or 0248043888
https://www.TeachersAvenue.net
https://TrendingGhana.net
https://www.mcgregorinriis.com

BASIC 8

WEEKLY LESSON PLAN – WEEK 6

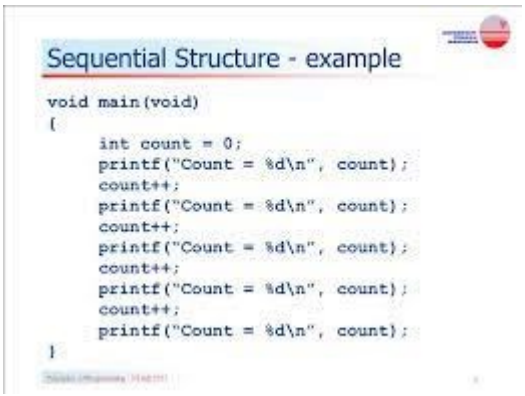
Strand:	Computational Thinking	**Sub-Strand:**	Algorithm		
Content Standard:	B8.4.2.1.Analyse the correct step-by-step procedure in solving any real-world problem				
Indicator (s)	B8.4.2.1.1 Apply variables, expressions, assignment statements and operator precedence order (BODMAS rule) to process and store numbers and text in a Programme. B8.4.2.1.2 Describe and use sequence, selection and iteration statements in a Programme. Understand the difference between variables and constants and be able to choose appropriate naming conventions when writing statements.	**Performance Indicator:** Learners can apply BODMAS rule to process and store numbers and text in Programme.			
Week Ending	04-08-2023				
Class	B.S.8	**Class Size:**		**Duration:**	
Subject	Computing				
Reference	Computing Curriculum, BS7 Computing Textbook, Teachers Resource Pack, Learners Resource Pack				
Teaching / Learning Resources	Chart, Poster, Video.	**Core Competencies:**	• Communication and Collaboration • Digital Literacy		

DAY/DATE	**PHASE 1 : STARTER**	**PHASE 2: MAIN**	**PHASE 3: REFLECTION**
THURSDAY	Review Learners knowledge on BODMAS rule they learnt in Mathematics.	1. Discuss with Learners about the meaning precedence of an operator in programming. 2. Assist Learners to identify examples of assignment statements. 3. Demonstrate for Learners to observe on how to place the operations in order of precedence 4. Learners brainstorm to identify the assignment operator that assignment statement use. 5. Assist Learners to compute an expression following the operator precedence order (BODMAS).	Through questions and answers, conclude the lesson. **Exercise;** 1. What is Precedence of an operator in Programming 2. Stat 3 examples of

		to exemplify how computers process input data to print out answer. **Precedence of an Operator in Programming;** The precedence of an operator specifies how "tightly" it binds two expressions together. For example, in the expression 1 + 5 * 3 , the answer is 16 and not 18 because the multiplication ("*") operator has a higher precedence than the addition ("+") operator. Parentheses may be used to force precedence, if necessary. **Assignment Statements** Assignment statements enable the programmer to define or redefine a symbol by assigning it a value. This value may be a reference to another symbol, register name, or expression. The new value takes effect immediately and remains in effect until the symbol is redefined. Symbols defined in assignment statements do not have forward references. In addition, symbols defined in assignment statements cannot: • appear in the symbol table of an output object file. • be declared global. • be defined in an equate statement. There are two types of assignment statements: • Symbol assignment statements, which define or redefine a symbol in the symbol name space. • Register assignment statements, which define or redefine a register name in the symbol name space. **Symbol Assignment Statements** A symbol assignment statement has the following syntax: *identifier=expression* // comments Where: *identifier* — Represents a symbol in the symbol name space. *expression* — Specifies the type and value of the identifier. The expression	assignment statement.

		<table><tr><td></td><td>cannot contain forward references.</td></tr></table> The following is an example of an assignment statement that defines a symbol: C = L0+2 **Register Assignment Statements** A register assignment statement has the following syntax: *identifier=register name* // comments Where: <table><tr><td>*identifier*</td><td>Represents a register name in the symbol name space.</td></tr><tr><td>*register name*</td><td>Specifies an alternate register name. If the register name is a stack or rotating register name, the new register name continues to reference the previously-defined register name, even if the name is no longer in effect. See the Register Stack Directive and Rotating Register Directives sections.</td></tr></table> The following is an example of an assignment statement that defines a register name: A = r1	
FRIDAY	Discuss with Learners on the meaning of sequence, selection and iteration in programming	1. Assist Learners to describe the use of sequence, selection and iteration. 2. Learners brainstorm to identify examples of sequence in Computer Programming 3. Using a Poster, explain three types of iteration in Programming. 4. Discuss with Learners on the three programming structures used in programming. 5. Demonstrate on the three 3 general methods of implementing a programming language. **Sequence in Computer Programming;** Sequence is the first programming construct. In programming, statements are executed one after another. Sequence is the order in which the statements are executed. The sequence of a	Learners in small groups to discuss and report to the class on the difference between selection statement and iteration statement

program is extremely important as carrying out instructions in the wrong order leads to a program performing incorrectly.



The use of sequence in programming;

Sequence is the first programming construct. In programming, statements are executed one after another. Sequence is the order in which the statements are executed. The sequence of a program is extremely important as carrying out instructions in the wrong order leads to a program performing incorrectly

For example, a very simple algorithm for brushing teeth might consist of these steps:

1. put toothpaste on toothbrush
2. use toothbrush to clean teeth
3. rinse toothbrush

Each step is an instruction to be performed. Sequencing is the order in which the steps are carried out.

Why is sequencing important?

It is crucial that the steps in an algorithm are performed in the right order - otherwise the algorithm will not work correctly. Suppose the steps for the teeth-cleaning algorithm were in this sequence:

1. use toothbrush to clean teeth
2. put toothpaste on toothbrush
3. rinse toothbrush

		A toothbrush would still be used to clean the teeth and toothpaste would still be put on the brush. But because steps 1 and 2 are in the wrong sequence the teeth wouldn't get cleaned with the toothpaste, and the toothpaste would be wasted. A human would realise they had forgotten to add toothpaste at the start of the process, but a computer would not know that anything was wrong. A computer can only do what it is programmed to do. If the steps are programmed in the wrong sequence, the computer will perform the tasks in this sequence – even if this is incorrect.	

Name of Teacher: ***School:*** ***District:***