INTRODUCTION TO COMPUTER SCIENCE AND PROGRAMMING IN C LANGUAGE

BABCOCK UNIVERSITY

COMPUTER SCIENCE DEPARTMENT

Why COSC111?

- Write out what you want to gain from this course.
- Write out your plan to achieve this.

Special Notice

- Attendance is very important
- The first mid-semester test will take place in the fourth week of resumption
- The second mid-semester test will take place in the eighth week of resumption
- A mini project will be given at the sixth week of resumption, to be submitted on the ninth week of resumption. The project will be openly defended.

How to survive this course

- Work consistently: do not leave your assignment undone till the last moment, do it the same day you are given, this will get you prepared for the next class.
- Ask: when you don’t understand something ask for it becomes hard to work when you don’t understand something.

How to survive this course

For students to do well in programming you must:

- Attend practical classes as well as lectures and organized tutorials
- Start practical assignment as soon as you are given
- Do the tutorial exercises
- Seek help when things get difficult

Help……

- If you get help from your friends or tutors on assignment, make sure you understand it.
- You can also search the internet to get resources on learning C programming.
- You may mail me at ……………… if you still don’t understand what I taught and you have asked your colleague whom you think understands.
- You can also meet me at schedule time for interaction.

Things to do

* Register the following with the Course Representative:
* Matric
* Name
* Phone number (WhatsApp)
* E-mail address

DEFINING A COMPUTER

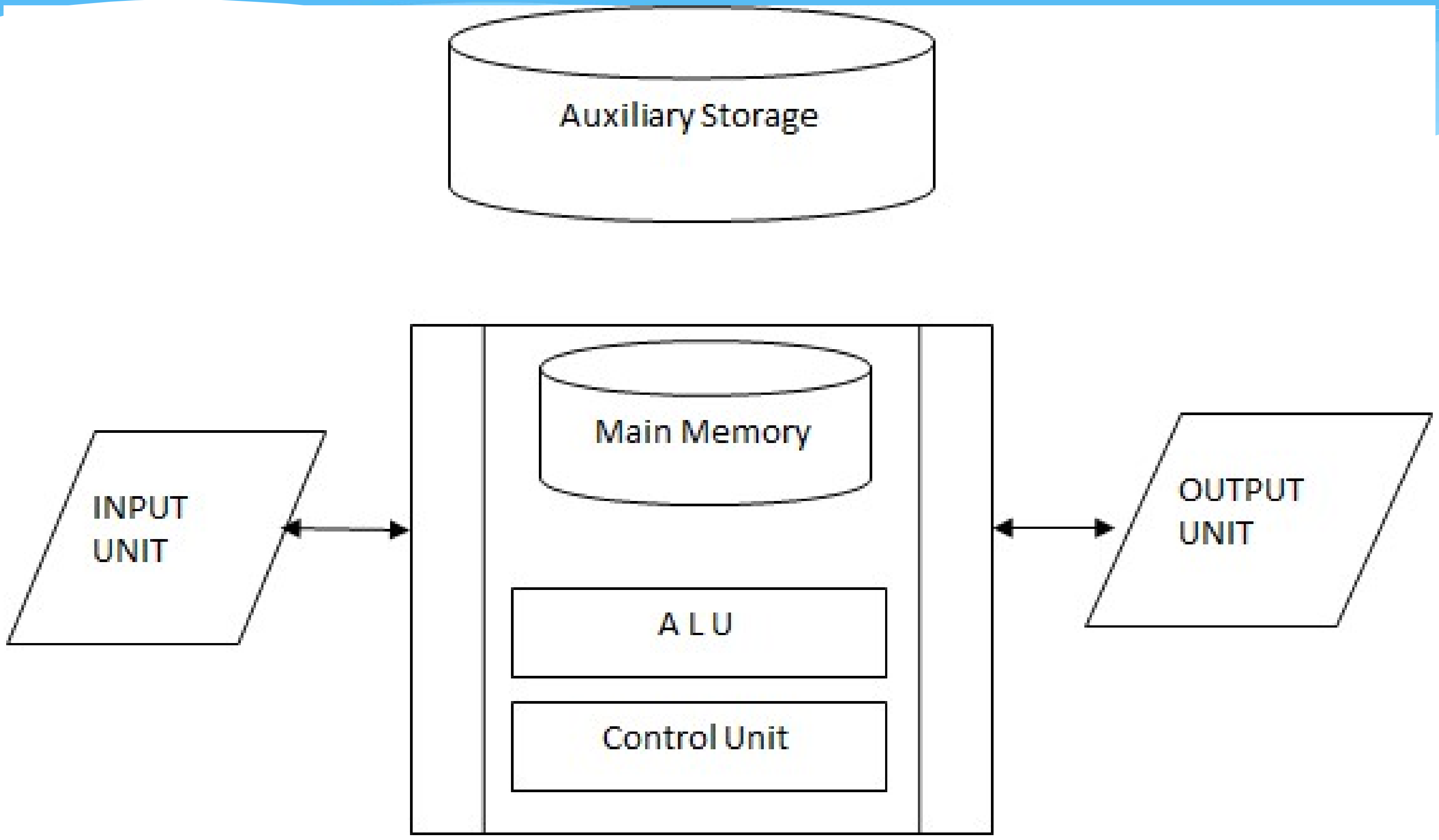
- A computer acts based on **instructions**.
- A computer is **not intelligent**, because it cannot analyze a problem on its own.
- It **needs a human expert** to analyze a problem and develop a program (a set of instruction) for the computer.
- The computer can only carry out the instructions given to it.

Computers Are Electronic Data-Processing Devices

* A computer’s four major functions:
    * Accepts data
    * Processes data into information
    * Outputs data or information
    * Stores data and information

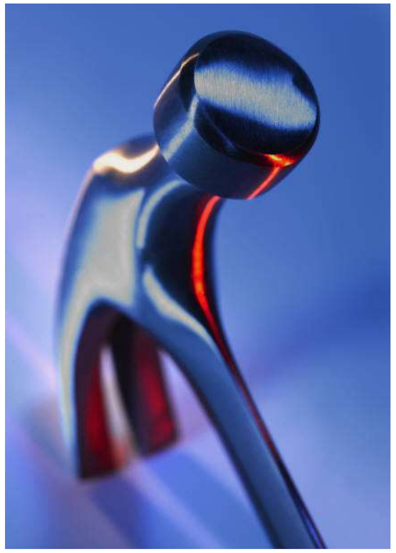
Thus **data is the raw fact** which computer uses. Data can be in form of diagram, numbers, alphabets, words, video etc.

Three Basic Units of a Computer



Defining Computer Science

- Computer Science is about building **abstract models** of real-world objects or phenomena, with the aim of using the model to **solve problems** by **automating** the models through **the use of algorithm,** and **implementing** the algorithm in form of **instructions** which the **computer** can understand.

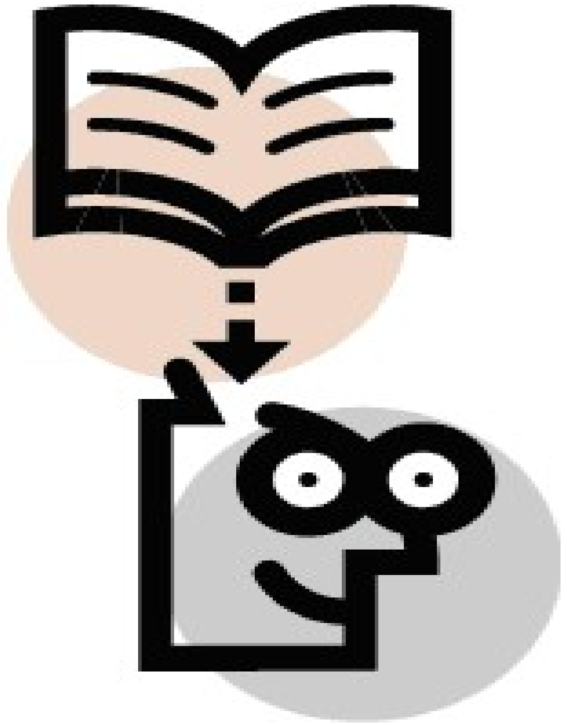


Why Computer?

- Computers are used to implement codes because they are amazingly fast at executing algorithms on data.

- Note that we could even use pencil and paper to work our algorithm but this will take a longer time and will easily attract error.

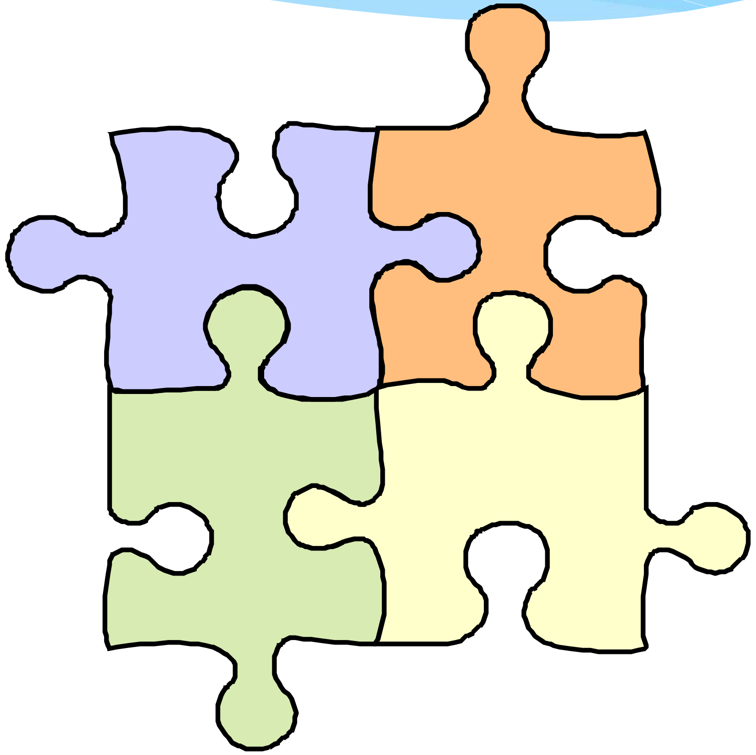
Computer Scientist



- Thus Computer Science is about solving problem.
- Therefore Computer Scientists are thinkers, they think on how problems should be solved.
- Thus abstraction starts from the mind.

Computer Scientist

- As you think, you have a picture of how a problem should be solved in your **mind** (i.e. an abstraction). Thinking then leads to **modeling**, this means you now make the picture of the solution in your mind into a model.



Computer Scientist

- A computer scientist then **automates** the model through the use of **algorithm**.
- Automation is an act of providing working solution to a problem through the use of algorithm.

Trends of Programming and Programming languages

* Native language of a computer machine is 1's and 0's
* The 1's and 0's is called the machine language
* The programming was done through punched cards, plugged boards, or front panel switches
* Machine language grew from 1's and 0's to hexadecimal (four binary) due to the advent of terminal such as keyboards, and monitors.
* Machine language was tedious, difficult, and only fits for small programs.

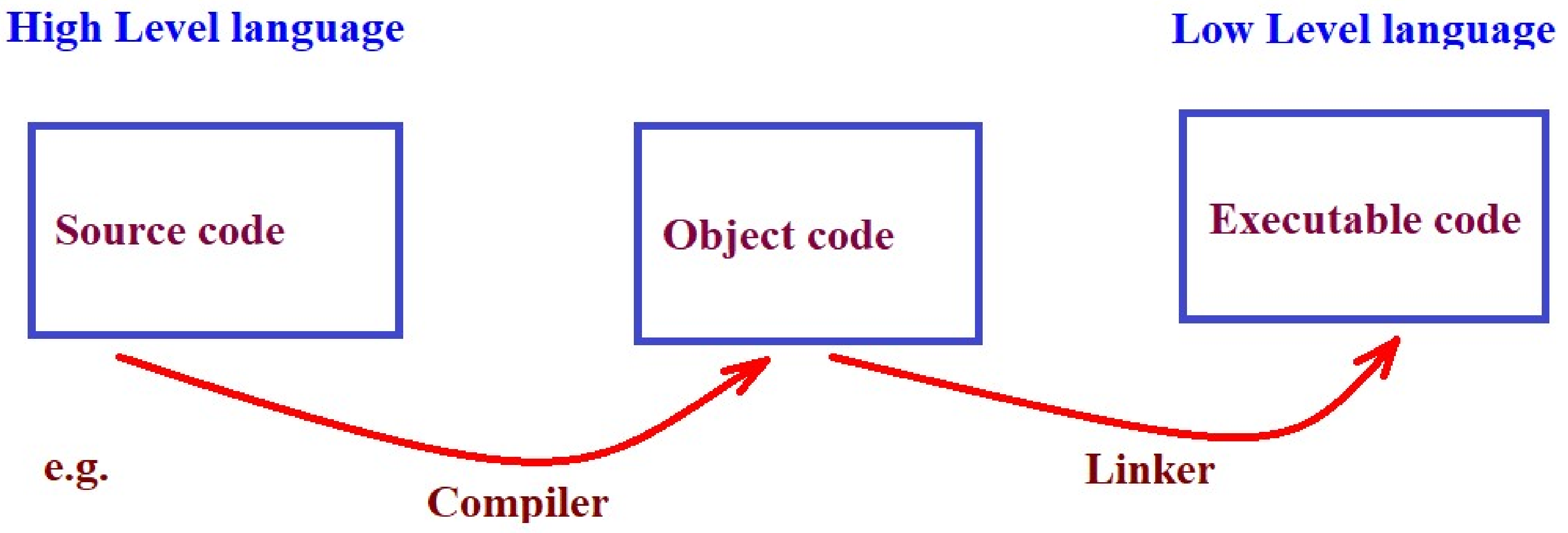
Trends of Programming and Programming languages

- For better expression of operations, assembly language was developed
- Programs in assemble language are translated into machine language using an assembler
- Assembly language is still a low-level language and not fit for large programs
- Assemble language is not portable

Trends of Programming and Programming languages

- High level language which is English-like were developed
- It uses control structures which made program flow to be easily expressed
- It is portable unlike assembly language
- It needs compiler to first translate to machine-specific assembly language, and then an assembler to convert it to machine language which is executable by the machine.
- Examples of high-level language is Fortran, Algol, Ada etc.

Trends of Programming and Programming languages



```
if (x >50) {
   printf("It's filled-up!");
}
```

Trends of Programming and Programming languages

- Another alternative to compiled code or compilation process is the use of interpreter.
- An interpreter directly converts an high level language into low-level language which is an executable code.
- Another alternative is a virtual machine e.g. Java virtual machine, which works on top of the real machine.

Trends of Programming and Programming languages

- Virtual machine converts high-level language into an intermediate code called **byte code**.
- Byte code can be easily interpreted into executable code than high-level language.
- Compiled code is the fastest, interpreted code is portable and quick to implements and test, virtual machine has the advantage both speed and portability.

PROBLEM SOLVING PROCESS

- Problem solving is an act of finding a solution to a perplexing question.
- problem solving is a natural way of providing solution to a problem

Problem Solving Process

- There are **different ways** of solving a problem, e.g. use of **well defined mathematical principles** will help to solve a mathematical problem, using of **tractor on hectares of land** will help to plough the land etc.
- In Computing and Software Engineering, we are interested in solving problems through the use of computers.
- Thus we ask questions that can be **solved using computer**.
- These are questions that do not involve physical activity or emotions this is because a computer only acts on instructions given to it, thus it is not intelligent i.e. garbage in garbage out.

Problem Solving Process

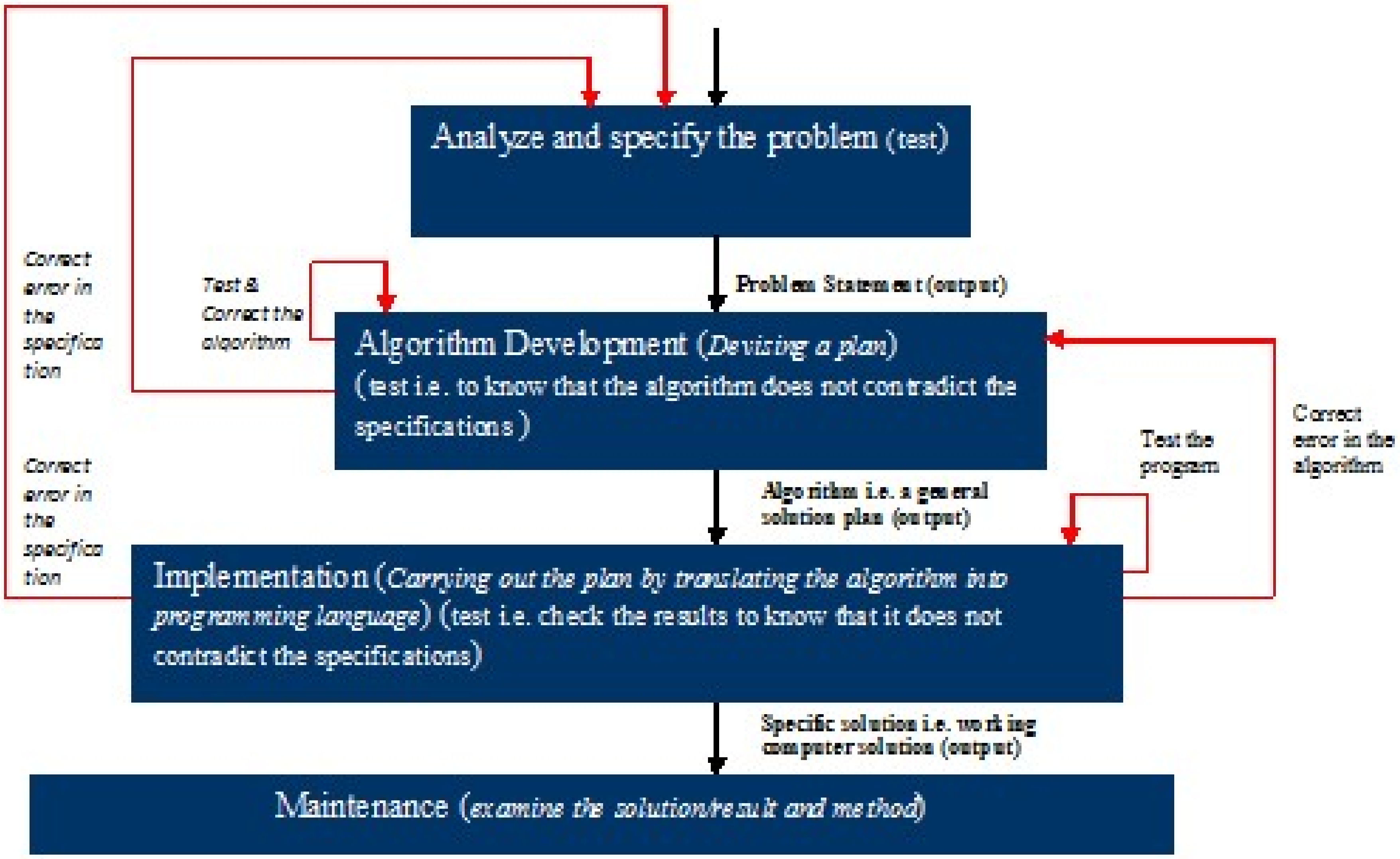
- Advantage of using computer to solve a problem is that once the instruction is written for the computer, the computer can repeat the solution very quickly and consistently, again and again, for different situations or data.
- This means, a computer program for computation of CGPA of students can do it accurately for as many students as possible.

Steps in Problem Solving Process

- Understanding and analysis of the problem
- Development of algorithm i.e. devising a plan
- Coding and implementation
- Deployment and Maintenance

Note: At each stage there should be testing and proper documentation.

Diagrammatic representation of Problem Solving Process



Analyze and specify the problem (test)
Problem Statement (output)
Algorithm Development (Devising a plan) (test i.e. to know that the algorithm does not contradict the specifications)
Algorithm i.e. a general solution plan (output)
Implementation (Carrying out the plan by translating the algorithm into programming language) (test i.e. check the results to know that it does not contradict the specifications)
Specific solution i.e. working computer solution (output)
Maintenance (examine the solution/result and method)
Correct error in the specification
Test & Correct the algorithm
Correct error in the specification
Test the program
Correct error in the algorithm

Understanding and analysis of the problem

* To have a proper understanding of the problem, some analytical questions must be asked and answered, for example

What **input data/information** is available ?

What does it **represent** ?

What **format** is it in ?

Is anything **missing** ?

Do I have everything that I need ?

What **output information** am I trying to produce ?

What do I want the result to look like i.e. **result format** e.g. text a picture, a graph … ?

What am I going to have to **compute** ?

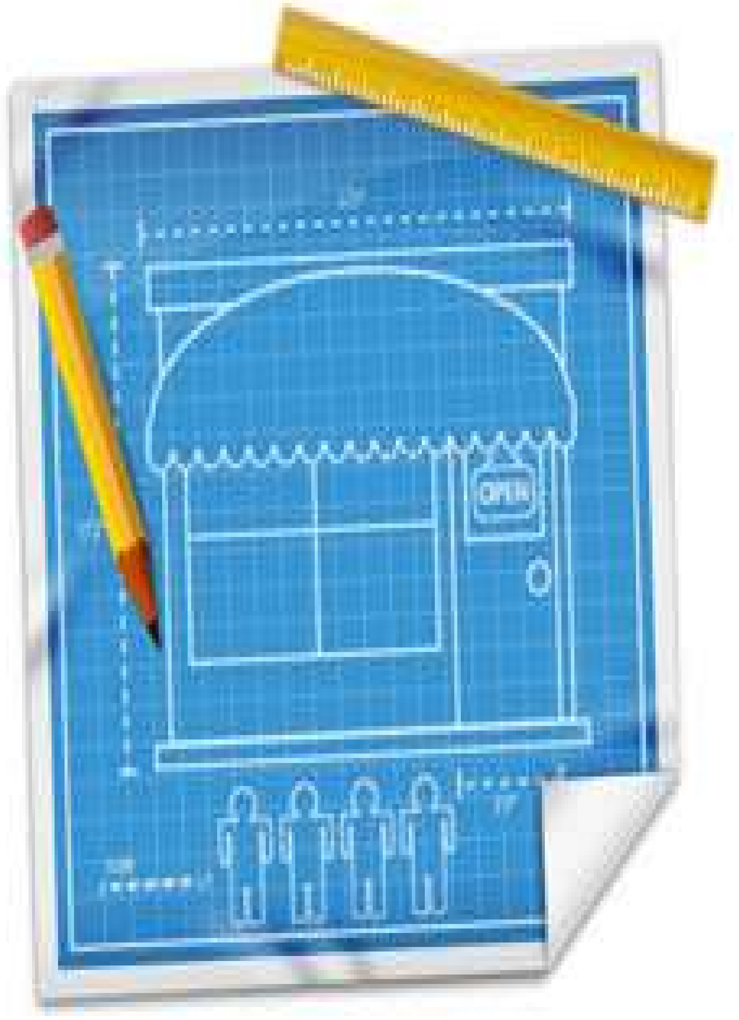
Understanding and analysis of the problem

These questions will help in:

- Identifying the **input data**
- Identifying the **output requirement** (results)
- Describe the **processing requirements**
- Determine whether **computer is require to solve the problem**

Understanding and analysis of the problem (Problem decomposition)

* Knowing the input, process and output requirement will help to break the problem down into **smaller sub problems**.
* Each sub problem has data that can be computed in order to process the data.



Lab I: Understanding and Analyzing the Problem (Example 1)

- Note that the first thing is the problem statement, for example if we have a problem statement as given below:
- **Example 1**
- Calculate the **average grade** for all students in a class.

Lab I: Understanding and Analyzing the Problem (Example 1)

a. Determine the **input data**:

What input data/information is available? (score, total number of students)

What format is it in ? (integer? float? string? video? picture?)

Is anything missing ? (is there any missing score?)

b. Determine the **processing requirement.**

Do I have everything that I need ? (sum, total number of students)

c. What **output information** am I trying to produce ?

What do I want the result to look like … text

a picture, a graph … ?

Relating step 1 to identifying variables

* Based on input data, output data and processing requirements we identify **VARIABLES** which will be subsequently used in programming.
* Variables
    * Named memory location
    * Their value can change during the execution of a program
    * In ANSI C programs, all variables must be declared prior to their use. This property makes C to be a **strongly typed language**.

Data format

* Data format represents **data types**

C has a small family of data types.

* Numeric (int, float, double)
* Character (char)
* User defined (struct,union)

Lab I

- Identify (stage 1):
- Output requirement and respective formats
- Input requirement and respective formats
- Processing requirement
- Stage 2:
- Identify the sub-problems

Lab I: Understanding and Analyzing the Problem (Example 1)

- **Output requirement:**
- Expected Result: Average grade
- Format: floating-point or double
- **Input data:**
- Total number of students (format: integer)
- Score per student (format: integer)
- Total score: (format: integer)
- **Processing requirement:**
- Average grade= total score/total number of student
- Format: digit

Problem Decomposotion (Example 1)

- In the problem at hand there are data like scores, sum of scores, and number of student in the class. These data are used in calculating average grade.
- Thus the sub problems are:
- **Total score**: problem of adding all the students scores together.
- **Total number of students**: problem of counting the number of all students.
- **Average grade**: problem of finding the average score per student.
- Average=sum of scores/total no of students
- If no mathematical expression is available you can formulate one if necessary.

Lab I: Understanding and Analyzing the Problem (Example 2)

- **Example 2:**

  The personnel office of Babcock University is assigned the task of computing the monthly emolument of all category of employees of the institution

  The employee taxation is base on the following constraints;

Lab I: Understanding and Analyzing the Problem (Example 2)

- If basic salary is less than or equal to ₦5000, 5% of basic salary is deducted as tax otherwise 8% deducted.
- Also 10% of the employee gross pay should be deducted as part of pension scheme.
- It is require that each employee pay slip should contain employee number, name, total allowance, tax, basic salary, gross pay and Net pay

Lab I: Understanding and Analyzing the Problem (Example 2)

- Analyse the above problem and identify the output, input and processing requirements
- Draw the flowchart for the payroll system

Lab I: Understanding and Analyzing the Problem (Example 2)

- **Input data:**
  - Employee Number
  - Employee Name
  - Basic salary
  - Housing allowance
  - Transport allowance
  - Meal subsidy

A class teacher needs to compute the average score of 20 pupils in eight (8) subjects. Thus, before developing the application you need to understand and analyse the problem clearly stating the input data, output data, and processing requirements. Also, indicate variable name and data type of each variable.

Lab 19/9

- Babbie.com, a game website request for submission of game from developers. The game is to teach kindergarten on how to detect their year of birth through their current age and current year. Before developing the game application, you are to analyse the problem indicating input data, output data, and processing requirements.

Solution:

Output requirements:

- Employee Number
- Employee Name
- Total allowance
- Tax
- Contributory Pension
- Basic salary
- Gross pay
- Net pay

Lab I: Understanding and Analyzing the Problem (Example 2)

- **Processing requirements:**
  - If Basic salary is <=5000, Tax = 5% of Basic salary

  Else Tax = 8% of basic salary.
  - Total allowances = Housing + Transport + meal subsidy
  - Gross Pay = Basic salary + Total allowances
  - Contributory pension= 10% of Gross Pay
  - Net pay = Gross pay – (Tax + contributory pension)

Classwork: Problem Decomposotion (classwork)

- Identify the sub problems in Example 2
- The volume of a cylinder is given by
  - $V = \pi r^2 h$
  - Analyse the above problem and identify:
  - the input, output and processing requirements
  - And the sub problems

- The area of a circle is given by the formula

  Area = πr^2

  Analyse the above and identify the output, input and processing requirements. And identify the sub problems.

- All programming languages have their uniqueness in terms of
- Syntax (Grammatical rules of constructing a statement)
- Semantics (Intended interpretation of a statement, keywords, or reserved words)