ELEMENTARY PROGRAMMING PRINCIPLES.

Computer Programming
A program – is an organized list of statements (instructions) that when executed, cause the computer to behave in a pre-determined manner or carry out a defined task.

Programming – Refers to the process of developing computer (instructions) programs used to solve a particular task.
A computer program is designed using a particular programming language. Each language has a special sequence or order of writing characters usually referred to as **syntax**

Terms used in programming
Source program.
This refers to the program code that the programmer enters in the program editor window that is not yet translated into machine-readable form.

Object code
This refers to the program code that is in machine –readable i.e a source code that has been translated into machine language.

Translators
These are programming tools that translates /convert the source program into object code. E.g. Assemblers, compilers, interpreters etc.

Assembler
An assembler translates a program written in assembly language into machine language.

Interpreter
This translates the source programs line-by-line, allowing the CPU to execute one line before translating the next. The translated line is not stored in the computer memory, hence every time the program is executed, it has to be translated.

Compiler
This translates the entire source program into object code. The compiler translates each high level instruction into several machine code instructions in a process called COMPILATION and produces a complete independent program that can be run by the computer as often as required without the original source program being present.

Levels of programming languages
There are two major levels namely; -
i) Low level languages
ii) High- level languages.

1. Low-level languages.
- These languages are classified as low because they can be directly, or easily understood by the computer with little effort to translate into computer understandable form.
- These languages are hardware oriented and therefore they are not portable. I.e. a program written for one computer cannot be installed and used on other.

Types of low level languages
A . Machine language: First generation (languages)

- In this language, instructions are written using binary logic. Given that data and instructions are in binary form, many lines of codes are needed to accomplish even a simple task like adding two numbers i.e. program written in this language look like this.

```
111000110     0000011       10000001
0001111       10001101
10001111      1111111       1000011
```

The program code is hard for humans to understand but it's easily understood by computers.

B. Assembly languages (second generation languages)

- This language is close to the machines vocabulary rather than the human beings vocabulary. It was developed in order to overcome the difficulties of understanding and using machine language. This language helps the programmers to write programs as a set of symbolic operation codes called **mnemonics**. Mnemonics are basically shortened two or three letter words. A sample program written in Assembly language.

```
Mov    AX, 15        (move 15 to register AX)
SUB    Ax, 10        (subtract 10 from the value Ax.
```

Programs written in this language require an assembler to convert them into machine language.

2. High-level languages.

These languages are very close to the human language (English –like) and they can be read and understood even by people who are not experts in programming. These languages are machine independent. This means that a programmer concentrates on problem solving during a programming session rather than how a machine operates.

Classes of high-level languages

i) Third generation languages (3 GLS)

This generation language is also called structured or procedural languages. A procedural language makes it possible to break a program into components called modules. Each performing a particular task. Structured programming has advantages because it's flexible, easier to read and modify.

Examples of third generation programming language.
Pascal – Was developed to help in teaching and learning of structured programming.
Fotran – Was developed for mathematics , scientists and engineers. It enables writing of programs with mathematical expressions.
Cobol – Was designed for developing programs that solve business programs.
Basic – Developed to enable students to easily learn programming.
c- Used for developing system software e.g the operating systems. Its very powerful high level language because of its ability to provide programmer with powerful aspects / features of low level.
Ada – This language is situatable for developing military, industrial and real time systems.

ii) Forth generation languages (4 GLs)

This generation make programming an even easier task than the third generation language because they present the programmer with more programming tools.

Examples of such tools are command buttons, forms etc. the 4 GLs are easy to learn and understand because they are user based. The languages syntax (grammar) is natural , near English language and use menus to prompts to guide a non-specilist or retrieve data with ease.

Examples of 4GLs

a) Visual Basic
b) Delphi Pascal
c) Visual cobol
d) C + +

iii) Fifth generation languages (5 G's)

These languages are designed around the concept of solving problems by enabling the computer to depict human like intelligence. These programs are designed to make the computer solve the problem programmer rather than programmer spending a lot of time to come up with the solution.

Examples of 5GL's

a) PROLOG
b) MERCURY
c) LISP
d) OCCAM.

iv) Object oriented programming languages. (OOP)

The concept behind OOP languages is to look at a program as having various objects instructing to make up a whole. Each object has a specific data values that are unique to it (called state) and a set of the things it can accomplish called (functions or behavior). This process of having data and functions that operate on the data within an object is called **Encapsulation**. Several objects can then be linked together to form a complete program OOP has greatly contributed to development of **Graphical user interface operating systems and application programs.**

Examples of OOP

a) Java
b) Simula
c) Small talk.

v) Web scripting languages.

These languages are used to develop or add functionalities on web pages. Web pages are hypertext documents created in a language called **Hypertext markup languages** (HTML) .the language consists of markup tags that tell the internet browser that the file contains HTML- code information and is distinguished by a file extension of HTML . the markup tags define the various components of a world wide Web document such as heading , tables , paragraphs , lists etc. HTMl does not have the declaration part and control structures , hence its not considered as a true programming language. Due to its simplist, it has many limitations and can not be used alone when developing functional websutes. Some special blocks of codes called Scripts may be inserted in HTML pages using scripting languages like JavaScript, VBScript etc in order to add functionality to HTMl PAGES.

Advantages of Low-level languages.

1. The CPU understands machine language directly without translation.

2. They are suitable and hardly crash or breakdown once written
3. Running a program is fast, no compilation is needed.
4. They are economical in terms of the amount of memory they use.

Disadvantages

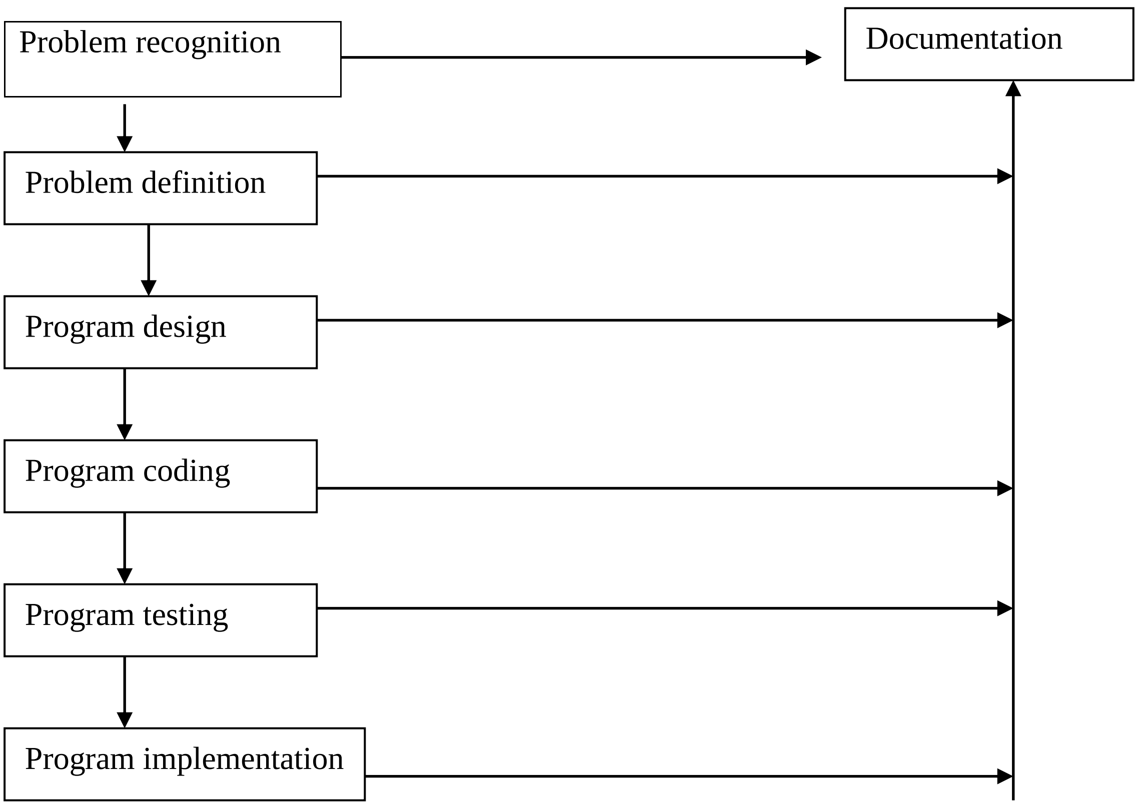
1. They are difficult and cumbersome to use and learn
2. Requires highly trained experts to both develop and maintain programs.
3. Debugging programs is difficult
1. They are machine dependant
2. the programs are long.

Advantages of high-level languages
1. The programs are portable (not machine dependant)
2. They are user friendly and easy to use and learn.
3. They are more flexible
1. They provide better documentation
2. They are easy to debug
3. Require less time to code.

Disadvantages
i) Program executed more slowly.
ii) Require larger CPU storage capacity for compilation.
iii) They have to be translated to machine-readable form before the computer can execute them.

Program development

There are six stages of program development. They include



i) Program recognition

This refers to the understanding and interpretation of a particular problem. To understand a problem one has to look for key words such as computer, evaluate, compare etc. a programmer identifies problems in the environment and seeks to solve them by writing computer program that would provide the solution.

Circumstances that can cause the programmer to identify a problem.

1. Opportunity to improve the current program
2. A new directive given by the management requiring a change in the status quo.
3. Problems of undesirable solutions that prevent an individual or organization from achieving their purpose.

ii) Problem definition.

At this stage the programmer tries to determine or define the likely input, processing and expected output using the key words outlined at the problem recognition stage. The boundaries of the expected program are established and if several methods to solve the same problem are identified the best alternatives should be chosen. At the end of the stage requirement documentation for the new program is written.

iii) Problem design

This is the actual development of the program's processing or problem solving logic called algorithm. (A limited number of logical steps that a program follows in order to solve a problem) the programmer comes up with an algorithm after analyzing the requirements specifications.

Some of the problems are made up of large block code. Ie they are Monolithic while others are made of several units called modules, which work together to form the whole program.

In modular programming each module performs a specific task. This approach makes a program flexible, easier to read and debug. This phase enable the programmer to come up with models of the expected program. The model shows the flow of events and data throughout the entire program from input of a program.

iv) Program coding

This is the actual process of converting a design model into its equivalent program. This is done by creating the program using a particular programming language. The end result of this stage is source programs that can be translated into machine-readable form for the computer to execute and solve the target problem.

v) Program Testing and Debugging.

After coding the program has to be tested and the errors detected and corrected. Debugging refers to detection and correction of errors that may exist in the program. Program testing involves creating test data designed to produce predictable output.

There are two types of errors (bugs) that can be encountered when testing

i) **Syntax errors** – They occur as a result of improper use of language rules. e.g. grammar mistakes , punctuation , improper naming of the variables etc. These errors are detectable by translator and must be corrected before the program runs.

ii) **Logical errors-** They are not detectable by the translator. The program rules but gives wrong output or halts during execution.

Methods of error detection

i) Desk checking / dry-run

It involves going through the program while still on a paper before entering it in the program editor.

ii) Using Debugging Utilities.

In the program editor, you can run the debugging utilities during translation to detect syntax errors.

iii) Using Test Data

The programmer carries out trial runs of the new program .At each run he enters various data variation and extremes including data with errors to test whether the system will grid to a halt. A good program should not crash due to incorrect data entry but should inform the user about the anomaly.

vi) Implementation and maintenance Implementation.

This is the actual delivery and installation of the new program ready for use, creating data files and train people to use the system. The new system will change the way things are done hence it should be reviewed and maintained.

vii) Review and maintenance

This stage is important because of the errors that may be encountered after implementation. A program may fail due to poor use, hence proper training and post implementation support of users will reduce chances of having them entering invalid data that crash the program.

viii) Program documentation

This is writing of support materials explaining how the program can be used by users, installed by operators or modified by other programmers. All stages of development should be documented in order to help during future modification of the program.

Types of documentation

i) User oriented documentation

These type enables the user to learn how to use the program as quickly as possible and with little help from grammar.

ii) Operator oriented documentation

Meant for computer operators. E.g Technician. it helps them to install and maintain the program.

Development of algorithms.

Algorithms – Refers to a limited number of logical steps that a program follows in order to solve a problem.

Pseudo code – Refers to a set of statements written in a readable language (English – like) but expressing the processing logic of program.

Guidelines for designing a good pseudo code.

1. The statements must be short, clear and readable.
2. Pseudo code lines should be clearly outlined and indented clearly.
3. It should show clearly the start and stop of executable statements and control structures.
4. Statements must not have more than one meaning.

Examples of Pseudo code.

Write a pseudo code that can be used to prompt the user to enter two numbers, calculate the sum and average of the two numbers and then display the output on the screen.

<u>Solution</u>

```
START
Print “Enter two numbers”
Input x,y
Sum – x+y
Average = sum /2
```

PRINT sum
PRINT Average
STOP

Program flowcharts.

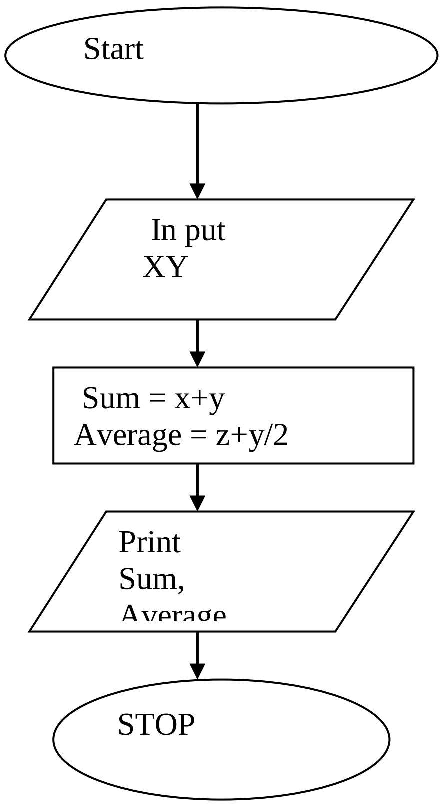
A flowchart is a diagrammatic representation of a program in algorithms. It uses statements and symbols that have specific meaning. The symbols are of different standard shapes linked to show the order of processing. Each shape contains note stating what the operation is.

Guidelines for drawing a flowchart

1. There should be only one entry and one exit point of a program algorithm.
2. Use correct symbol at each stage in the flowchart.
3. Avoid a cross flow lines.
4. Be as neat and tidy in drawing as possible.
5. General direction of flow in any flowchart is from top to bottom, left to right.

Examples of flowchart

Draw a flowchart for a program used to prompt the user to enter two number s. the program should find the sum, and average then display the output



Types of flowchart

System flowchart

It’s a chart that depicts the systems as a whole with only subsystems or a major elements shown.

Program flowchart

This chart shows the sequence of operations as carried out by a computer program.

1.

Program control structure
They are blocks of statements that determine how statements are to be executed.
There are 3 control structures namely.
i) Sequence
In this control structure, the computer reads instructions from a program file starting from the first top line and proceeding downwards one by one to the end. Hence sequential program execution enables the computer to perform tasks that are arranged consecutively one after another in the code.

Examples of how a sequential program execute

```
Begin {procedure name}        the program file reader reads sequentially
statements
        Action 1              by statements to the end of the file.
        Action 2
        Action n
End     {procedure name}
```

ii) Selection/decision
This structure is used to branch, depending on whether the condition returns a value of True or False (yes or no)

For example

```
If < condition >
        Then Action 1
        Else Action 2
Endif.
```

There are 4 types of selection controls used in high-level programming
1. IF…………………………..THEN
This is used if only one option is available. All other options are ignored. For example if a school wants to reward only the students who have mean mark of 80 and above, it will be reward only those students who have attained 80% and above and ignore the rest.

General format

```
If < condition > Then
Statements :
Endif
```

```
If mark >80 then
Print "Reward"
Endif
```

IF……………………………….THEN………..ELSE
Used when there are two available options for example in a football match a player is given a RED CARD if he does a very serious mistake otherwise he is given Yellow Card.

General Format

```
If < condition >THEN
```

```
        Statements 1,
ELSE
        Statement 2
EndIF
```

The algorithm will be;-

```
If fault = serious THEN
        Print "RED CARD"
ELSE
        Print " yellow card"
EndIf.
```

3. Nested IF

This is used where two or more options have to be considered to make a selection. For example, to award grade according to the marks as follows

a)	80 marks	Grade A
b)	60 marks	Grade B
c)	50 marks	Grade C
d)	40 marks	Grade D

General format

```
If < conditions >Then
        Statement
ELSE
If < condition >Then
        Statement
ELSE
If < condition >Then
        Statement
ELSE
        Statement
        EndIf
        EndIf
        End
```

CASE SELECTION

Its an alternative to the nested IF. This selection is preffered to the Nested if in order to reduce the many lines of codes . case selection can only be expressed using integers and alphabetic characters only. The Boolean expression should be CASE interger OF or CASE char OF.

General format

```
CASE x of
        Label 1: statement 1
        Label 2: statement 2
        :
        label n: statement n-1
        ELSE
        Statement n
        End case
```

Example

```
        CASE            average OF
        80…100:         Grade = "A"
```

```
    70-79        Grade= “B”
    60-69        Grade = “C”
    40-49        Grade =” E”
    ELSE
                 Grade = “F”
    End case.
```

iii) Iteration (coping) repetition

This is designed to execute the same block of code again and again until a certain condition is fulfilled . Itelaration is important in situations where the same operation has to be carried out on a set of data many times.

There are three main looping controls.

.

i) THE – WHILE-DO LOOP

This repetitive structure tests the condition first before executing the successful code if and only if the Boolean expression returns a true value.

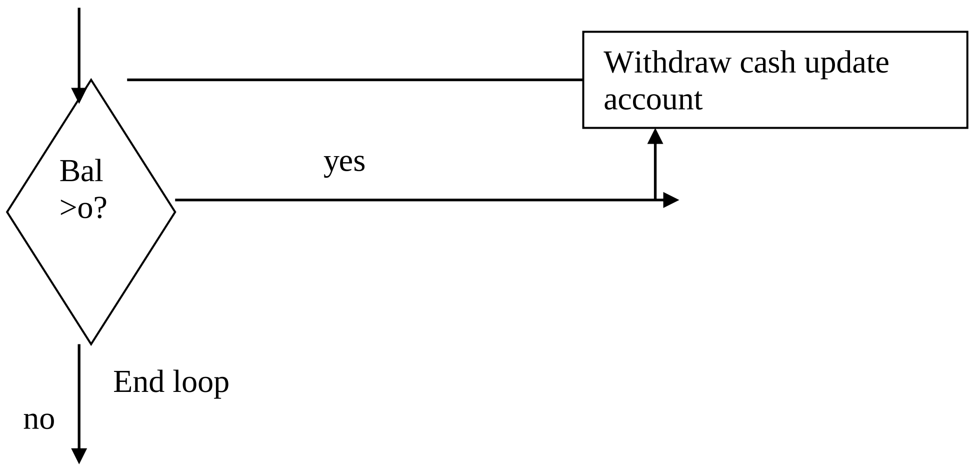
Example

To withdraw money using an ATM a customer must have a balance in his/her account.

General format

```
While <condition >Do
    Statement
    End while.
```

Flowchart



Pseudo code segment

```
While  balance >O do
    Withdraw cash
    Update account
End while
```

Example

Pseudo code segment

```
REPEAT
Withdraw cash
Update account
Until balance ≤0:
```

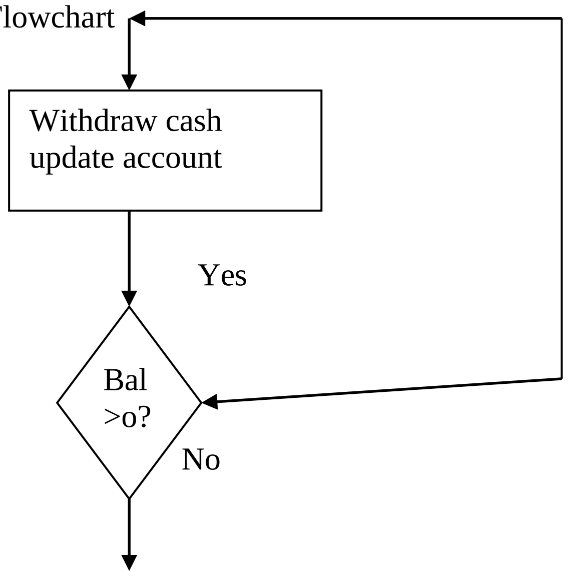
General format

```
REPEAT
Statement
UNTIL <condition>
```

iii) REPEAT……………..UNTIL LOOP

In this structure the code is executed before testing the condition. The repeat loop stops when the Boolean statement returns a value. For example in the case of ATM of discussed above the client can withdraw money until balance is zero.

Flowchart



Exit loop

iii) The FOR LOOP

This structure is used where execution of the chosen statements has to be repeated a predetermined number of times. For example if a program is to calculate the sum of ten numbers provided by the user. The FOR LOOP can be used to prompt the user to enter the 10 numbers at most ten times. Once the numbers are entered the program calculates and displays the sum.

Pseudo code

```
FOR count = 1 to 10 Do
Writeln "Enter a number (N)"
Readln N
Sum = sum +N
End FOR.

Display sum
```

The counter has to be set to a start value and sometimes to an end value.

General format of the loop

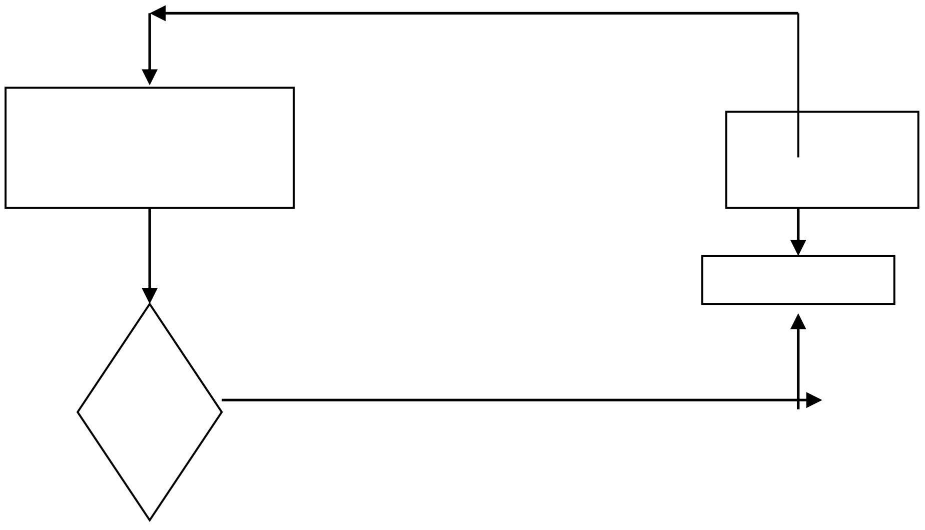
1. Format for the FOR loop that counts from lower limit.

```
For loop variables = lower limit To upper limit Do
        Statements
        Endfor
```

2. Fomart for the "for" loop that counts from upper limit down to lower limit .

```
for loop variable = Upper limit Down To lower Limit Do
        Statements
        Endfor
```

Flowchart for a forloop that counts upwards.



Flowchart for a FORLOOP that counts downwards

Diagram

Examples of complex pseudo codes.

1. Unshirika society pays 5% interest on shares exceeding 10,000 Ksh and 3% on share that do not meet theis target. However no interests is paid on deposits in the members bank. Account . Design a pseudo code for a program that would:

a) Prompt the user for shares and deposits of a particular member.

b) Calculate the interest and total savings.

c) Display the interest and total savings on the screen for a particular member.

Pseudo code

```
Start
Input Name, share,Deposit
If share> 10000 THEN
Intrest = 0.05 x shares
ELSE
Interest = 0.03 x shares
EndIf
Total savings = Deposit Interest + shares
Print Name., Totalsaving , Interest
Stop
```

Flowchart

Diagram

2. Botswana has a population of 3,000,000 but thi si filling by 4% each year . The island of Reunion has a population of 95,000 but this is increasing by 9% each year.

a) Draw a flowchart that predicts the year in which the population of reunion will be greater than that of Botswana if the trend continues.

Solution

Pseudo code

```
Start
Bts: 3,000
IR: = 95,000
Year : = 0
REPEAT
Bts = Bts – (Bts * (4/100)
IR = IR + (IR (9/100))
Year = year +1
UNTIL IR > Bts
Print year
Stop
```

3. Write a program that will allow the input of name of student marks obtained in 5 subjects (Math, English, Computer, Biology) .
   The program should calculate the total and average marks for each student and assign the grades depending on the average marks obtained as follows.

80- 100	A
70- 79	B
60- 69	C
50-59	D

Below 50 –E
The program should then display each students name , total marks and average .

Pseudo code

```
START
REPEAT
Print " Enter name and subject marks"
Input Name, maths,English , Kiswahil , Computer , Biology
Sum = maths,English , Kiswahil , Computer , Biology
AVG = sum/5
If (AVG ≥ 80) AND (AVG ≤ 100) THEN
Grade = "A"
If (AVG ≥ 70) AND (AVG ≤79) THEN
Grade = "B"
If (AVG ≥60) AND (AVG ≤69 THEN
Grade = "c"
If (AVG ≥ 50) AND (AVG ≤59) THEN
Grade = "D"
ELSE
Grade = "E"
Endif
Endif
Endif
Endif
Print name, sum, AVG, Grade
Until count = Number of students.
Stop
```

Flowchart

PAST KCSE QUESTIONS ON THE TOPIC

1. 2002
   State two types of documentation in program development and give the purpose of each . (4 marks)
2. state any three activities that occur in a program compilation process (3 marks)
3. The following can be used to list the add numbers between 0 and 100

1. Diagram

a) write a program segment for the flowchart using a high language (7 marks)

b) What would be the output from the flowchart if the statement in the decision box is changed to (3 marks)

i) odd = 100 ii) odd <100 iii) odd >100

2003

1 a) Distinguish between Machine and Assembly language (2 marks)
 b) State the type of translator necessary for a program written in
 i) High level language
 ii) Assembly language

2. Briefly explain the purpose of the following types of a program documentation (2marks)
 i) User manual
 ii) Reference guide

3. State any two features of user-friendly program (2 marks)

2. Study the flowchart below and the question that follow.

a) Write a high level language program for the above flowchart (7 marks)

b) List the outputs of the flow chart above (5 marks)

KCSE 2004

1. Distinguish between a compiler and an interpreter (2 marks)

2. What is meant by the term DRY Running as used in program development (2 marks)

3. Differentiate between source program and object program (2 marks)

4. Bidii wholesaler has two categories of customers for order processing. Category "A" obtains 10% discount on all orders up to Ksh 10,000. Otherwise the discount is 20% on the entire order. Category "B" obtains 30% discount on all orders if the debt repayment is "good" otherwise the discount is 15% . Draw a flowchart for the order processing (15 marks)

KCSE 2005

1. Distinguish between Real, Integer and character data types a used in programming (3 marks)

Diagram

2. a) Name the control structure depicted by the flowchart above (1 mark)
 b) Explain the following terms as used in program implementation (2 marks)
 i) Parallel running
 ii) Direct changeover

3. a) State the stages of program development in which (2 marks)
 i) A flowchart would be drawn
 ii) The program would check whether the program does as required.
 iii) The user guide would be written
 iv) The requirements specification would be written
 b) State the output of the following segment.
 Diagram
c) Draw a flowchart to computer the combined resistance ® of two resitors R1 and R2 in parallel using the formula: (5 marks)

$$R = \frac{1}{\frac{1}{R1} + \frac{1}{R2}}$$

KCSE 2006

1. a) List two examples of
    i) Third generation language
    ii) Object oriented languages.

2 . 2007
Write al algorithm to compute the area of a triangle (2 marks)

PRACTICE QUESTIONS ON THE TOPIC

1. Distinguish between the following
    a) Compiler and interpreter.
    b) Object code and source code.
2. State 3 advantages of high level languages over low level language.
3. Outline the stages of program development in their respective order
2. State two advantages of modula programming.
3. distinguish between pseudo code and Algorithm.
4. explain three types of control structures used in programming
5. write a pseudo code that will inform the user of what to waer depending on the weather . if its raining “wear rain coat” if not “wear overcoat”
6. draw a flowchart for a program to display the name of a suspect to a crime who is aged between 20 and 35 year and between 66 and 70 inches tall.
7. draw a flowchart to compute and print the grades for an examination. The input Data is Roll. No and marks for six subjects out of 100 . grades are allocated on the following basis

% marks
grades

75 and above	A
60 and less than 75	B
Less than 60	C

PREDICTION QUESTIONS ON THE TOPIC

1. a) What is meant by structured programming (1 mark)
    b) State 3 advantages of using modules in program development (2 marks)
2. Give a reason why its necessary to have a program design (1 mark)
2. Distinguish between user documentation and operator documentation (2 marks)
3. state two advantages and two disadvantages of using flowchart in program design (4 marks)
4. using a simple sketch , illustrate the
    i) REPEAT……….UNTIL control structure (3 marks)
    ii) WHILE ………..DO control structure
    iii) CASE control structure
5. Give an advantage of compiling a program rather than interpreting it (1 mark)

CHAPTER FOUR

SYSTEM DEVELOPMENT

A system

Is a set of items linked together to carry out a given task in order to achieve one or more objectives . a system is described as being either soft or hard.

i) Soft system

Human activity are said to be soft systems. They are said to be soft because:

a) Their boundaries may be fluid or keep on changing

b) Their goals and objectives conflict.

c) Its difficult to precise define exact measures of performance for them.

ii) Hard systems

These are systems whose goals and objectives are clearly defined and the outcomes from the systems processes are predictable and can be modeled accurately.

Systems classification

i) Deterministic system

These are systems whose outputs are known precisely by their input e.g a computer.

ii) Probalistic systems

These are systems whose output can only be predicted but not known precisely. e.g Business and economic system organization

iii) Cybernetic systems.

These are self-controlling systems, which adapt to the environment and regulate their own behavior by accessing the feedback. They access their output and adjust the input e.g. Human beings, plants systems.

Characteristic of systems

1. Holistic thinking

In this, a system is considered as a whole. A combination of various components that makes up a system creates a complex whole whose overall goals are more sophisticated than those of the individual components.

2. Sub systems.

A system is made up of different components (subsystem) . Therefore a system does not exist in solitude but it may be a component of large system.

3. Processor

This is an element of a system that does the actual transformation of input into output.

4. Objectives / purposes

Objectives of the system may be real or stated. An organisation should state objective and operate to achieve another user must know the major objective of a system.

5. Environment

The environment is the system within an organization operates. Environment provides a reason for the existence of the system.

6. Boundaries

System boundaries are external element whose changes in attitudes, behavior and property affect their state and are affected respectively. Ie it’s the limit of system.

7. Independence

For a system to be effective and efficient, its components or subsystems must be co-ordinate and linked together.

8. Feedback

A feedback is a check within a system to see whether predetermined goals are being met. The checks are conducted on a system to determine deviation.

9. Control

This is the process by which the system adapts to changes in the environment in order to give the expected level.

10. System Entropy

Entropy means decay. System decay naturally over time, hence it slowly becomes useless to the user either due to improvement in technology, new management or change in user requirements. Therefore it must be reviewed in order to improve it or develop a new one.

11. Inputs and out puts

A system communicates with its environment by receiving inputs and giving outputs.

12. Open and closed system

An open system receives input from and gives output to the environment while a closed system does not interact with the environment so that good quality reports are produced for easy understandings.

b) Input specification.

The input to the system is necessary because the content of their input are used to maintain the master files. The following should be considered in input specification .

i) The contents and volume of input
ii) The mode and devices of input selected.
iii) Layout and sequence of input.

c) File / data stores

File requirement involves making an informed decision on file required to store data and information in the system. The analyst should identify the number of files that will be needed by the system and determine the structure of each file.

d) Hardware and software requirements.

System analyst should specify all hardware and software requirements for the new system. He/she should consider; -

a) Economic factors e.g price
b) Operational factors e.g reliability
c) User friendliness.

4. System design

This involves detailing the physical design of the system, it's the how of the process. Once the logical design is outlined during the system analysis , the analyst determines the physical design, which describes the hardware, software and operating procedures required to make a system operational. Several tools are used for designing an information system. . Examples are flowchart, data flow diagram, structured charts. Etc.

5. System construction.

This refers to coding, installation and testing of the molecules and their components such as outputs, inputs and files. The purpose of the construction is to develop and test a functional system that fulfils the business and design requirements.

Testing the system

The system is tested be entering some test data to find out whether its output are as expected. The system is tested using the requirements specifications and the design specification to find out whether it meets all requirements specified.

6. System implementation

This involves installing the system in the user’s computers. Testing the installed system, converting from old system to the new one an d training the users.
System implementation involves;-
i) File creation and conversion
ii) Chang eover
iii) Staff training.

i) File creation and conversion
This involves setting up of the master files that are to be used in the support of the new system. The setting can be either from scratch or just to convert the files that were employed in old system.
ii) Training staff .
The training aim at ;-
a) Convincing the staff of the user department on the effectiveness of the new system.
b) Remove fear of change from them.
c) Enabling the staff to cope with processing task of the new system.

Methods of Training
- ✓ Demonstration
- ✓ Manual reference
- ✓ Group discussion
- ✓ Visits

iii) Changeover
This is the process of changing over from old system to rthe new system.

Types of changeover
a) Direct changeover
In this approach the new system commence live operation and immediately the old system is abandoned.

Advantages of Direct changeover
- ✓ It’s a cheap method

Disadvantages of Direct changeover
- ✓ Extremely risky
- ✓ Places an organization in a do or die situation.

b) Parallel changeover
In this changeover the operation of the old system and the new system run alongside each other.
Advantages
- ✓ Provides a method of training in the new system
- ✓ Personnel become familiar with the new system prior to actual changeover.

Disadvantages
- ✓ Costly to run two systems
- ✓ Cross checking is difficult
- ✓ Requires more staff hence more training required for additional staff.

c) Phase changeover.
In this approach implementation is done only one part of the new system at one time or a step by step.
Advantages
- ✓ Allow the system to be implemented quickly
- ✓ Less costly

- ✓ Failure of the system are limited
- ✓ Causes minimal disruption.

Disadvantages

- ✓ Problem on ensuring that the first phase is implemented and converted.

7. System review and maintenance.
   System maintenance is the adjustment and enhancement or correction or errors after the system has been implemented. The reviewing involves going through the specification and testing the system after implementation to find out whether it still meets the original specification.

8. System documentation
   This is written graphical record of the steps taken during the system development process.

System documentation consists of

i) Report of fact-finding which outlines methods used to collect data, weakness of the current system and recommendation of the current system.

ii) Requirements specification, which entails the output requirements, input, hardware and software required for the new system.

iii) System flowchart, which shows the overall functionality of the proposed information system.

iv) Tables or file structures depending on the approach used in system construction.

v) Sample test data to test whether the new computerized information system is working as expected.

vi) User manual, which helps the user work with the new system minimal guidance . The manual contains information like.

- ✓ How to install, start the system
- ✓ The interface of the system
- ✓ How to carry out various tasks
- ✓ Error collection and how to get help.

A System:

A system is a set of ***inter-relate*** components/Elements set together to perform a given task.

Information Systems (IS):

These are interrelated elements (hardware, software, users, data and information) working together to collect, process, store, and disseminate information for:

- Decision making
- Coordination
- Control
- Analysis
- And visualization (conceptualization/seeing complex subjects for org'nal survival.

Types of Information systems.

Can be classified basing on;

- Behavior of the system
- Nature of work done by the system

Therefore information systems can be;

- **Deterministic system:** - Is a system where given the input the output can be determined.
- **Probabilistic/stochastic system**. Is a system where given the input the output cannot be successfully determined.

- **Self–organizing/Adaptive/Cybernetic systems.** Systems which are highly complex. i.e. system which continuous adapt to changes in the environment.
- Computer based information systems (CBIS): these rely on computer HW and SW
- Formal systems: These are systems that rest on clearly fixed and accepted data and procedures with predefined rules.
- **Management Information System (MIS):**

Focuses on computer based information systems aimed at organizational management provide reports and on-line access to the orgn's records

Are system that convert data from internal and external source into information to be used by managers if effective decision making for planning, Directing and control of original activities.

They include:

Executive information system (EIS) or Executive Support system ESS.

Systems used at the strategic level of management to facilitate non-routine decision making through illustrations and communication. They are at highest levels of control.

EIS/ESS usually provides summarized reports.

Decision support Systems (DSS)
MISs at the organization's mgt level combining data and complex models or tools to facilitate decision making. Used to help manager in effective decision-making where data is unstructured.
Unstructured data is one with very high level of uncertainty and difficulty to making the right decision using it.

Expert System:
Systems designed to provide specialized information/data for specialized areas or fields. E.g.
Accounting/Finance expert system.
Marketing expert system.
Human resource expert system.
Health applications

Systems can also be classified as;

- Data processing systems.
- Transaction system.
- Knowledge based system

Data processing systems (DPS):
These are systems that automate many of the routine clerical and administrative procedures in organisation. For document processing, order mgt, Stock control, and Routine billing of clients.
Management can not base on them for tactical and strategic decisions.

On-line Analytical Processing Systems (OLAPS):
These are real-time Interactive systems with direct connectivity between the data source and the users.

Transaction Processing System (TPS):
Support the operation of organizations by processing transactions as they and keeping them in master files. They work as the interface between the organization and its customers.

Other Systems include;
Neural Network System
Artificial intelligence systems.
Fuzzy logic systems
Open systems
Closed Systems

Information system literacy:
Involves a broad based understanding of information systems given the behavioural knowledge of orgns, individuals benefiting and using the IS, and technical knowledge about ICTs/Computers

Benefits of ISs

- Making organisational structures flatter.
- Separation of work from location is minimised. eliminated the physical distance

- Slowly doing away with manual work procedures as workflows get reorganised
- Orgns are becoming increasing flexibility as they can now easily sense and respond to challenges and changes in the work and market place.
- Redefinition of org'nal boundaries: as businesses get conducted across boundaries.
- Managers can now coordinate, plan, supervise and control organisational functions better on implementation of MIS

SYSTEM DEVELOPMENT

This (SD) defines the activities an organization should go through to come up with a properly functioning information system. The most common approaches to SD are;

- **System life cycle methodology**
- **Application software packages**
- **Prototyping approach**

1. System life cycle: This is an approach that **parcels** the SD process into formal stages that must be followed sequentially.
Logical stages:

- **System/situational analysis**
    - **System project definition**
    - **Feasibility study**
    - Needs assessment or problem identify and definition.
    - Specify solution
    - Establishment of information system requirements
    - **System study**. Looks at the problems of the existing system, defines the objectives the new system is to attain, and evaluates the various alternative solutions. Writes a system proposal report
- **System design** – Details how the new system is going to meat the information requirements of the organization. It creates design specifications for the system solution. Designs include inputs, outputs, user interface, databases, electronic processing and manual procedures designs.
- **Programming** – usually done by a technical/professional programmer who translates design specifications into program codes
- **Testing**: Is a thorough and exhaustive process aimed at determining as to whether the system gives the desired results. It involves **unit testing**, **system testing,** and **acceptance testing**. It also involves drawing conscious **test plans**.
- **Installation and documentation:** involves actual installation, re-testing, training of users and writing of user's manuals. Documentation involves a descriptive write-up about how the new system works. It is both technical and end-user oriented
- **Conversion stage: Involves switching from the old IS to the new one.**
    - **Parallel conversion strategy**
    - **Direct cutover**
    - **Pilot study strategy. Where the new system is introduce to a limited area**
    - **Phased approach.** Introduce in stages either by function or unit.

- **Production:** The system is being put to actual use, while being evaluated by end-users and technical specialists.
- **Maintenance stage:** This is a **post implementation phase** involving **servicing and repair** of the system**, while** making changes in **hardware, software and documentation.**

2. Application software packages:
These are sets of prewritten or coded software programs which are commercially available for sale or lease. Some are specialized while other generalized software applications can be customized.

3. Prototyping:
A Prototype is a preliminary working version of an information system for demonstration and evaluation. It is an *original functional model* of a new product (software) that a developer can put into the hands of potential users or customers, so that they can see it, test it and use it.

Prototyping is one the most effective ways of gauging the viability of a product (an information system).

Dimensions of Prototypes

- ***Horizontal Prototype.*** *This provides a broad view* ***of an entire system or subsystem, focusing on user interaction more than low-level system functionality, such as database access. It is a common term for a User Interface Prototype***
- ***Vertical Prototype:*** This is a more complete elaboration of ***a single subsystem or function. It is useful for obtaining detailed requirements for a given function, with the following benefits:***

Types of Prototyping

1. Throwaway Prototyping also called ***close ended prototyping*** or ***Rapid Prototyping: refers to the creation of a model that will eventually be discarded rather than becoming part of the final delivered software.***

2. Evolutionary Prototyping: - The main goal when using Evolutionary Prototyping is to build a very <u>robust</u> <u>prototype</u> in a structured manner and constantly refine it. "The reason for this is that the Evolutionary prototype, when built, forms the heart of the new system, and the improvements and further requirements will be built.

3. Incremental Prototyping: This works in such a way that in Evolutionary Prototyping, developers can focus themselves to develop parts of the system that they understand instead of working on developing a whole system.

4. Extreme Prototyping: Extreme Prototyping as a development process is used especially for developing web applications. Basically, it breaks down web development into three phases, each one based on the preceding one:
The process is called Extreme Prototyping to draw attention to the second phase of the process, where a fully-functional UI is developed with very little regard to the services other than their contract.

The DSDM lifecycle of a prototype is to:

1. ***Identify Prototype: This requires the determination of the basic requirements including the input and output information desired. Well, details, such as security, can typically be ignored at this level.***

2. Agree to a Plan: At this stage, the initial prototype is developed that includes only user interfaces. (See Horizontal Prototype, below)
3. Create the INITIAL Prototype: At this stage, the initial prototype is developed that includes only user interfaces. (See Horizontal Prototype, below)
4. Review the Prototype: Using the feedback both the specifications and the prototype can be improved. Negotiation about what is within the scope of the contract/ product may be necessary. If changes are introduced then a repeat of steps #3 and #4 may be needed.
Advantages of Prototyping
1. Reduced Time and Costs: Prototyping can improve the quality of requirements and specifications provided to developers.

2. Improved and Increased User Involvement: Prototyping requires user involvement and allows them to see and interact with a prototype allowing them to provide better and more complete feedback and specifications.

Disadvantages of Prototyping

1. Insufficient Analysis: The focus on a limited prototype can distract developers from properly analyzing the complete project.

2. **User Confusion of Prototype and Finished System**: Users can begin to think that a prototype, intended to be thrown away, is actually a final system that merely needs to be finished or polished.
3. **Developer Misunderstanding of User Objectives**: Developers may assume that users share their objectives (e.g. to deliver core functionality on time and within budget), without understanding wider commercial issues. For example, user representatives attending Enterprise software (e.g. PeopleSoft) events may have seen demonstrations of "transaction auditing" (where changes are logged and displayed in a difference grid view) without being told that this feature demands additional coding and often requires more hardware to handle extra database accesses.

4. Developer Attachment to Prototype: Developers can also become attached to prototypes they have spent a great deal of effort producing; this can lead to problems like attempting to convert a limited prototype into a final system when it does not have an appropriate underlying architecture.

5. Excessive Development Time of the Prototype: A key property to prototyping is the fact that it is supposed to be done quickly.

6. Expense of Implementing Prototyping: the start up costs for building a development team focused on prototyping may be high. Many companies have development methodologies in place, and changing them can mean retraining, retooling, or both.

7. A common problem with adopting prototyping technology is high expectations for productivity with insufficient effort behind the learning curve. In addition to training for the use of a prototyping technique, there is an often overlooked need for developing corporate and project specific underlying structure to support the technology.

PAST KCSE QUESTIONS ON THE TOPIC

1. 2002

Give four reasons why a firm may decide to computerize its operations (4 marks)

2005

Explain the following terms as used in program implementation (2 marks)

i) Parallel running

ii) Direct changeover.

1. 2006

List two methods of gathering information during system development process. (2marks)

4. 2007

a manager wishes to replace the current manual system with a computerized one.

a) Describe three main areas that must be evaluated to justify the replacement (6 marks)

b) List three areas that would be considered in the requirement specifications (3 marks)

c) State and explain three ways that can be followed to replace the current system. (6 marks)

PRACTICE QUESTIONS ON THE TOPIC

1. Distinguish between soft system and hard system

2. Explain the purpose of an information system in an organization.

3. Explain any three circumstances that necessitate the development of new information systems.

2. What is feasibility study in relation to system development?
3. State 3 advantages of using interview method in information gathering
4. Highlight three circumstances in which a questionnaire will be the best method to collect data.
5. Explain the seven stages of a system development
6. Differentiate between parallel and pilot changeover method.
7. State for importance of training the staff on implementation of a new system.
8. Explain three tasks carried out during system development.

PREDICTION QUESTIONS ON THE TOPIC.

1. Name three circumstances in which it is better to use questionnaire than interview for gathering information.

2. Distinguish between parallel and pilot modes of changeover when migrating from a manual system to a computerized system. (2 marks)

3. Describe what happens in the second phase of the systems development life cycle. (2 marks)

4. Explain the following characteristics of a system

i) Holistic thinking

ii) System entropy

3. List down 2 importance of training the staff / user after system implementation (2 marks)
4. Explain three situations that can initiate development of a new information system (3 marks)

7. Distinguish between open and closed system (2 marks)

8. State 2 disadvantages of observation when used infact gathering (2 marks)