Programming Fundamentals

Introduction to Program Design

Week 1 | Prasan Yapa

ROBERT GORDON UNIVERSITY ABERDEEN
TEF Gold
INFORMATICS INSTITUTE OF TECHNOLOGY

Learning Outcomes

- Covers part of LO1 & LO2 for Module
- On completion of this lecture, students are expected to be able to:
  - Identify main components in program design.
  - Analyze a real-world problem to break down its requirements .
  - Demonstrate competence in program design to solve problems.

What is Programming?

- Think about these machines. Their functionality

What is Programming?

- Now think about the Computer. And its functionality.

What is Programming?

- A computer program is a list of instructions that tell a computer what to do. Everything a computer does is done by using a computer program.

What is Programming?

- A computer program is a list of instructions that tell a computer what to do. Everything a computer does is done by using a computer program.
- Computers, although quite sophisticated electronically, are, nevertheless, unintelligent*.
- In order for them to carry out a task, they must be instructed how to complete the task.
- Instructions must be given in a language that the computer understands.
- A computer program is therefore a series of instructions to carry out a particular task written in a language that the computer understands.

What does a Programmer do?

Gather requirements

Prepares instructions of a computer program

Runs the instructions on the computers

Tests to see of its working properly

Writes up the program documentation

The Programming Process

Define	Define the problem :Analyze the problem, Produce a written agreement (SRS) containing input, the processing required and Output. Mainly Business Analyst is responsible for this
Design /Plan	Design /Plan the solution : Once the programmer has defined the problem, they begin to design the specific steps that will be used in the program. Systems Architect is responsible for this. The commonly used tools that help the programmer design the solution are •Flowcharts, Pseudo codes
Code	Code the program : Express the algorithms (flowcharts/ pseudo codes) in a programming language. Software Engineer is responsible for this. The source codes and comments can be considered as the documentation generated at this stage.
Compile, Debug & Test	Compile, Debug & Test the program. After coding the program, you must prepare to test it on the computer. Tests help detect the errors and to ensure that the program does what it is supposed to do. Software Quality Assurance Engineers are responsible for this. Test cases are the documentation generated at this stage.
Maintain	Maintain the program. Updates, Patches a d bug fixes.

The Evolution of Programming Languages

- ## Machine languages

  Machine languages (first-generation languages) are the most basic type of computer languages, consisting of strings of numbers the computer's hardware can use. Different types of hardware use different machine code. For example, IBM computers use different machine language than Apple computers.

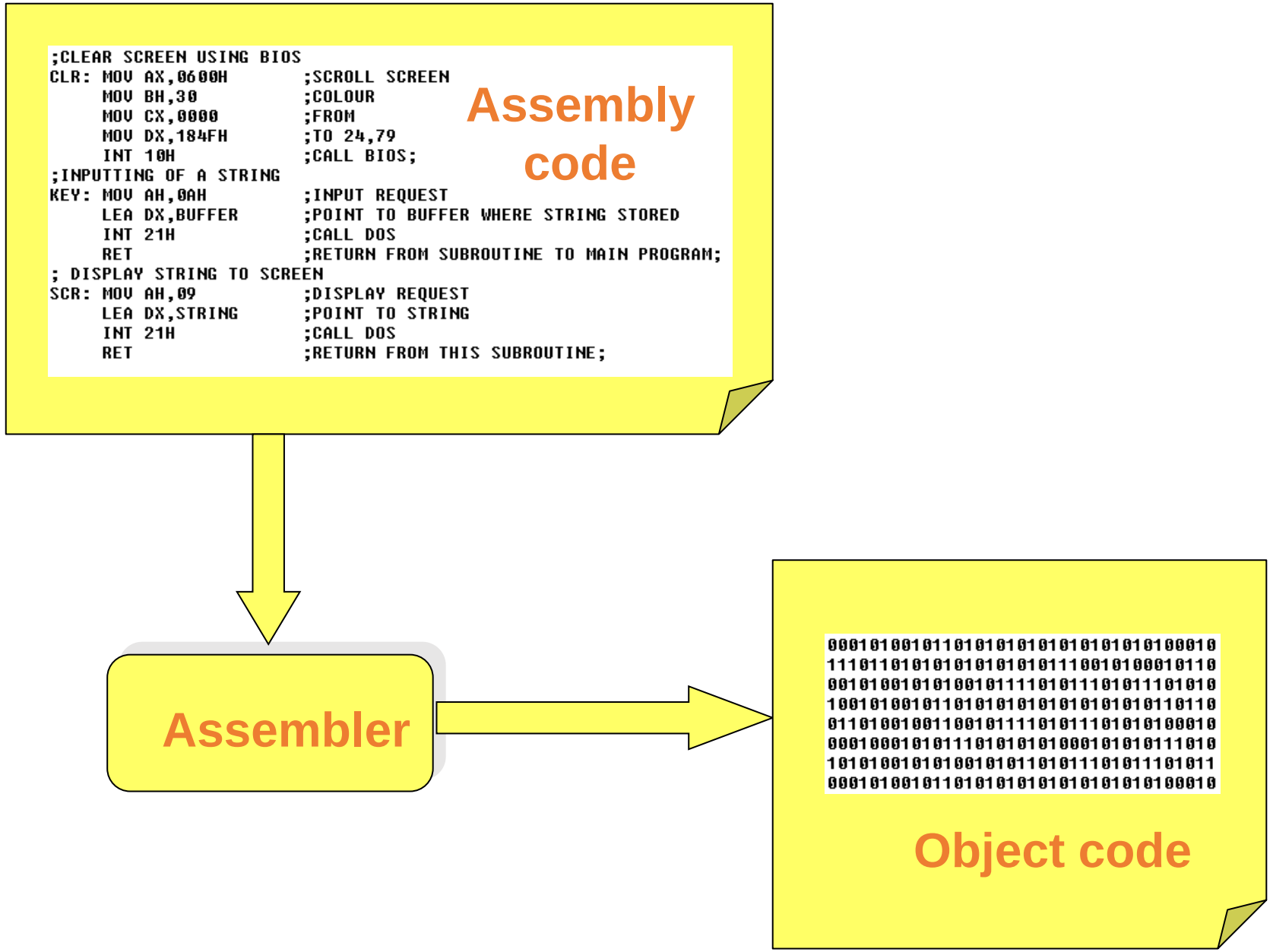
- ## Assembly languages

  Assembly languages (second-generation languages) are only somewhat easier to work with than machine languages.

  To create programs in assembly language, developers use cryptic English-like phrases to represent strings of numbers.

- ## Higher-level languages

  Higher-level languages are more powerful than assembly language and allow the programmer to work in a more English-like environment.


;CLEAR SCREEN USING BIOS
CLR: MOV AX,0600H ;SCROLL SCREEN
MOV BH,30 ;COLOUR
MOV CX,0000 ;FROM
MOV DX,184FH ;TO 24,79
INT 10H ;CALL BIOS;
;INPUTTING OF A STRING
KEY: MOV AH,0AH ;INPUT REQUEST
LEA DX,BUFFER ;POINT TO BUFFER WHERE STRING STORED
INT 21H ;CALL DOS
RET ;RETURN FROM SUBROUTINE TO MAIN PROGRAM;
; DISPLAY STRING TO SCREEN
SCR: MOV AH,09 ;DISPLAY REQUEST
LEA DX,STRING ;POINT TO STRING
INT 21H ;CALL DOS
RET ;RETURN FROM THIS SUBROUTINE;
Assembly code
Assembler
0001010010110101010101010101010100010
1110110101010101010101110010100010110
0010100101010010111101011101011101010
1001010010110101010101010101010110110
0110100100110010111101011101010100010
0001000101011101010101000101010111010
1010100101010010101101011101011101011
0001010010110101010101010101010100010
Object code

Third-Generation Languages

- Third-generation languages (3GLs) are the first to use true English-like phrasing, making them easier to use than previous languages.

FORTAN	**C**
COBOL	**C++**
BASIC	**Java**
Pascal	**ActiveX**

Fourth-Generation Languages

- 4GLs may use a text-based environment (like a 3GL) or may allow the programmer to work in a visual environment, using graphical tools.

Visual Basic (VB)
VisualAge
Authoring environments
Maple, Mathematica, Postscript, SPSS, SQL

Fifth-Generation Languages

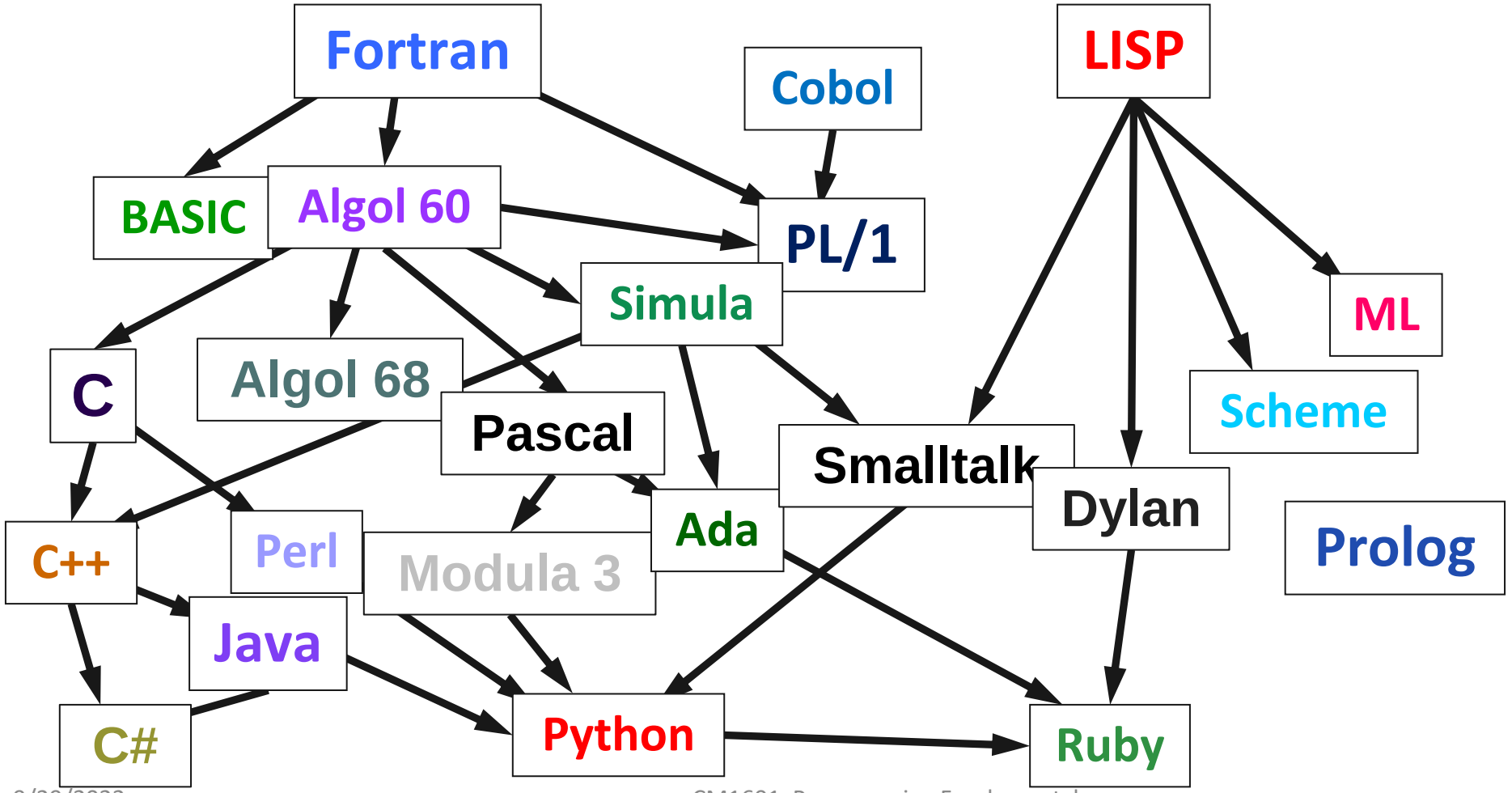
- Fifth-generation languages (5GLs) Fifth-generation languages (5GLs) are an issue of debate in the programming community – some programmers cannot agree that they even exist.

- These high-level languages would use artificial intelligence to create software, making 5GLs extremely difficult to develop.

- Solve problems using constraints rather than algorithms, used in Artificial Intelligence

Prolog

A Family Tree Of Languages



Procedural versus Object-Oriented Programming

- Procedural programming is based on a structured, top-down approach.
- The approach concentrates on what a program must do and involves identifying and organizing the processes in the program solution.
- The problem is usually broken down into separate tasks or functions and includes top-down development and modular design

Procedural versus Object-Oriented Programming

- Object-oriented programming is also based on breaking down the problem; however, the primary focus is on the things (or objects) that make up the system.
- The system is concerned with how the objects behave, so it breaks the problem into a set of separate objects that perform actions and relate to each other.
- These objects have definite properties (attributes), and each object is responsible for carrying out a series of related tasks (methods)

What Is An Algorithm ?

- It is a set of detailed, unambiguous and ordered instructions developed to describe the processes necessary to produce the desired output from a given input.
- The algorithm is written in simple English and is not a formal document.

What Is Pseudocode?

- Pseudocode is structured English. It is English that has been formalized and abbreviated to look like the high-level computer languages.
- Statements are written in simple English.
- Each instruction is written on a separate line.
- Keywords and indentation are used to signify particular control structures.
- Each set of instructions is written from top to bottom, with only one entry and one exit.
- Groups of statements may be formed into modules, and that module given a name.

Data Types

- **Integer** - representing a set of whole numbers, positive, negative or zero e.g. 3, 576, –5
- **Real**-representing a set of numbers, positive or negative, which may include values before or after a decimal point. These are sometimes referred to as floating point numbers. e.g. 19.2, 1.92E+01, –0.01
- **Character**- representing the set of characters on the keyboard, plus some special characters. e.g. ‘A’, ‘b’, ‘$’
- **Boolean**-representing a control flag or switch that may contain one of only two possible values, true or false.

Data Types

- A data structure is a structure that is made up of other data items.
- **Array**-a data structure that is made up of a number of variables or data items that all have the same data type and are accessed by the same name. For example, an array called scores may contain a collection of students' exam scores. Access to the individual items in the array is made using an index or subscript beside the name of the array. For example, scores (3) represents the third score in the array called scores.
- **String**-a collection of characters that can be fixed or variable. For example, the string Jenny Parker may represent a student's name

Data Validation

- Data should always undergo a validation check before it is processed by a program.
- Correct type: the input data should match the data type definition stated at the beginning of the program.
- Correct range: the input data should be within a required set of values.
- Correct length: the input data – for example, string – should be the correct length.
- Completeness: all required fields should be present.
- Correct date: an incoming date should be acceptable

Class Activity

- Divide into groups of three.
- Follow the given case Study

Summery

- A computer program is a list of instructions that tell a computer what to do. Everything a computer does is done by using a computer program.
- Gathering requirements, Preparing instructions of a computer programs , Running the instructions on the computers ,Testing to see of its working properly, Writing up the program documentation are programmer's tasks.
- Procedural programming is based on a structured, top-down approach
- Object-oriented programming is also based on breaking down the problem; however, the primary focus is on the things (or objects) that make up the system.
- The algorithm is written in simple English and is not a formal document

Thank You !!