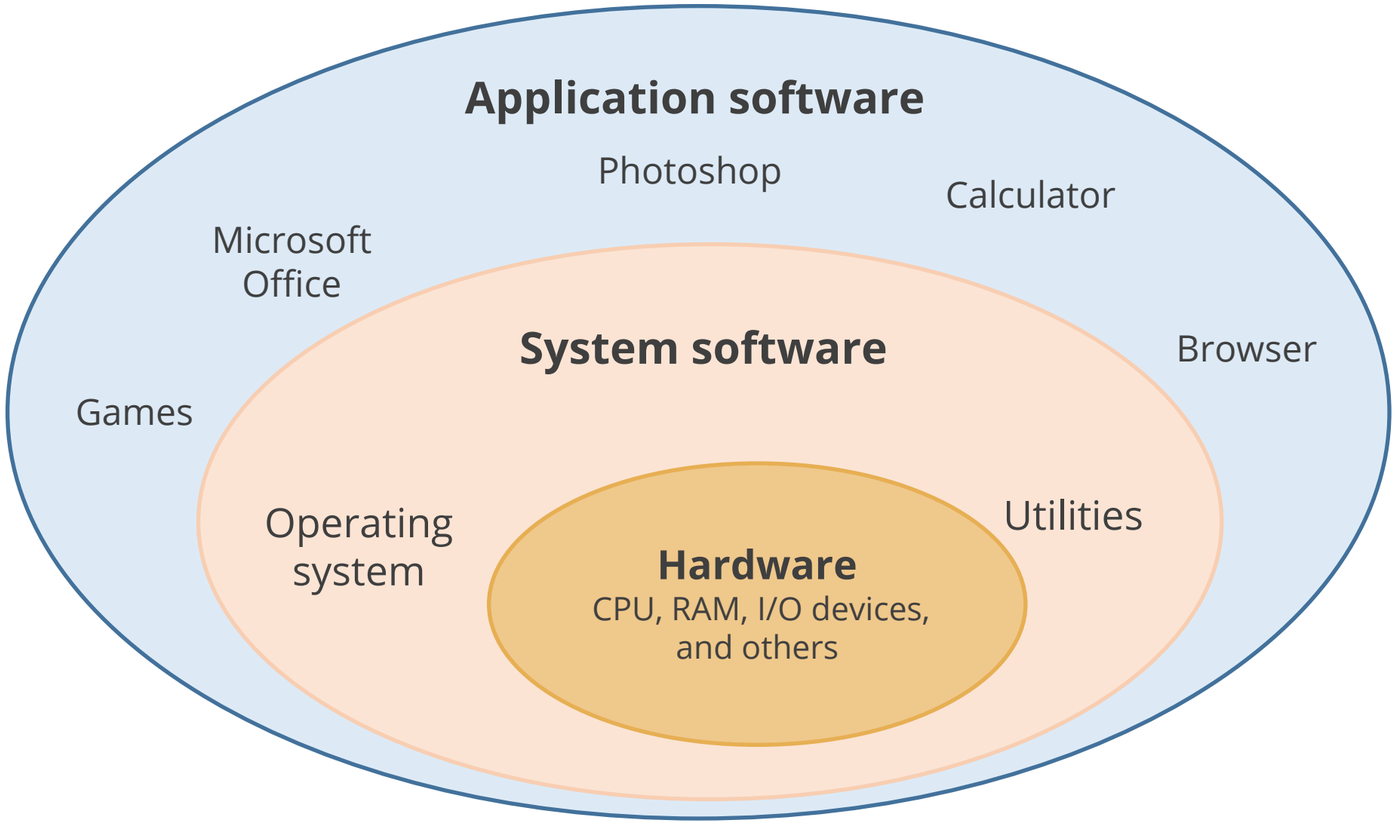
Why Do We Need Software?

- Software has evolved with the advancement of technology.
- In today’s era, software can create complex enterprise applications for various industries, such as transportation, medicine, defense, and telecommunications.

Categories of Software

Types of Software

Software can be broadly classified in two categories:



- System software
- Application software

System Software

It acts as an interface between hardware and application software. These programs run background processes. System software can be categorized based on its functions. The machine understands binary language, and system software helps convert human instructions into machine-understandable instructions.

Example

- Operating systems such as Windows, macOS, Linux, UNIX, Android, and iOS
- Utilities such as antivirus software, backup software, and file management tools

Application Software

It is a set of programs that perform a specific task on a system. These are customized to solve specific problems and provide functionalities.

Example

- Excel is used for operations with numbers.
- PowerPoint creates presentations.
- A browser is used for surfing the internet.
- A library management system provides functionalities related to the library.
- A tally is used for accounting.

Distribution of Software

Freeware

It refers to software that is freely available without any charge. It is typically released under a freeware license, which allows users to use, copy, distribute, and modify the software without any restrictions.

Shareware

It is typically released under a shareware license. It is a time-limited, free software that requires payment for continued use after the trial period.

Distribution of Software

Open-source software

It is often released under an open-source license and enables unrestricted usage, modification, and distribution. Its source code is generally accessible to the public, fostering inspection and improvement by a wide range of individuals.

Licensed software

It is protected by copyright and must be purchased from a vendor. It is governed by a proprietary license, granting users usage rights but not allowing them to modify or distribute the software.

Programming Models

Procedural Programming

- Procedural programming is a programming paradigm built around the idea that programs are sequences of instructions to be executed.
- A program is divided into multiple sections called modules, subroutines, or block structures. The aim is to improve the clarity, quality, and development time of a computer program.
- Instead of using a go-to statement like assembly language, procedural programming (structured programming) uses blocks, control structures, and loops.
- Example: Pascal and C

Object-Oriented Programming Language

OOP refers to languages that use objects in programming. It aims to implement real-world entities such as inheritance, information hiding, and polymorphism in programming.
Example: C++ and Java

Advantages of OOP

- Executes programs faster and easier
- Provides a clear structure for programs
- Offers more security than procedural languages
- Allows objects to move freely within member functions
- Helps create reusable applications with less code and shorter development time

OOP: Principles

OOP focus on the following principles:

- Inheritance
- Encapsulation
- Abstraction
- Polymorphism

Functional Programming

- Functional programming is a programming paradigm where programs are constructed by applying and composing functions.
- It emphasizes the use of pure functions and higher-order functions, treating computation as the evaluation of mathematical functions without changing state or data.
- Functional programming provides a structured and predictable approach to software development, offering numerous advantages in terms of reliability, maintainability, and concurrency, which are essential in modern computing environments.
- Example: Scala

Program Structure

Program Structure

1. A program consists of multiple components that interact with each other and have interrelationships. A well-structured program might be complex but not complicated.

2. In a well-structured program, the division into components follows a recognized principle, such as information hiding, and the interfaces between components are explicit and simple.

Program Structure

3

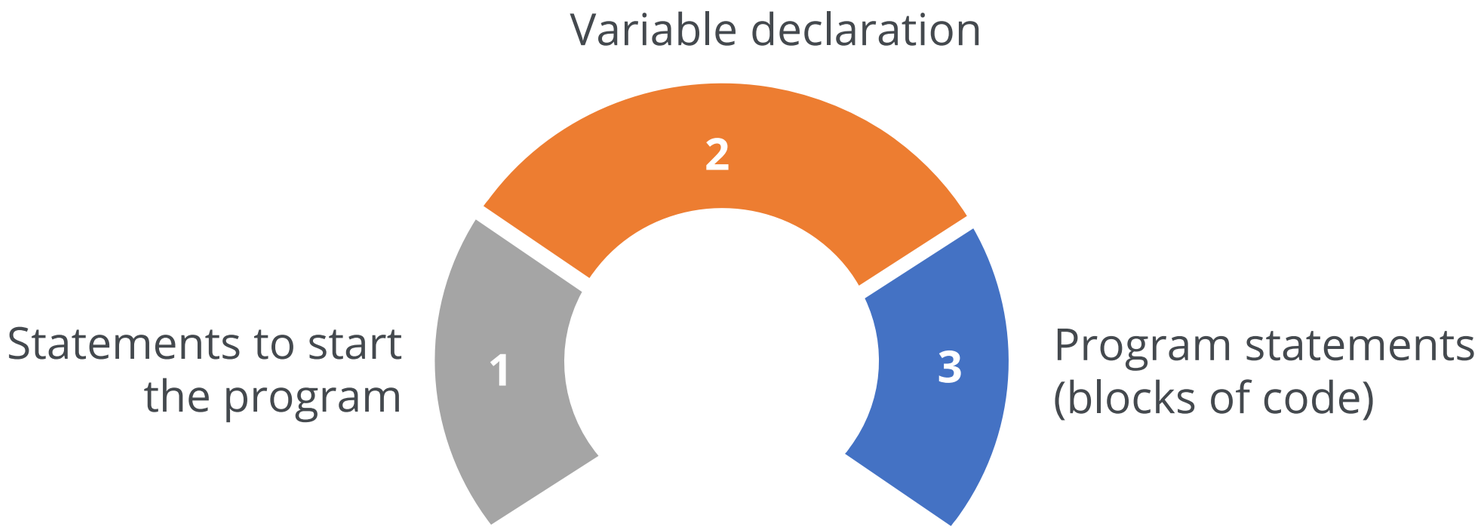
In contrast, a poorly structured program has interfaces that are implicit and difficult to understand, and the division into components is largely arbitrary (or non-existent).

4

A well-structured program employs appropriate data structures and program units with a single-entry point and a single exit point, while a poorly structured program employs arbitrary data structures and flow of control.

Structured Program

All structured programs share a similar overall pattern, as shown below:



Syntax in Different Languages

The programming languages listed below support the following syntax:

Language	Syntax example
C	#include<stdio.h> void main(){printf(“Hey Hi!!”);}
C++	#include<iostream> int main(){cout<<(“Hey Hi!!”; return 0;}
Pascal	program HelloWorld; begin writeln(“Hey Hi!!”); end
Oracle PL/SQL	CREATE OR REPLACEPROCEDURE helloworld AS BEGIN DBMS_OUTPUT.PUT_LINE(‘HELLO WORLD’); END;

Syntax in Different Languages

The programming languages listed below support the following syntax:

Language	**Syntax example**

Java	class helloworld{
	public static void main (String args []) {
	System.out.println(“Hello World!!”);}}

Perl	#!/usr/bin/env perl use strict; use warnings; print “Hello World”;
Basic	print “Hello World”

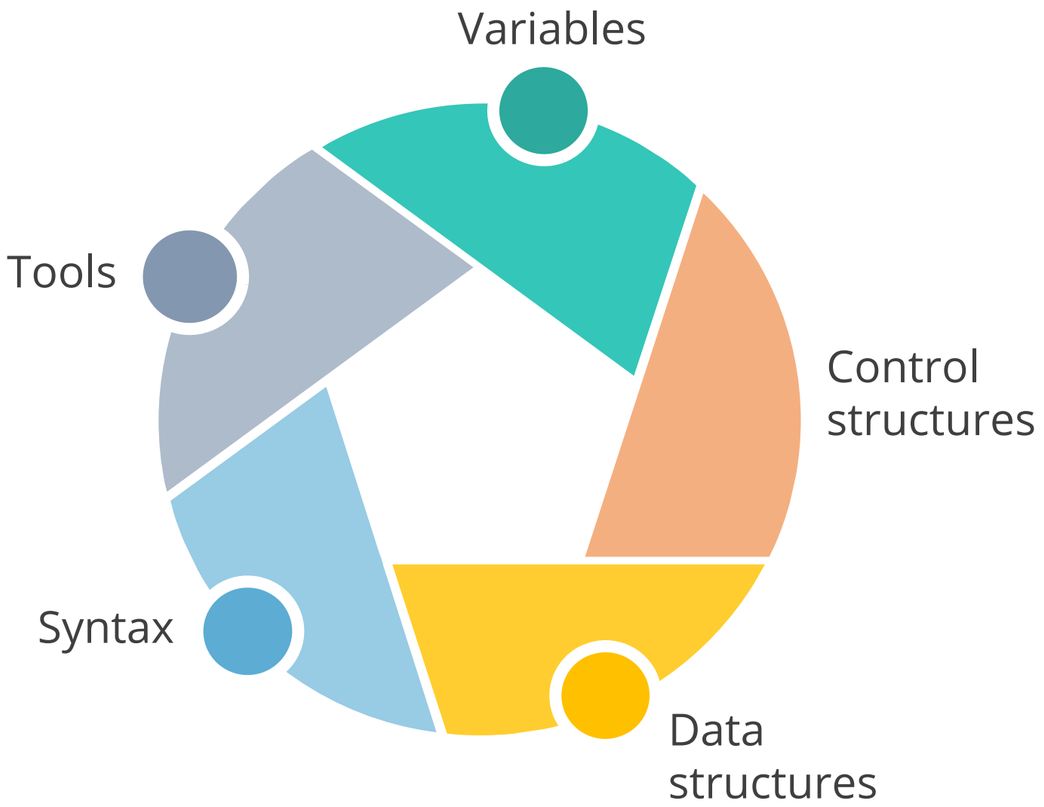
Types of Programming Language

Programming languages can be compiled or interpreted.

Compiler-based language	**Interpreter-based language**
A compiler takes the entire program in one go.	An interpreter takes a single line of code at a time.
The compiler generates intermediate machine code.	The interpreter never produces any intermediate machine code.
The compiler is best suited for the production environment.	An interpreter is best suited for a software development environment.
Programming languages, such as C, C++, C#, Scala, and Java are compiler-based languages.	Python, Ruby, PHP, and Perl are interpreter-based languages.

Programming Concepts

There are five basic concepts used in any programming language.



Variables

A variable is a storage location with an associated symbolic name that contains some known or unknown quantity or information, that is, a value.

Example

- int no;
- string name;
- double amount;

In some programming languages, it is important to specify the type of value that will be stored in the variable, known as the data type.

Control Structures

Control structures direct the program's flow depending on a decision, causing it to jump from one line to another or return a section of code. It can be done using an if-else construct.

Example

```
if (age >= 18)
“a person can vote “
else
“a person cannot vote “
```

It controls the flow of the program structure. When a program runs, a compiler reads and executes the user program line by line. This is called the top-to-bottom approach.

Data Structures

Data structures are a way of storing and organizing data on a computer that can be effectively used. Lists, dictionaries, and arrays are a few examples of data structures.

Example

To store a list of contacts for 100 people, instead of creating 100 variables, a data structure like a list or array can store these contacts.

Syntax

Syntax defines a set of rules that specify the combinations of symbols that are correctly structured statements or expressions in that language.

Example

```
String name = “Jack Ford”;
```

Symbols, characters, and words in a programming language have special meanings and rules to enable the compiler to understand the instructions.

Tools

Tools in programming are the editors that help the user code quickly and efficiently.

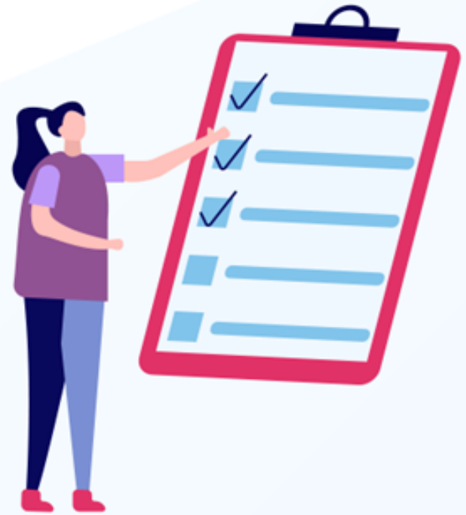
Example

- Eclipse
- Atom
- Jupyter
- Microsoft Visual Studio

There are various IDEs (Integrated development environments) available for coding based on technology and language.

Key Takeaways

- Software is a set of instructions given to the computer to process and provide the desired output.
- Software can be categorized based on its purpose into system or application software.
- Procedural, object-oriented, and functional programming are the three types of programming models.
- A program must be well-defined and structured to enhance, scale, and extend its capabilities for future requirements.



Knowledge Check

Knowledge Check

1

The language understood by a computer without translation is called _____

A. Command language

B. High-level language

C. Assembly language

D. Machine language

Knowledge Check 1

The language understood by a computer without translation is called _____

A. Command language

B. High-level language

C. Assembly language

D. Machine language

The correct answer is **D**

The computer understands machine language without translation.

Knowledge Check 2

Which of the following software defines an operating system and utility program?

A. System software

B. Application software

C. Device driver

D. Customized software

Knowledge Check 2

Which of the following software defines an operating system and utility program?

A. System software

B. Application software

C. Device driver

D. Customized software

The correct answer is **A**

Operating systems and utility programs are examples of system software.

Thank You!