Grade 9 Pre-Technical Studies

Term 2 - 47 complete lesson plans

Ready to teach, in the KICD lesson-plan format | 40-minute lessons | PDF and editable Word

ABOUT THIS PACK

47 complete, ready-to-teach Grade 9 Pre-Technical Studies lesson plans for Term 2, covering visual programming, wood, handling waste materials, holding tools and driving tools.

TOPICS IN THIS PACK

Topic	Name	Lessons
2.2	Visual Programming	15
3.1	Wood	8
3.2	Handling Waste Materials	8
4.1	Holding Tools	8
4.2	Driving Tools	8

INSIDE EVERY LESSON PLAN

- The specific learning outcomes, key inquiry question, core competencies, values and PCIs, and the resources needed.
- A timed lesson flow: what the teacher does and what the learners do, from introduction to summary.
- The key content (a worked example, notes or a practical with its expected results), then four practice questions with the answers for the teacher.
- The common mistake to watch for, and ideas for learners who need support and for those ready for more.
- A four-level CBC assessment rubric (EE, ME, AE, BE), homework, and a blank reflection space to fill in after teaching.

A TASTE OF LESSON 1

What visual programming is

By the end of the lesson, the learner should be able to: a) define visual programming; b) name visual programming tools

> **Programming with blocks**
> A program is a set of instructions a computer follows. Visual programming builds programs by snapping together blocks instead of typing code.
> Tools: Scratch, ScratchJr, Blockly, MIT App Inventor, Code.org.

You also receive this pack as an editable Word file, so you can change any lesson to suit your class before you print it. Lessons follow the order of the KICD curriculum design.

CONTENTS

No.	Lesson	Topic	Page
1	What visual programming is	2.2 Visual Programming	4
2	Application areas of visual programming	2.2 Visual Programming	7
3	The Scratch interface	2.2 Visual Programming	10
4	Events and sequence: making a sprite move and talk	2.2 Visual Programming	13
5	Loops: repeating actions	2.2 Visual Programming	16
6	Conditions: making decisions	2.2 Visual Programming	19
7	Variables: keeping score	2.2 Visual Programming	22
8	Interaction: asking questions and building a quiz	2.2 Visual Programming	25
9	Planning an interactive story or animation	2.2 Visual Programming	28
10	Building the story: scenes and messages	2.2 Visual Programming	31
11	Planning a simple game	2.2 Visual Programming	34
12	Building the game	2.2 Visual Programming	37
13	Testing and debugging	2.2 Visual Programming	40
14	Presenting applications	2.2 Visual Programming	43
15	Visual programming in day-to-day life	2.2 Visual Programming	46
16	Wood as a material	3.1 Wood	49
17	Softwoods and hardwoods	3.1 Wood	52
18	Physical characteristics and sorting wood	3.1 Wood	55
19	Preparing wood: conversion	3.1 Wood	58
20	Preparing wood: seasoning	3.1 Wood	61
21	Uses of wood in different trades	3.1 Wood	64
22	Exploring wood use in the locality	3.1 Wood	67
23	The importance of wood and its conservation	3.1 Wood	70
24	Types of waste materials	3.2 Handling Waste Materials	73
25	Effects of poor waste handling	3.2 Handling Waste Materials	76
26	Safe ways of handling waste: reduce, reuse, recycle	3.2 Handling Waste Materials	79
27	Safe ways of handling waste: composting and burning	3.2 Handling Waste Materials	82
28	Safety when handling waste	3.2 Handling Waste Materials	85
29	Practical: sorting and disposing of school waste	3.2 Handling Waste Materials	88
30	Waste handling in the locality	3.2 Handling Waste Materials	91
31	The need for proper waste handling	3.2 Handling Waste Materials	94
32	Identifying holding tools	4.1 Holding Tools	97

33	Uses of different holding tools	4.1 Holding Tools	100
34	Selecting holding tools for tasks	4.1 Holding Tools	103
35	Safe use of holding tools	4.1 Holding Tools	106
36	Practical: using a vice and clamps	4.1 Holding Tools	109
37	Practical: using pliers, tongs and clips	4.1 Holding Tools	112
38	Caring for and storing holding tools	4.1 Holding Tools	115
39	Importance of holding tools	4.1 Holding Tools	118
40	Identifying driving tools	4.2 Driving Tools	121
41	Uses of driving tools	4.2 Driving Tools	124
42	Selecting driving tools and safe use	4.2 Driving Tools	127
43	Practical: driving and pulling nails	4.2 Driving Tools	130
44	Practical: driving screws	4.2 Driving Tools	133
45	Practical: using spanners and punches	4.2 Driving Tools	136
46	Caring for driving tools	4.2 Driving Tools	139
47	Importance of driving tools	4.2 Driving Tools	142

ELIMU254

What visual programming is

Strand 2.0 Communication in Pre-Technical Studies > Sub-strand 2.2 Visual Programming | Lesson 1 of 15 | 40 minutes

SCHOOL	TEACHER	DATE
CLASS / STREAM	TIME	ROLL (B / G)

SPECIFIC LEARNING OUTCOME(S)

By the end of the lesson, the learner should be able to: a) define visual programming; b) name visual programming tools

KEY INQUIRY QUESTION(S)

- How are applications developed using visual programming software?

CORE COMPETENCIES, VALUES AND PCIS IN THIS LESSON

COMPETENCY	**Self-efficacy**	Learners attempt each task with confidence and see that effort improves results.
COMPETENCY	**Critical thinking and problem solving**	Learners choose a method, check their answers and give reasons for their choices.
VALUE	**Social justice**	Every learner gets a turn and a fair chance to answer.
VALUE	**Responsibility**	Learners look after shared materials and finish every task.
PCI	**Peer education and mentorship**	Developed through the activities in this lesson.
PCI	**Internet safety and security**	Learners use shared devices and the internet responsibly and keep personal details private.

LEARNING RESOURCES

Computers or tablets with Scratch (online or offline) | Printed block cards for unplugged planning | A projector (if available) | Learners' exercise books

ORGANISATION OF LEARNING

Pairs share a device (driver and navigator, swapping every 10 minutes); groups without a device plan scripts on paper with block cards.

KEY WORDS

visual programming | block | sprite | stage | script | event | loop | condition | variable | debug

THE LESSON AT A GLANCE

Introduction	Teacher models	Guided practice	On their own	Wrap-up
6 min	9 min	10 min	9 min	6 min

LESSON FLOW

Stage	Minutes	The teacher	The learners
Introduction	6	Greet the class and ask: "How does a phone game know what to do when you tap the screen?" Take two or three answers, link them to earlier work, then state today's outcome.	Answer the starter question, recall what they already know and read the outcome aloud.
Step 1: Teacher models	9	Take learners through "Programming with blocks" (key ideas on the next page), explaining as you go. Use computers or tablets with Scratch (online or offline), printed block cards for unplugged planning, a projector (if available). Ask learners to predict what comes next before you show it.	Watch, predict the next step, ask questions and note the key steps in their exercise books.
Step 2: Guided practice	10	Give questions 1 and 2. Move round the groups, ask "How do you know?" and correct errors kindly on the spot.	Work in pairs or small groups, explain each step to a partner and check one another's answers.
Step 3: Independent practice	9	Set questions 3 and 4 for individual work. Mark as you walk round and note who needs help in the next lesson.	Work on their own, then swap books with a partner to check.
Summary and conclusion	6	Ask two or three learners to state the key idea in their own words. Give this exit question and check the answers: "Why do beginners make fewer errors with blocks?"	Say the key idea in their own words and answer the exit question in the exercise book.

KEY IDEAS

Programming with blocks
A program is a set of instructions a computer follows. Visual programming builds programs by snapping together blocks instead of typing code.
Tools: Scratch, ScratchJr, Blockly, MIT App Inventor, Code.org.
Blocks only fit where they make sense, so beginners make fewer errors.

PRACTICE QUESTIONS - WITH ANSWERS FOR THE TEACHER

No.	Question	Answer
1	What is a program?	**A set of instructions a computer follows.**
2	How is visual programming different from typed code?	**Programs are built by snapping blocks together.**
3	Name two visual programming tools.	**For example Scratch and App Inventor.**
4	Why do beginners make fewer errors with blocks?	**Blocks only fit where they make sense.**

WATCH OUT FOR

Learners think visual programming is "not real programming". It uses the same ideas.

ASSESSMENT RUBRIC - THE FOUR CBC LEVELS

Level	What the learner can do
Exceeds expectations (EE)	Can define visual programming confidently and accurately, explain the method used, and apply the idea to new or unfamiliar problems.
Meets expectations (ME)	Can define visual programming accurately in familiar situations; notices and corrects slips without help.
Approaches expectations (AE)	Can partly define visual programming, or does so with prompts from the teacher or a partner; makes some errors.
Below expectations (BE)	Cannot yet define visual programming without direct help; needs the idea taught again with concrete objects.

SUPPORT AND EXTENSION

Learners who need support	Show a short Scratch project.
Learners ready for more	Search for one app made with MIT App Inventor. (Answer: Answers will vary)
Inclusion	Seat learners with low vision near the board and read aloud whatever you write; use large, clear, tactile materials; allow extra time; pair with a patient partner; use gestures or Kenyan Sign Language for the key words.

HOMEWORK AND FOLLOW-UP

Redo questions 3 and 4 neatly in the exercise book. Then tell someone at home what you learned about "what visual programming is" and bring one question of your own to ask next lesson.

TEACHER SELF-CHECK

- Did every learner get a turn to handle materials or explain their thinking?
- Did most learners meet the outcome? Write down the names of those who did not.
- Was the pace right — did any step take too long or too short a time?
- Which competency and value did I see learners showing?
- What will I change the next time I teach this lesson?

REFLECTION ON THE LESSON / SELF-REMARKS

Write this after teaching - what worked, what to change, which learners need follow-up.

Application areas of visual programming

Strand 2.0 Communication in Pre-Technical Studies > Sub-strand 2.2 Visual Programming | Lesson 2 of 15 | 40 minutes

SCHOOL	TEACHER	DATE
CLASS / STREAM	TIME	ROLL (B / G)

SPECIFIC LEARNING OUTCOME(S)

By the end of the lesson, the learner should be able to: a) explain where visual programming is used; b) link it to problem solving

KEY INQUIRY QUESTION(S)

- How are applications developed using visual programming software?

CORE COMPETENCIES, VALUES AND PCIS IN THIS LESSON

COMPETENCY	**Self-efficacy**	Learners attempt each task with confidence and see that effort improves results.
COMPETENCY	**Critical thinking and problem solving**	Learners choose a method, check their answers and give reasons for their choices.
VALUE	**Social justice**	Every learner gets a turn and a fair chance to answer.
VALUE	**Responsibility**	Learners look after shared materials and finish every task.
PCI	**Peer education and mentorship**	Developed through the activities in this lesson.
PCI	**Internet safety and security**	Learners use shared devices and the internet responsibly and keep personal details private.

LEARNING RESOURCES

Computers or tablets with Scratch (online or offline) | Printed block cards for unplugged planning | A projector (if available) | Learners' exercise books

ORGANISATION OF LEARNING

Pairs share a device (driver and navigator, swapping every 10 minutes); groups without a device plan scripts on paper with block cards.

KEY WORDS

visual programming | block | sprite | stage | script | event | loop | condition | variable | debug

THE LESSON AT A GLANCE

Introduction	Teacher models	Guided practice	On their own	Wrap-up
6 min	9 min	10 min	9 min	6 min

LESSON FLOW

Stage	Minutes	The teacher	The learners
Introduction	6	Greet the class and ask: "What problems in school could a simple app solve?" Take two or three answers, link them to earlier work, then state today's outcome.	Answer the starter question, recall what they already know and read the outcome aloud.
Step 1: Teacher models	9	Take learners through "Where it is used" (key ideas on the next page), explaining as you go. Use computers or tablets with Scratch (online or offline), printed block cards for unplugged planning, a projector (if available). Ask learners to predict what comes next before you show it.	Watch, predict the next step, ask questions and note the key steps in their exercise books.
Step 2: Guided practice	10	Give questions 1 and 2. Move round the groups, ask "How do you know?" and correct errors kindly on the spot.	Work in pairs or small groups, explain each step to a partner and check one another's answers.
Step 3: Independent practice	9	Set questions 3 and 4 for individual work. Mark as you walk round and note who needs help in the next lesson.	Work on their own, then swap books with a partner to check.
Summary and conclusion	6	Ask two or three learners to state the key idea in their own words. Give this exit question and check the answers: "How does visual programming help problem solving?"	Say the key idea in their own words and answer the exit question in the exercise book.

KEY IDEAS

Where it is used
Interactive stories and animations (teaching a lesson), games (learning tables), quizzes, simulations (how the water cycle works), simple phone apps (a timetable reminder), controlling robots and microcontrollers (micro:bit).
Examples of problems: a quiz to revise for exams, an animation explaining handwashing, a game teaching road safety.
Visual programming helps people plan logic step by step.

PRACTICE QUESTIONS - WITH ANSWERS FOR THE TEACHER

No.	Question	Answer
1	Name three application areas.	**For example stories, games and quizzes.**
2	Suggest an app to help revise.	**A quiz.**
3	What device can be controlled with block code?	**For example a micro:bit or robot.**
4	How does visual programming help problem solving?	**It helps plan logic step by step.**

WATCH OUT FOR

Ideas are too big (a full social media app). Keep them simple.