UGANDA CHRISTIAN UNIVERSITY

A Center of Excellence in the Heart of Africa

CSC1101
Structured Programming

Facilitators;

Lecturer: Ms. Mukalere Justine Thelma

Tutors:

- ✓ Mr. Kasozi Brian
- ✓ Mr. Wambete Joseph
- ✓ Mr. Musasizi Kenneth
- ✓ Mr. Tibenkana Christopher

Department of Computing & Technology

Faculty of Engineering, Design & Technology

May,2023

Reference text

Main Course Book

Kochan, G. S. (2015). Programming in C (4th ed.). Addison-Wesley

Note:

The E-Copy of this book is uploaded on the course page.

WEEK 1

INTRODUCTION TO C PROGRAMMING

Objectives & Learning Outcomes

Objectives

i. Explain the basic terminologies used in computer programming and write a basic C program.
ii. Explain and write functional C -programs.
iii. Explain and write structures and functions in C programs.

Learning Outcomes

i. Understand basic terminologies used in computer programming and be able to write a basic C program.

ii. Understand the C program structure and be able to write functional C - programs.
iii. Understand and apply the concept of structures and functions in C programs.

Assessment

Item	Weight
Tests/Quizzes	20%
Assignments	15%
Course Project	15%
Final Exam	50%

Tools

- C-Complier
- VS Code - Editor

Some terminology

- C*omputer program*
  - ✓*A* set of instructions used to operate a computer to produce a specific result.
- C*omputer programming*
  - ✓Writing computer programs
- P*rogramming languages*
  - ✓Languages used to create computer programs
- *Software*
  - ✓Program or a set of programs

MODULE 1: PROGRAMING LANGUAGES

- Unit 1: Computer Programming
  - ❑Introduction
    - ✓ Hardware
    - ✓ Software
    - ✓ Storage
    - ✓ Machine and Human sensible language
  - ❑Classification of Programming Languages
    - ✓ Low Level Language
    - ✓ High Level Language
    - ✓ Features of High-Level Language

- Unit 2: Program Design
  - ❑Characteristics of a good program
  - ❑Phases of Program Development

Introduction: Hardware

- Input unit and Output unit
- Memory unit
- ALU
- CPU
- Secondary storage
- Etc...

Input Unit and Output Unit

- Input Unit
  - Obtains information from various *input devices* and places this information at the disposal of the other units.
  - Examples of input devices: keyboard, mouse, etc.

- Output Unit
  - Takes information that has been processed by the computer and places it on various *output devices*.
  - Most output from computer is displayed on screens, printed on paper, or used to control other devices.

Memory Unit

- The memory unit stores information.
    - ➢Each computer contains memory of two main types: RAM and ROM.

❑RAM (*random access memory*) is volatile.

- ✓Your program and data are stored in RAM when you are using the computer.

❑ROM (*read only memory*) is non-volatile .

- ✓Contains fundamental instructions that cannot be lost or changed by the user.

ALU and CPU

- Arithmetic and Logic Unit (ALU)
  - ✓ALU performs all the arithmetic and logic operations.
  - ✓Ex: addition, subtraction, comparison, etc..
- CPU
  - ✓The unit is in charge of the overall operation of the computer.

Secondary Storage

- Used as permanent storage area for programs and data.
  - ✓Examples: magnetic tapes, magnetic disks and optical storage CD.
  - ✓ Magnetic hard disk
  - ✓Floppy disk
  - ✓CD ROM
  - ✓Etc …

Introduction: Software

- Application software and System software.
  - ❑ *Application software*
    - ✓ Consists of programs written to perform particular tasks required by the users.
    - ✓ E.g. ……………….??
  - ❑ *System software*
    - ✓ Collection of programs that must be available to any computer system for it to operate.
      - o E.g. operating systems, *language translators*

Introduction: Storage

- Open conversation….!

Introduction: Machine language

- Machine languages are the lowest level of computer languages.
- Consist of 1s and 0s.
- Control directly to the computer's hardware.
- Example:
- 00101010 000000000001 000000000010
- Consists of two parts: an instruction part and an address part.
  - ✓**Instruction part** (*opcode*) tells the computer the operation to be performed.
  - ✓ ***Address part*** specifies the memory address of the data to be used in the instruction.

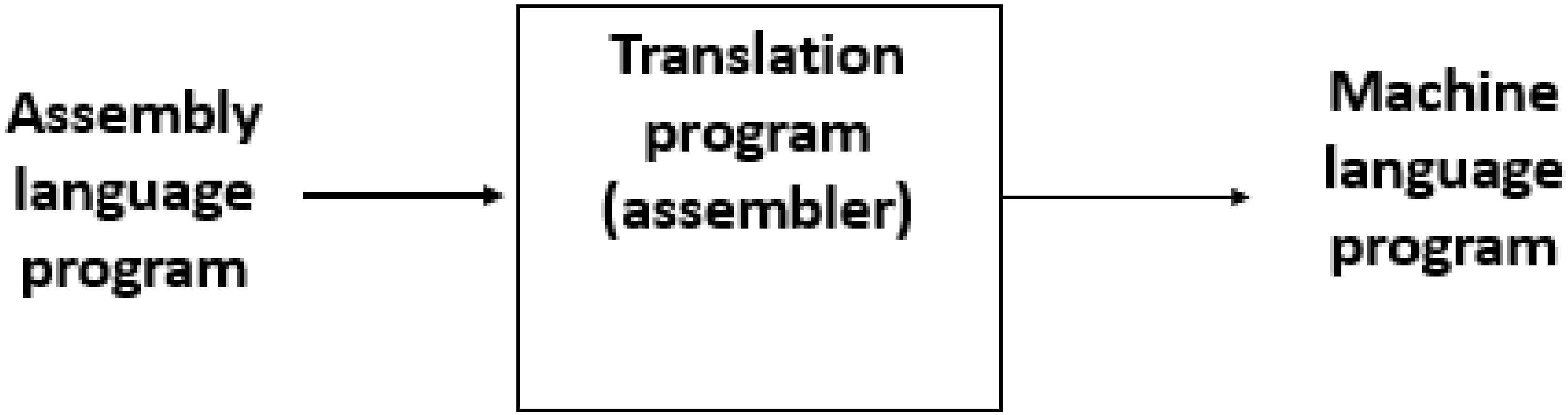
Introduction: Assembly languages

- Assembly languages perform the same tasks as machine languages, but use **symbolic names** for opcodes and operands instead of 1s and 0s.

```
LOAD BASEPAY
ADD OVERPAY
STORE GROSSPAY
```

- An assembly language program must be translated into a machine language program before it can be executed on a computer.

Assembler



Assembly language program

Translation program (assembler)

Machine language program

Introduction: Human sensible language

- Open conversation…!

Classification of Programming Languages: Low Level Language

- Open conversation…!

Classification of Programming Languages: High Level Language

- Create computer programs using instructions that are much easier to understand.
- Programs in a high-level languages must be translated into a low level language using a program called a *compiler*.
- A compiler translates programming code into a low-level format.
- High-level languages allow programmers to write instructions that look like every English sentence(s) and commonly-used mathematical notations.
- Each line in a high-level language program is called a *statement*.
  - ✓Example: Result = (First + Second)*Third

Popular High Level Language

- COBOL (COmmon Business Oriented Language)
- FORTRAN (FORmula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (named for Blaise Pascal)
- Ada (named for Ada Lovelace)
- C (whose developer designed B first)
- Visual Basic (Basic-like visual language developed by Microsoft)
- Delphi (Pascal-like visual language developed by Borland)
- C++ (an object-oriented language, based on C)
- C# (a Java-like language developed by Microsoft)
- Java

Program Design: Characteristics of a good program

- Open conversation…!

Program Design: Phases of Program Development

- Software development consists of three overlapping phases
    - ✓ Development and Design
    - ✓Documentation
    - ✓Maintenance
- The concerned is creating; readable, efficient, reliable, and maintainable programs and systems.

Program Design: Phase I: Development and Design

- The first phase consists of four steps:

1. ***Analyze the problem:*** Analyze problem requirements to understand what the program must do, what outputs are required and what inputs are needed.

2. ***Develop a Solution:*** *Algorithm* is a sequence of steps that describes how the data is to be processed to produce the desired outputs.

3. ***Code the solution:*** Translate the algorithm into a computer program using a programming language.

4. ***Test and validate the program***

Program Design: Phase II: Documentation

- Documentation requires collecting critical documents during the analysis, design, coding, and testing.
- There are five documents for every program solution:
  - ✓Program description
  - ✓Algorithm development and changes
  - ✓Well-commented program listing
  - ✓Sample test runs
  - ✓User’s manual

Program Design: Phase III: Maintenance

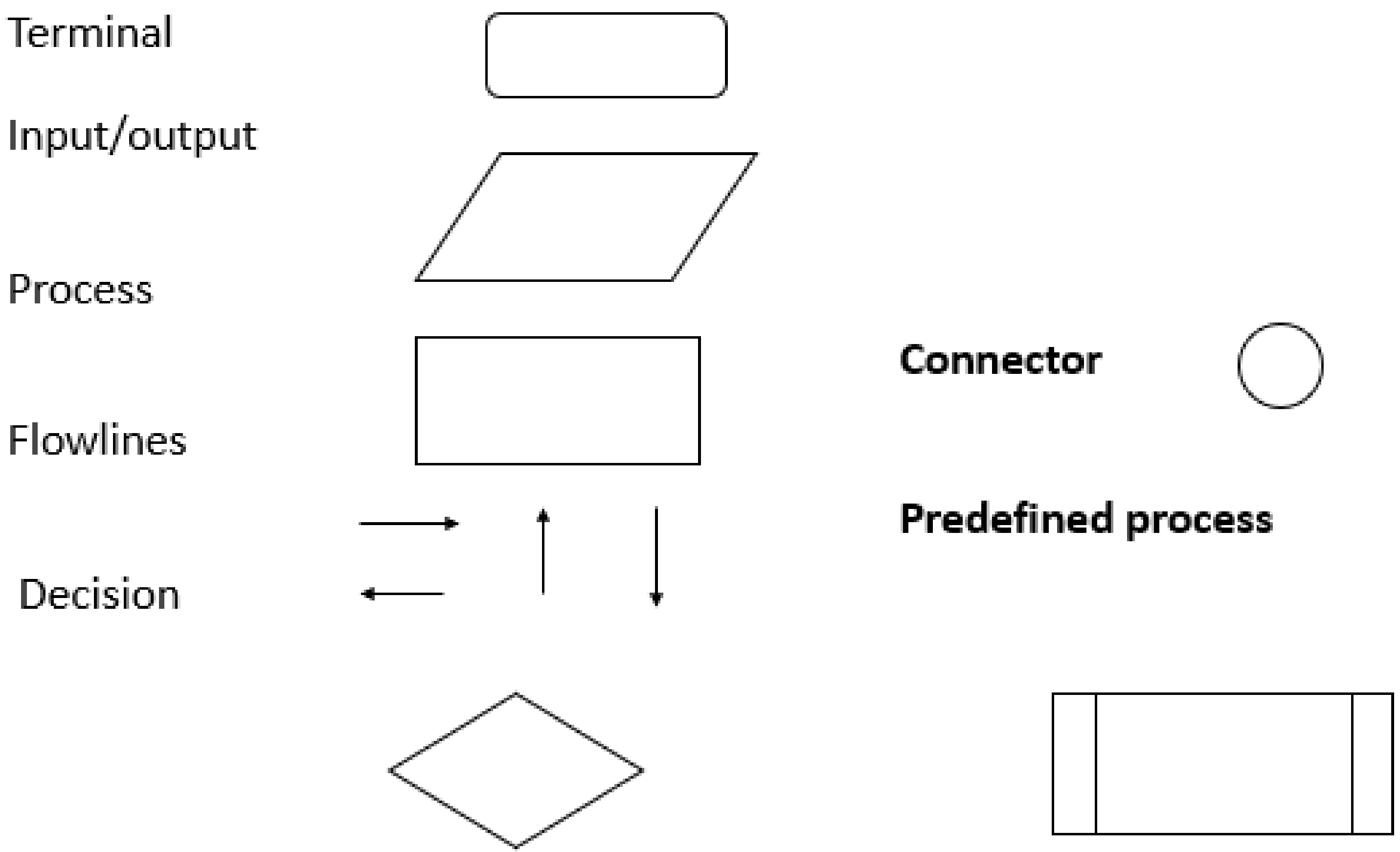
- This phase is concerned with
  - ✓Ongoing correction of problems/errors
  - ✓Revisions to meet changing needs
  - ✓Addition of new features.

Program Design: ALGORITHMS

- You can describe an algorithm by using *flowchart* symbols or *pseudo code*.
  - ✓***Flow chart*** is an outline of the basic structure or logic of the program using various symbols.
  - ✓***Pseudo code*** is a description of the program using words.

- ▪The use of pseudo code is increasingly gaining acceptance due to its flexibility when there is need for adjustment.

Program Design: Flowchart symbols



Terminal

Input/output

Process

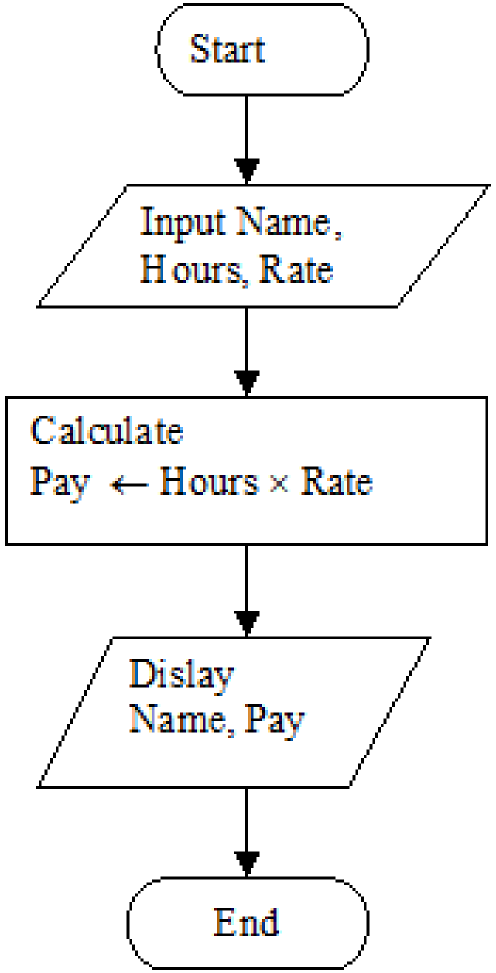
Flowlines

Decision

Connector

Predefined process

Program Design: Flowchart Example…



Note: Name, Hours and Pay are *variables* in the program.

Program Design: Algorithms in pseudo-code

- You can use English-like phases to describe an algorithm which is called ***pseudocode***.
  - Example:

```
Input the three values into the variables Name, Hours, Rate.
Calculate    $Pay = Hours \times Rate$.
Display Name and Pay.
```

Individual Work

- *Flow chart and pseudocode for a program helping with withdraw of funds from one’s personal bank account*