# Programming Fundamentals

- computer only understand binary code.

01) what is computer ~~language?~~ program?
A computer program is a sequence or set of instructions in a programming language for a computer to execute.

02) what is source code? computer program in its human-readable ~~humdreadable~~ form.

03) what is Software?
set of programs that enables the hardware to perform a specific task.

$$\begin{tikzcd}
 & \text{Software} \arrow[dl] \arrow[dr] & \\
\text{System Software} & & \text{Application Software}
\end{tikzcd}$$

04) what is programming language?
is a computer language engineered to communicate instruction to machine.

- Machine language - is low level computer language that is designed to be directly understand by a computer.

Machine language

Pros: can run execute • very fast as the code.

Cons: • impossible for human use
- Programs are hard to maintain and debug
- No mathematical functions
- Memory ~~ne~~ management needs to be done manually.

(symbolic programming language)
Assembly language - machine dependent language. It use mnemonics (symblos and short words) instead of binary instructions. Assembler is translate, ~~a~~ assembly code into machine language.

Assembly languge :- It is difficult to read
- There are no symbolic~~al~~ names for memory location
- machine dependent

High level language (python, C++)

- High level languages are written in a form that is close to our human language, enabling programmers to just focus on the problem being solved.
- platform Independent language.

Source code / source program - A program written in the highlevel language.

- High level language → Machine language

↑
Translated by

Compiler or interpreter

- easy to modify
- It is designed to run on multiple machine

- All at once translate and ~~the~~ ~~compute~~ computer then execute the machine language that the compiler produce.

Jav, C++ use compiler
1) native code compiler
2) cross compiler

~~the~~ Program that read source code one statement at a time, translates the statement into machine language, ~~exec~~ executes the machine language statement, then continues with the next statement

- native code compiler - that is intended to produce machine language to run on the same platform that the compiler itself runs on is called natve code compiler

- Cross compiler - ~~They~~ It produces machine language that is intended to run on a different platform than it runs on

This about compiler

- ~~Interpreter (like python, Ruby)~~

● Compiler code run faster > Interpreter code

because, Compiler code is translated into machine code befor execution, while an interpreter runs slower since it translates line by line during execution.

But (ခဏခဏ) frequently use interpreter for developing and testing ~~Soure~~ Source code for new programs

↑

why? ( interpreter [illegible] programs [illegible] )

- Ease of debugging ( error can be caught and fixed quickly since code runs line by line )
- the same code ~~run~~ Can run different platforms without recompiling.

Programming paradigms ( style or way of programming )

1) Imperative programming.

The program describes ~~*~~ a sequence of steps that that change the state of the computer as each one is executed in turn

Subset of this

1) Structured programming

- is a methode of writing programs using Control structure to improve clarity and reduce complexity

2) procedural programming

- writing programs on sequens of ~~Procedure~~ Procedures

It can combine sequence of instructions into a procedure so that these instructions can be invoked from many places. without restoring or duplicating the same instructions.

(01) what is the diffrence between procedure and ~~function~~ function* programming?

- function return a value and ~~p~~ procedure do not.

---

- Assembly language -& Not structured or procedural
- C - imperative and structured.

~~to page 29 compleated~~

↦ Logical Programming

rules are written as logical clauses with a head and body. Facts are expressed similar to rules but without body.

Functional Programming

* you can pass a function as an argument to another function, or a function may return another function.
ex - F#, LISP, Scheme.

↦ object-oriented programming.

This paradgm allows to organize software as a collection of objects that consist of both data and behavior.

ex:- python, Java, c++, c#

↦ Software development / Application development.

Software development is proceed by which stand-alone or individual software is created using a specific programming language.

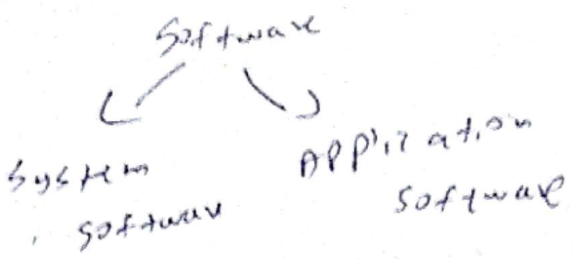
(01) what is Software Development Lifecycle (SDLS)

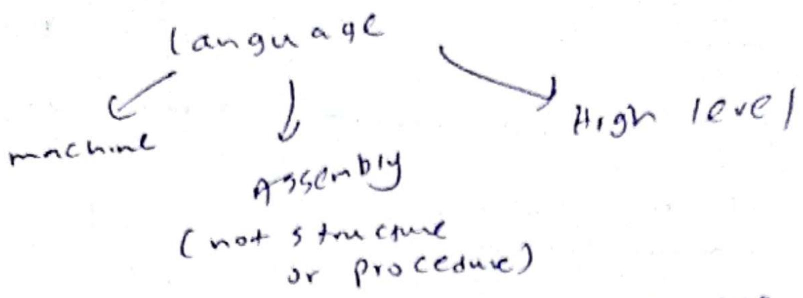
planning → Analysis and Requirement gathering → Design → Development → Testing →

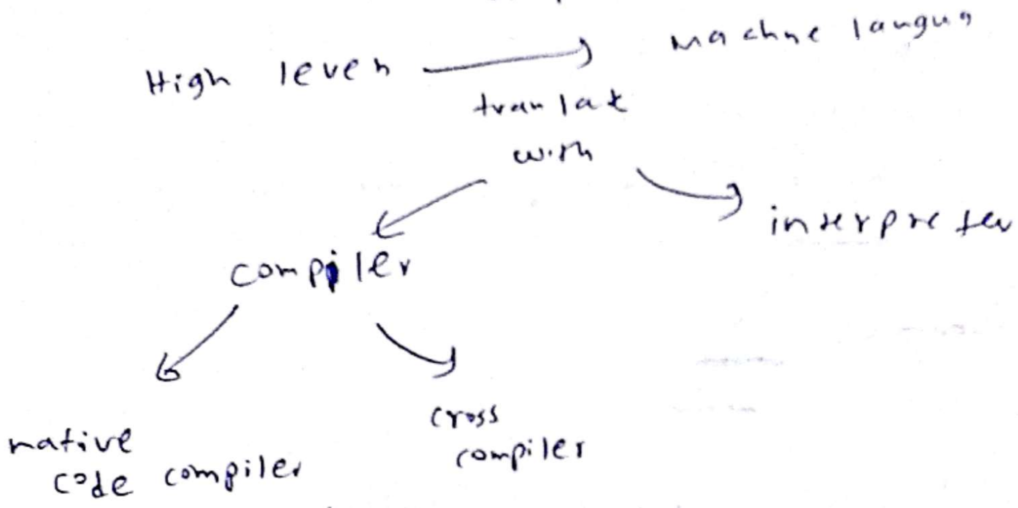
→ Deployment → Maintenance

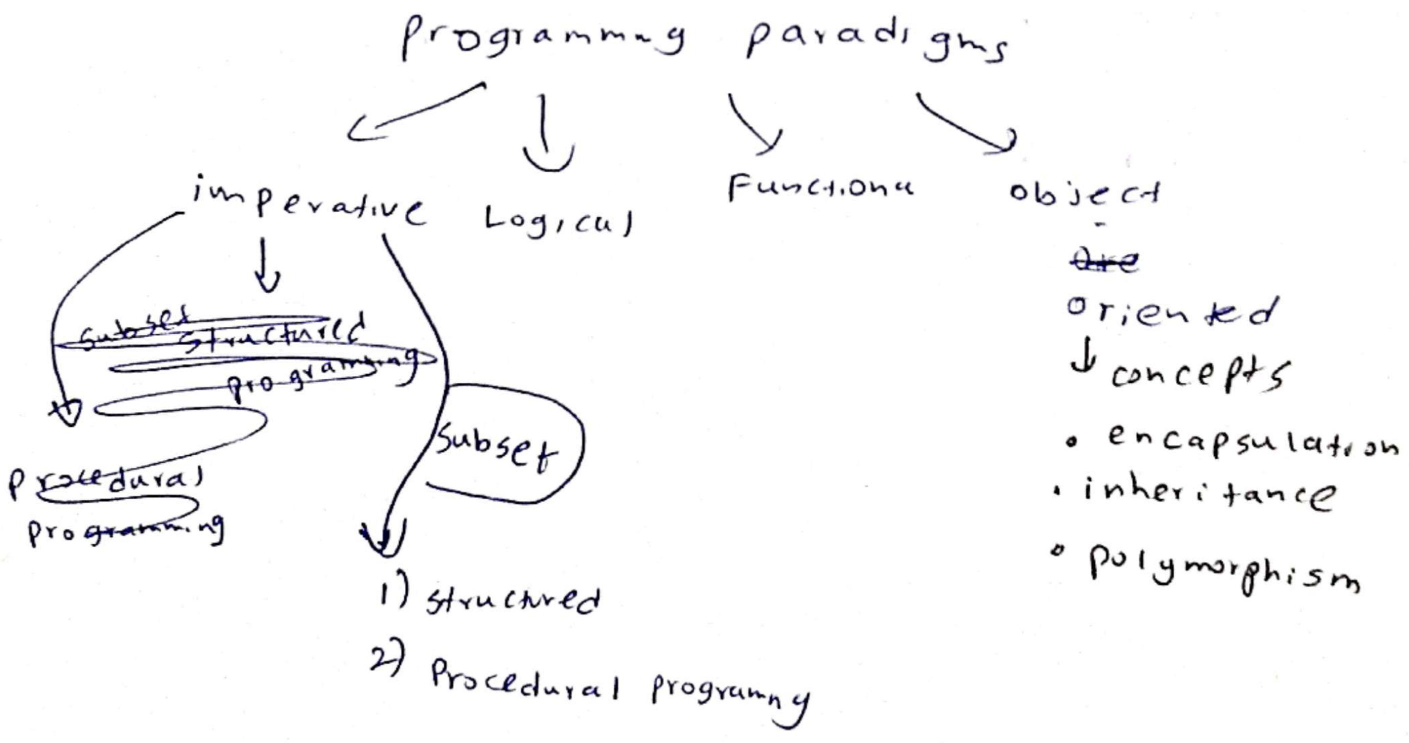
(02) why SDLS ?

to get product cost effective and of high quality.


Software
System software
Application Software


language
machine
Assembly
(not structure or procedure)
High level


High leven
machne languag
translate with
compiler
interpreter
native code compiler
cross compiler


Programming paradigms
imperative
Logical
Functiona
object
~~Are~~
oriented
concepts
• encapsulation
• inheritance
• polymorphism
~~Subset structured programming~~
Procedural programming
Subset
1) structured
2) Procedural programmny