**COMPUTER ENGINEERING**

Author: **Q15919676**

Course Title: **Computer Fundamental**

Module Leader : **PRINS BUTT**

Date: 10/09/22

Table of Contents

[1. Summary of computing concepts 2](#_Toc112585984)

[**1.1 computer system architecture** 2](#_Toc112585985)

[**1.2 Data storage and representation** 3](#_Toc112585986)

[**1.4 Number System** 4](#_Toc112585987)

[**1.5 File organization and disk storage** 5](#_Toc112585988)

[2. Summary of construction 5](#_Toc112585989)

[3. Designing and developing computer programs 10](#_Toc112585990)

[**3.1 Overall architecture** 11](#_Toc112585991)

[**3.2 Modules and Function** 12](#_Toc112585992)

[4. Recommendations 13](#_Toc112585993)

[5. References 14](#_Toc112585994)

[14](#_Toc112585995)

# Summary of computing concepts

Computer Engineering allows learners to acquire knowledge of various computer-related concepts critical in their education journey. During the introduction to computer science, key concepts such as computer architecture, data storage and representation, and how number systems work are used to give background information regarding the course.

## **1.1 computer system architecture**

The arithmetic logic unit, an input unit, a storage unit, a control unit, and an output unit are a few examples of the system's architectural components. Each of these units performs its unique function depending on the instructions that are given to the computer. For instance, the input unit comprises computer parts such as a keyboard, video camera, and mouse. It is used to receive information from the instructor. The output unit is composed of things such as the printer and the monitor screen, communicating data, or showing feedback after the computer memory has processed it. Arithmetic and the logical unit (ALU) act as the specialized part of the brain that performs mathematical calculations and helps the computer make straightforward decisions (Huang et al., 2021).

## **1.2 Data storage and representation**

Data representation involves various tasks, including how data is stored, processed, and transmitted from one unit to another. Any computerized device such as smartphones and laptops stores data in digital formats, making it easy to handle by electronic circuity. The only language computers can understand is the data representation series of 0s and 1s. In other words, each 0 or 1 represents a specific message utilized by the computer memory to provide feedback to the computer instructor. According to Abdulkareem et al. (2019), the value “1” represents an “on” bit while a “0” represents an “off” bit, making up the binary form understood as the computer language. Data representation and storage are discussed together as they depend on each other to create meaning out of the information given by the computer instructor. According to Huang et al. (2021), the bit, the most minor yet fundamental component of computer memory, is crucial to comprehend when storing data. It is a minimal magnetic area on a hard drive, like a DVD or compact disk. A byte is created when 8 bits are combined. A word that can be read and interpreted by the computer teacher is made of 4 bytes or 32 bits. Hence, data storage is ensured by storing each data value using one or more bytes of memory (Huang et al., 2021). Data storage in a computer is an extensive collection of bits. Each storage format hardware and even the operating system determine the number of bytes and words that should be used for an individual data value.

## **1.4 Number System**

Computers only understand numbers. As a result, when an instructor types words or letters, the computer memory translates them into numbers. While computer keyboards are lined with digit symbols, it is essential to note that each character represents unique values depending on their position in the number. The value of each digit can be calculated using the digit itself, the number system's base, and the digit's position inside the number. The entire number of digits accessible within the number system is the base, according to Huang et al. (2021). Every point in the decimal number system corresponds to a particular base power, which illustrates how the computer memory interprets it (10). First, there are ten digits in the decimal number system: 0 through 9. 1342 can be comprehended by considering each digit and its placement. 4 in the tens, 3 in the hundred, 2 in the unit position, and 1 in the thousands. As a result, the number can be expressed as follows:

(1\*1000) +(3\*100) +(4\*10) +(2\*1)

(1\*103) + (3\*102) + (4\*101) + (2\*l00)

1000+300+40+2

1342

## There are many number systems, including the binary number system, which employs base two and only has the numerals 0 and 1 as its digits. Another is the base-16 Hexadecimal number system, which uses digits 0-9 and letters A-F, and the base-8 octal number system, which utilizes digits 0-7.

## **1.5 File organization and disk storage**

Files are commonly used as computer storage spaces where related data can be put together and stored for future retrieval. File organization explains how the data is stored as it determines its accessibility, storage, flexibility, and efficiency. File organization is composed of four methods, including serial, random, indexed-sequential, and sequential organization. The sequential organization puts records in a particular order sorted by a key field—data retrieval demands sequential searching. The random organization is also known as direct file storage, in which documents are stored but accessed directly.

Data retrieval from a magnetic or optical disk at random is an actual case. It enables the sequential storage and access of each file. One illustration is the use of magnetic tapes for storage.

# 2. Summary of construction

Writing trustworthy and meaningful software for a particular purpose is computer program creation. Integration testing, coding, debugging, unit testing, and even verification is necessary to build a program. Knowledge from other computer disciplines, such as software design and testing, is used in program building.

Software design and testing are critical aspects of program construction because the process requires designing and testing before it can be approved as a reliable program. Hence, designing and testing are core activities of program construction. The program type will set precise boundaries between construction, designing, and testing (Abdulkareem et al., 2019). Software or program construction comprises certain fundamentals, including change anticipation, minimizing complexity, construction standards, and structure for verification. While engaging in program construction, it is critical to note that most programs will change over time. As technology advances, there will be room for improvement of the constructed program to enable it to continue being viable in the new society. Abdulkareem et al. (2019) explain why change anticipation is crucial to program construction. Hence, while developing the program, the developer must understand the techniques that support change anticipation, including communication methods, programming languages, tools such as diagrammatic standards, and platforms like the programmer interface.

Construction standards that affect program construction activities include the Use of External Standards. Such measures may depend on construction tools, language, technical interfaces, and how to program construction interactions with other software activities (Abdulkareem et al., 2019). Construction standards can be retrieved from various sources, including software and hardware interface specifications. A real-life example is international organizations such as ISO and Object Management Group (OMG) specifications. A program constructor can also rely on internal standards that provide organizational guidelines that enhance construction coordination for verification, change anticipation, group activities, and minimizing complexity.

* 1. **Minimizing complexity**

Minimizing complexity is a critical aspect of program construction. Even though most people delight in using computers and their programs, one of the challenges they encounter is the severely limited ability to hold complex structures. As a result, it is critical to consider how complex systems can be represented and stored while developing a program. Minimizing complexity can be achieved through developing this aspect as one of the strongest drivers in software construction. If a program developer desires to achieve a reduced complexity, they can ensure an emphasis on the creation code, making it as simple and readable as possible.

**2.2 Constructing for verification**

Tanaka et al. (2019) define constructing for verification as building software so that other software engineers can easily detect any faults or system issues. Some techniques enable creating for validation, such as code reviews, organizing code, unit testing, and other coding standards that are used to support automated testing. These coding standards can be used to ensure the restricted use of complex language structures.

**2.3 Process of program construction**

Various steps guide computer program construction. First, it is essential to note that programs are created for a reason whereby in most cases, they are built to solve an existing problem in society. Problem formulation, program design, coding, debugging, testing, documentation, and maintenance are steps in creating a program.

In problem definition, the program developer focuses on establishing the critical problem that needs to be solved. Tanaka et al. (2019) claim that this step allows the developer to decide which program or software can be used to eliminate an existing or suspected future problem. The problem must be defined and understood well. This is the stage during which factors such as memory requirement, input/output, interfaces, error handling, and processing requirement are considered.

Program design is done once the problem has been identified. The program developer must be competent in using tools such as flowcharts and algorithms to create a unique program. Thirdly, coding is done once the program design has been completed. Coding does not consume a lot of time and is a small aspect of the entire process. It entails the elimination of all syntax errors, such as spelling errors, undefined levels, and missing labels. Some guidelines can ensure effective coding, including giving simple expressions, using comments, and indenting the code appropriately.

Debugging involves allowing the system to detect and correct any remaining errors. Tanaka et al. (2019) suggest that this is a critical process called program validation. However, errors can still occur as the program continues to be developed. Some of these errors involve mixing up numerals and letters and inverting conditions, like when someone jumps on zero rather than on not zero. The program must subsequently be evaluated to establish its effectiveness and whether the goal of eliminating the specific problem has been achieved. During program design, a testing plan is formulated. Testing gives the developer ample time to understand the program specification thoroughly. It is advisable to include minimum and maximum values for every variable as test data during this stage.

Documentation involves activities that keep the software in the desirable conditions. This is the step during which room for future modification is created. Lastly, maintenance includes updating and correcting the program in case of any changed conditions. Field experiences are also accounted for during maintenance. Maintenance becomes extremely important when specifications change, equipment is chosen differently, or faults are easily discovered during the program's execution.

1. Problem definition
2. Program design

Process of program construction

1. Coding
2. Debugging
3. Testing
4. Documentation
5. Maintenance

Figure 1: the process of program construction

# 3. Designing and developing computer programs

Program design provides the programmer with the steps to be accomplished before coding is commenced. Program developers must understand that other programmers may use their programs in the future. For this reason, it is important to document each program design step correctly for easier future maintenance. The three main components of program design are developing test data, comprehending the program, and employing design tools to build a model. The program's outputs, inputs, and processing (IPO) must be considered. This approach is critical, especially for beginning programmers who may not have extensive programming knowledge. By reviewing program design steps, an individual should be able to imagine what the IPO will look like, instructions given to the computer via its keyboard, and even the kind of processing changes that can be done.

The design tools can be used in program designing to create a model. Tanaka et al. (2019) suggest that this is another essential step that can guide other programmers to operate a program they did not create. The tools include the pseudocode, which allows the programmer to document the algorithm or logic of the program. Additionally, one should know how to generate test data, which entails supplying input values and projecting outputs. For small programs, these activities may be straightforward.

## **3.1 Overall architecture**

Three categories make up the overall computer architecture: system design, instruction set architecture (ISA), and microarchitecture. The physical framework of a computer system, including all of its hardware components, is known as the system design. They include memory controllers, data processors, graphics processing units (GPU), and multiprocessors. The instruction set architecture (ISA) holds the functions and capabilities of the central processing unit (Tanaka et al., 2019). ISA determines which programming language and the kind of programming that should be processed. Every computer uses this software to run, with the help of an operating system like windows on a PC—microarchitecture functions as a computer organization unit. According to Tanaka et al. (2019), a microarchitecture is the physical implementation of an ISA in a particular processor.

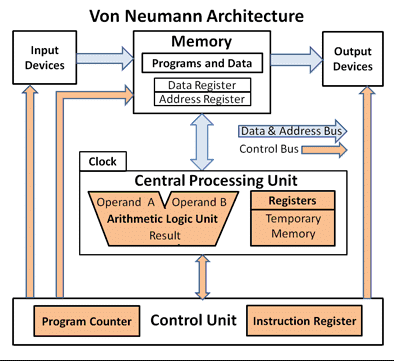


Figure 2: (Admin, 2021)

## **3.2 Modules and Function**

Understanding that a module is equivalent to a code library or a file with a set of actions or functions is crucial while constructing one. For instance, while creating an application such as a calculator. The module should help organize the related classes, procedures, or codes. Every module has three significant components: variables, definitions, and implementation of courses and processes that can be used in different programs (Tanaka et al., 2019). If an application is being made for a calculator, It is necessary to incorporate some operations, like multiplication, addition, subtraction, and division. The entire code will be broken into separate parts to create a single module for all those operations. It is also possible to create different modules for every function.

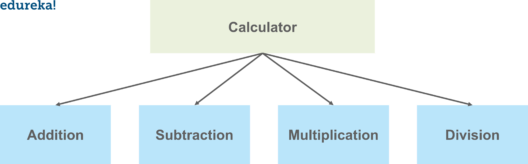


Figure 2 - Database model (Goyal, 2021).

# 4. Recommendations

Today's world is centered around computer science, making most daily activities dependent on knowledge. The most recent technologies, such as cloud computing and artificial intelligence, have been achieved through computer science. The computer science industry has become increasingly reliable and popular due to the advantages it has given to society. For instance, the recent covid-19 pandemic has made most companies utterly reliant on technology experts and cybersecurity professionals, making them more needed and a treasured part of society. Technology has attempted to solve many problems experienced by humankind (Tanaka et al., 2019). Most of them have found solutions by using the knowledge of computer science. For instance, the need to satisfy the human nature of socialization has been achieved through the constant interaction of people worldwide. This has been made easy by developing computer programs and applications that can be used via mobile phones, such as Twitter and WhatsApp, where people from all walks of life can meet and interact.

While certain aspects of cybersecurity have been achieved, computer science should seek to improve cybersecurity guidelines to ensure that no information can be hacked or leaked by people who hardly have any knowledge of computer science. Again, even though artificial intelligence has been one of the most intriguing and controversial aspects, it is crucial to continue enhancing knowledge in this sector. Computer science experts should continue exercising their skills as there is no shortage of opportunities to develop real-world applications using the technology.

# 

# References

Abdulkareem, K. H., Mohammed, M. A., Gunasekaran, S. S., Al-Mhiqani, M. N., Mutlag, A. A., Mostafa, S. A., ... & Ibrahim, D. A. (2019). A review of fog computing and machine learning: concepts, applications, challenges, and open issues. *IEEE Access*, *7*, 153123-153140.

Admin. (2021). What is computer architecture? *Get Education Skills.* <https://geteducationskills.com/computer-architecture/>

Goyal C. (2021). Working with Modules in Python: Must Know Fundamentals for Data Scientists. *Analytics Vidhya*. Retrieved 28th August 2022 <https://www.analyticsvidhya.com/blog/2021/07/working-with-modules-in-python-must-known-fundamentals-for-data-scientists/#:~:text=In%20Python%2C%20Modules%20are%20simply,to%20include%20in%20your%20application>.

Huang, M. Q., Ninić, J., & Zhang, Q. B. (2021). BIM, machine learning and computer vision techniques in underground construction: Current status and future perspectives. *Tunneling and Underground Space Technology*, *108*, 103677.

Tanaka, R., da Silva, R. F., & Casanova, H. (2019, November). Teaching parallel and distributed computing concepts in simulation with a wrench. In *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)* (pp. 1-9). IEEE.