



SuperNovaSchool®
L E A R N T O B E D I S T I N C T

Computing

Grade VI

(Instructional Resource)

Unit:1

**Computational Thinking &
Programming**

Books

**Cambridge Primary Computing,
Learner's book stage 6**

Session

2025-26

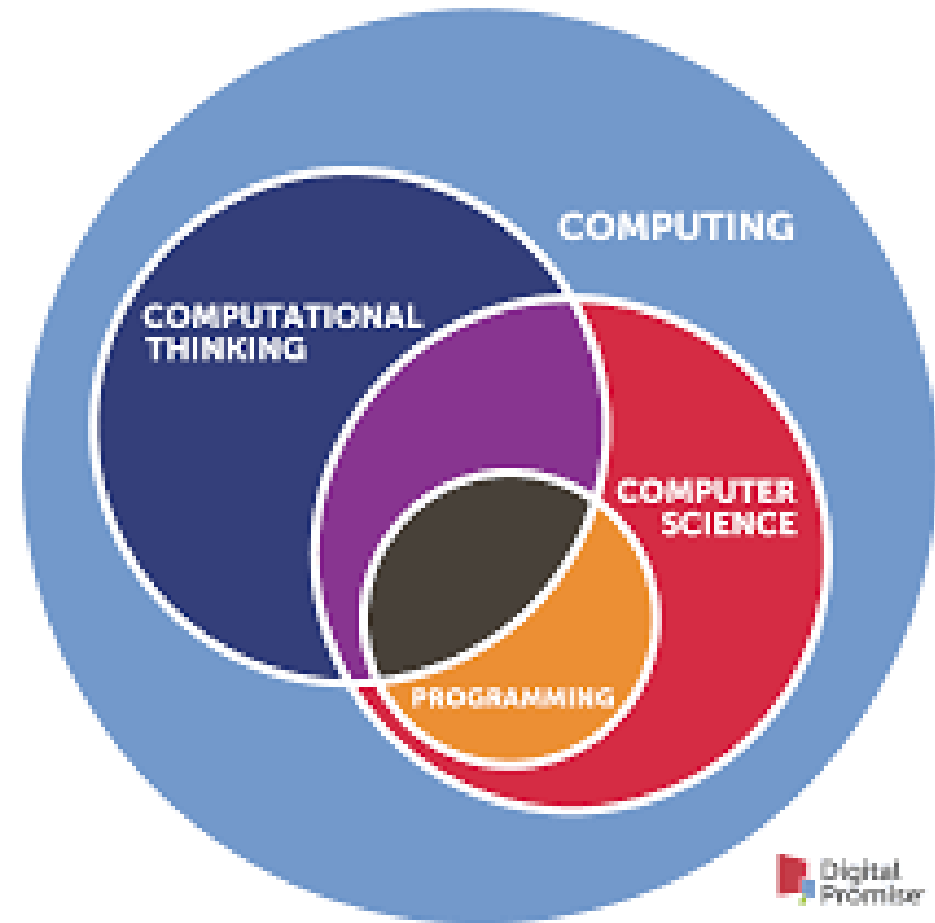
Prepared By

Ayesha Usman



Sub-Topics

- 1.1 Planning Flowcharts
- 1.2 Programming Constructs
- 1.3 Sub-Routines
- 1.4 Planning Programs
- 1.5 Evaluating and Testing Programs
- 1.6 Using Variables with a physical device





1.1 Planning Flowcharts

What is Computational Thinking?

- **Computational Thinking is a way of solving problems by breaking them down into smaller steps that a computer or a human can understand and follow.**
- **It is thinking like a computer—breaking big problems into smaller parts, spotting patterns, making rules (algorithms), and checking if the solution works.**





The Process of Computational Thinking

Computational thinking involves four steps:

- **Decomposition** – Breaking a big problem into smaller ones.
- **Pattern Recognition** – Finding similarities or patterns.
- **Abstraction** – Focusing on important details, ignoring the rest.
- **Algorithms** – Creating a step-by-step solution.





What is an Algorithm?

- An algorithm is a set of clear, step-by-step instructions used to solve a problem or complete a task.
- It tells you exactly what to do, in what order, to get something done.
- For example, a recipe to bake a cupcake



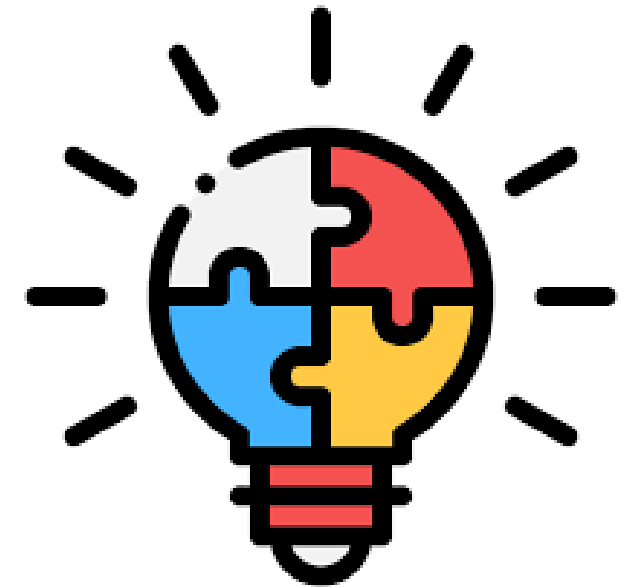


What is Logical Reasoning?

Logical reasoning is the process of analysing a problem, making predictions and finding solutions.

The application of logical reasoning can include:

- Thinking based on real-world evidence
- Drawing on prior knowledge and experience of programming
- Borrowing ideas, blocks of code and techniques or constructs seen in other computing applications
- Solving complex problems by breaking them down, through decomposition
- Predicting, from facts and evidence

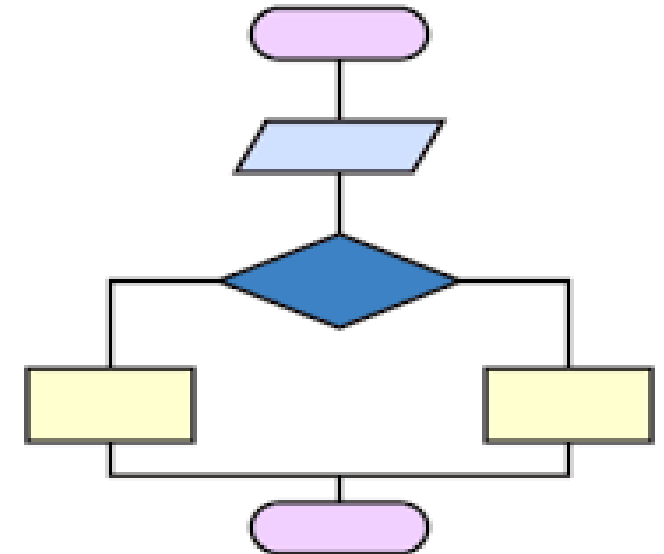


Unit:1 Computational Thinking and Programming

1.1 Planning Flowcharts

What is a Flowchart?

- A Flowchart is a graphical representation of an algorithm.
- It is a diagram that uses different symbols to represent the steps in an algorithm



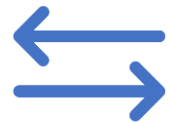
Unit:1 Computational Thinking and Programming

1.1 Planning Flowcharts

Why do we use Flowcharts to represent Algorithms?



Easy to Understand – Pictures help us see the steps clearly.



Shows the Order – Arrows show what happens first, next, and last.



Helps with Planning – It's a great way to plan before writing code.



Easier to Spot Mistakes – We can find problems before we start programming



Good for Teamwork – Others can understand your idea quickly.






Unit:1 Computational Thinking and Programming

1.1 Planning Flowcharts

Symbols used in a Flowchart

The shapes used in a flowchart have different names and meanings

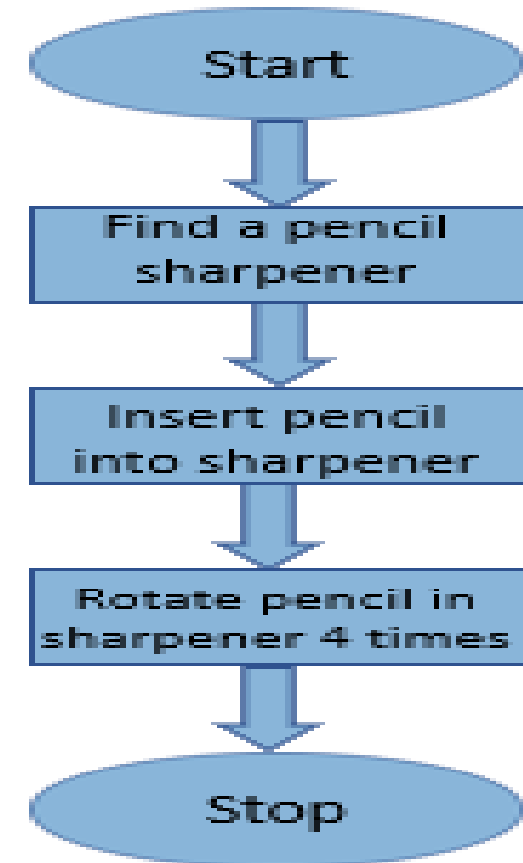
- Oval represents start or stop
- Arrow or connectors represents the flow
- Parallelogram represents input or output
- Rectangle represents a process
- Diamond represents a decision

Shapes	Name
	Start or end
	Connectors
	Input or output
	Process
	Decision

Unit:1 Computational Thinking and Programming

1.1 Planning Flowcharts

- Can you explain what you see in this flowchart?
- Can you identify the inputs?
- Can you identify the outputs?
- What are the processes in this flowchart?



Unit:1 Computational Thinking and Programming

1.1 Planning Flowcharts

Predicting the outcomes of Flowcharts

We can use flowcharts to make predictions.

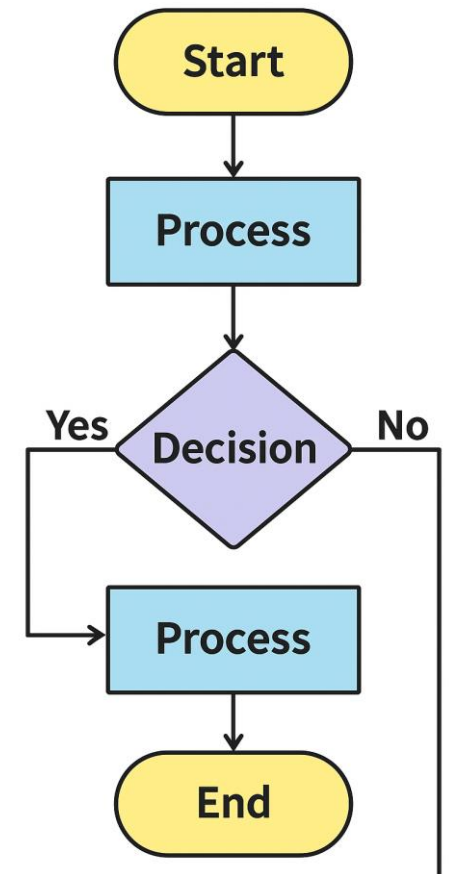
Follow the Path: You can trace the arrows step by step to see what will happen next.

Check Conditions: Decision shapes help us see what happens if something is true or false.

See Different Outcomes : You can predict different results by following different paths in the chart.

Spot Errors Early: Predict where something might go wrong before writing the actual code.

Think Like a Computer: It helps you think logically about what the program will do before it runs.



Unit:1 Computational Thinking and Programming

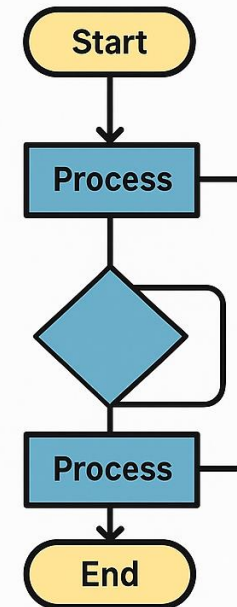
1.1 Planning Flowcharts

Types of Flowcharts

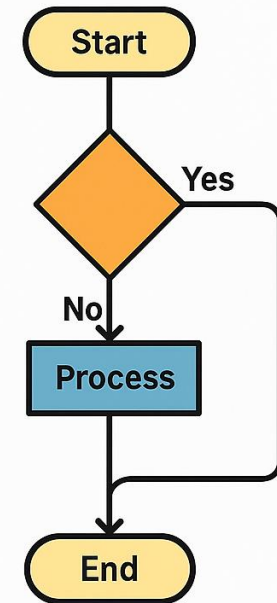
- Linear
- Flowchart with a loop
- Flowchart with a decision



Linear
Flowchart



Flowchart
with a Loop

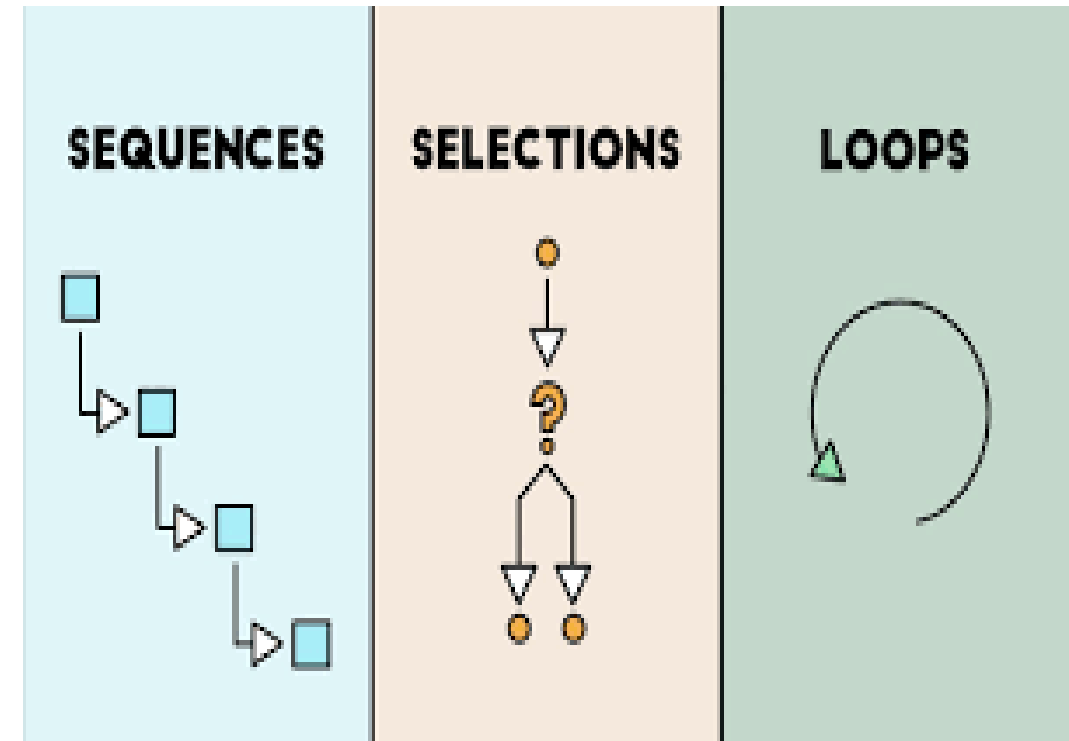


Flowchart
with a Decision

Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

- Programming constructs are the basic building blocks of any computer program.
- They help us control the order, decisions, and repetition of instructions.
- There are three types of programming constructs
 1. Sequence
 2. Selection
 3. Repetition or Iteration

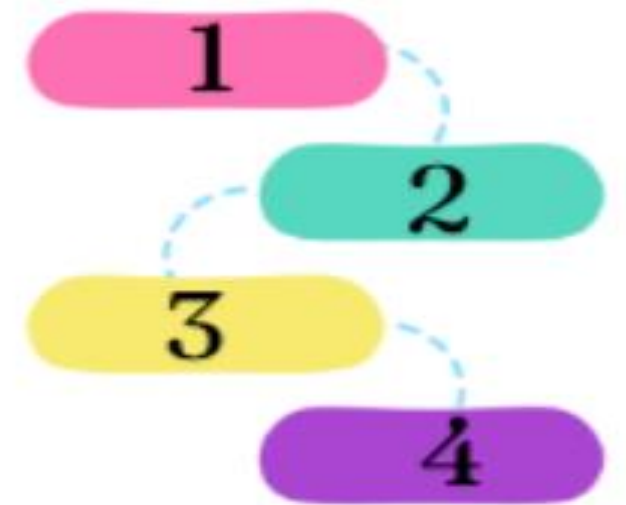


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

- Sequence is the order in which the program or an algorithm statements run.
- If the program sequence is wrong, then it might not give correct output
- In this programming construct, the instructions are carried one after another.
- For Example, To search on a website, we need to follow this sequence:
 - 1) Turn on computer
 - 2) Open browser
 - 3) Visit website
 - 4) Search for the required information

Sequence



Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

- Selection is a programming construct in which the program makes a choice based on a condition.
- Selection is done using conditional statements.
- A conditional statement is a section of code that tells your program to run either one set of instructions or another set depending on whether the condition is true or false

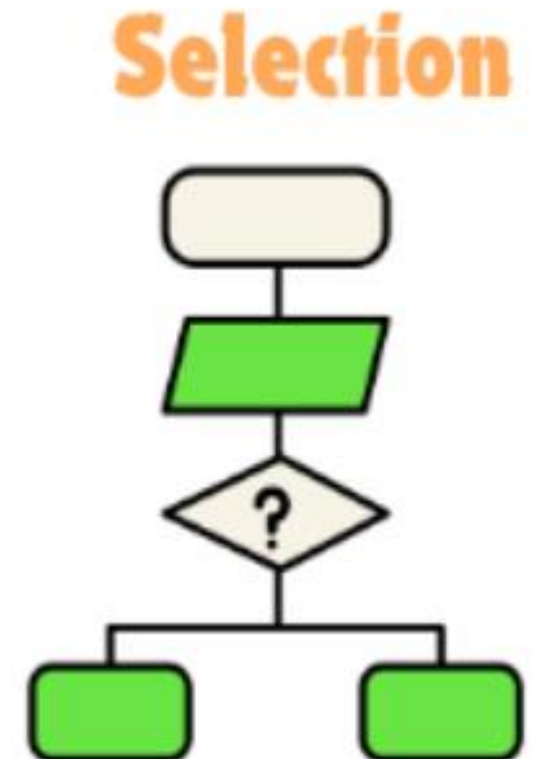
- For example:

If it's raining

Take an umbrella

Else

Wear sunglasses



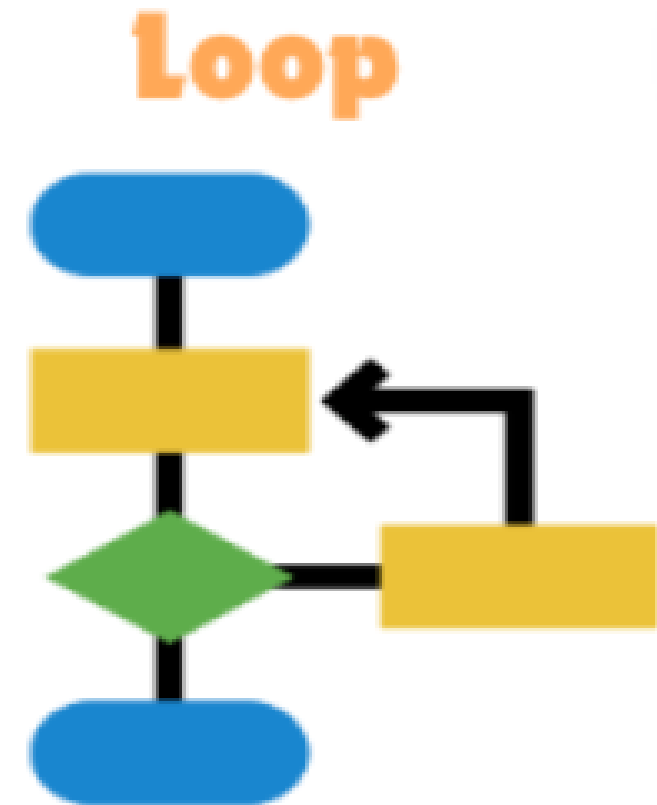
Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

- Repetition or Iteration is a programming construct which repeats a set of instructions several times.
- It is called a Loop in programming

There are two main types of iteration(Loops):

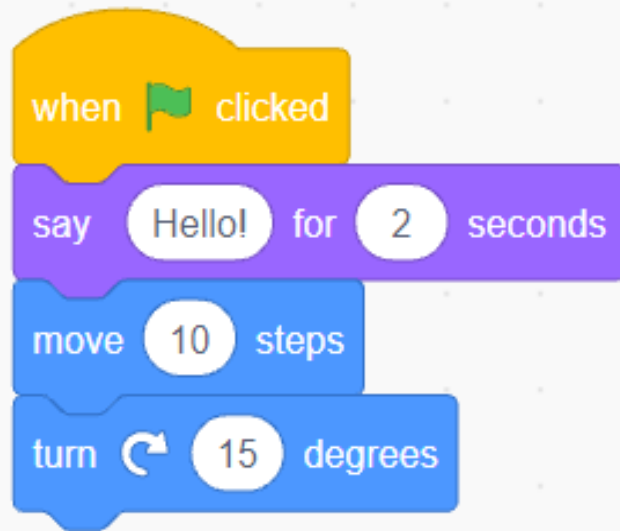
1. Counter controlled iteration
2. Condition controlled iteration



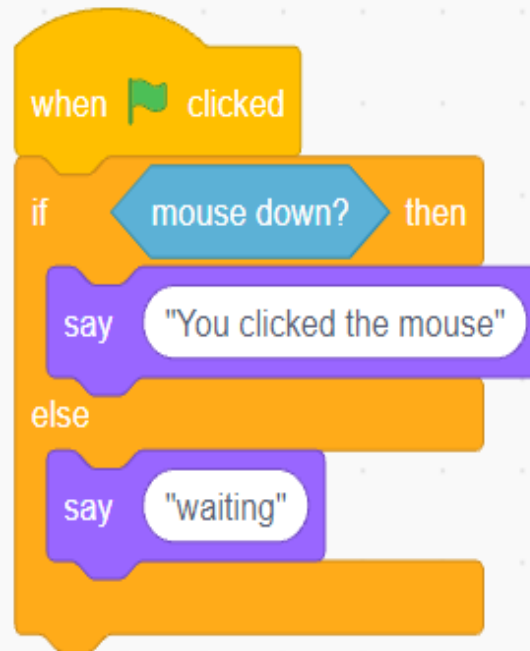
Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

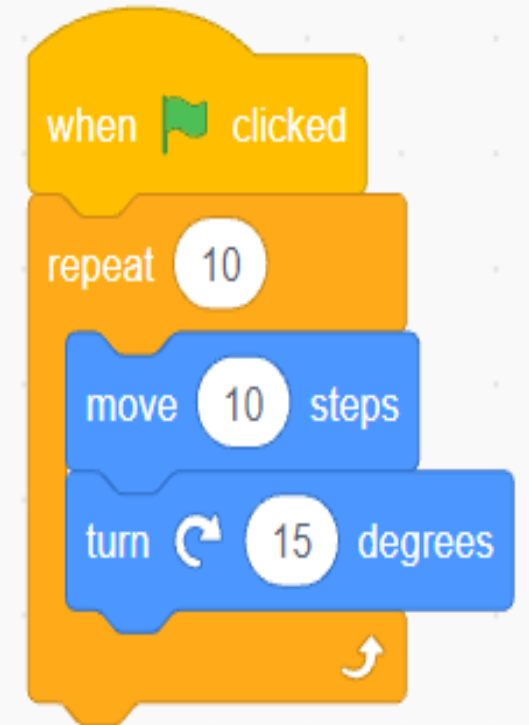
Sequence



Selection



Iteration



Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

What is a Variable in programming?

- A variable is a named container that stores a piece of data in your program.
- An input variable is a type of variable that takes information from the user.
- A variable can hold different types of data.
- In Scratch, a variable can be created using the variables block.

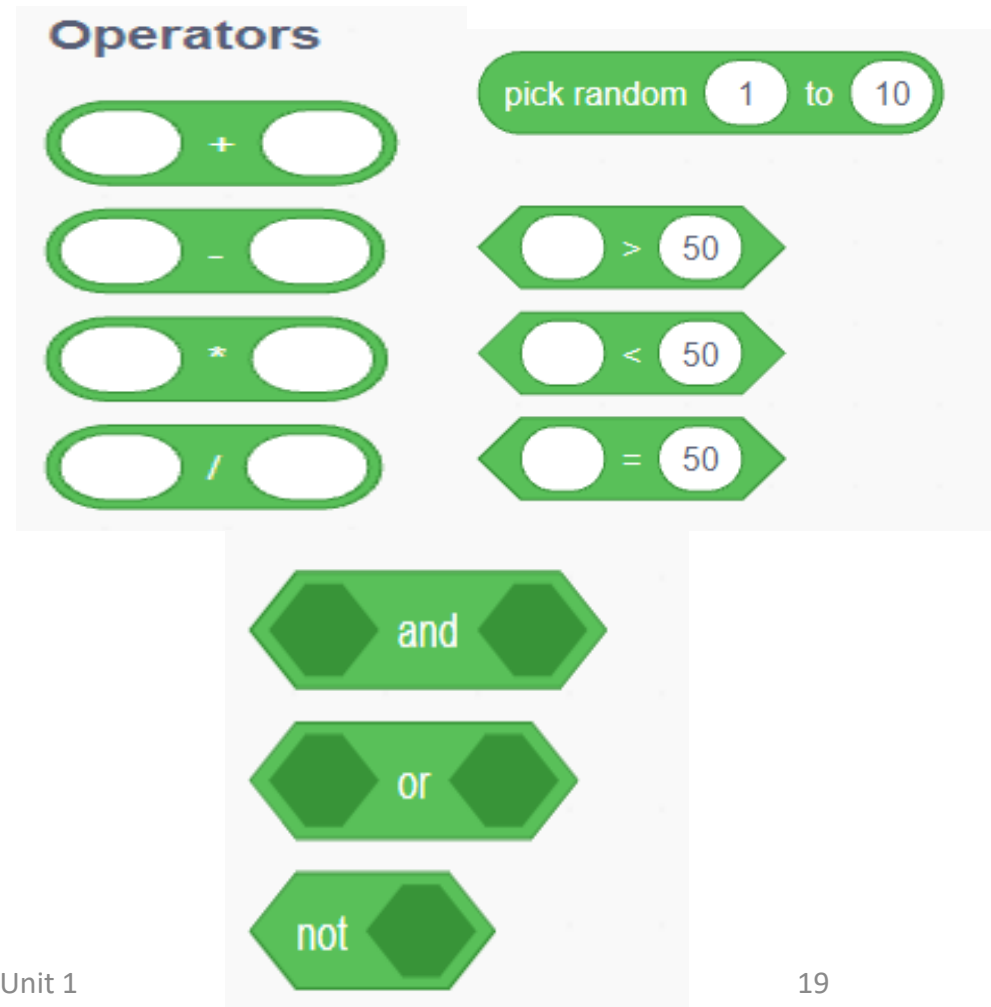


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

What are operators in programming?

- An operator is a symbol that tells the program to do a certain action on the data given.
- There are two types of operators:
 1. Arithmetic operators
 2. Comparison operators

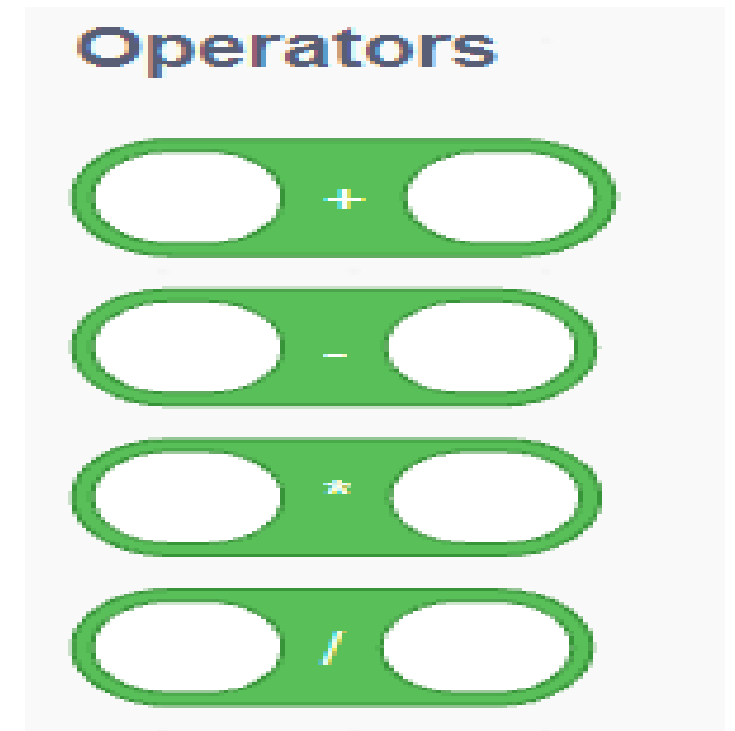


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

Arithmetic Operators

- Arithmetic operators are symbols used in programming to perform math calculations.
- These symbols perform calculations on the data.
- The main arithmetic operators are plus(+), minus(-), multiply(*) and divide(/)

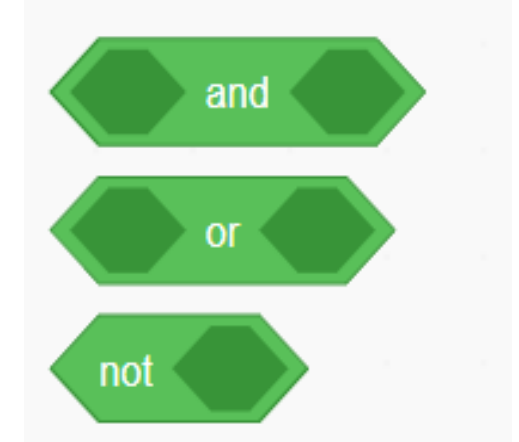


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

Comparison Operators

- Comparison operators are used to compare two values. They help the program make decisions by checking if something is true or false.
- There are three comparison operators in Scratch
 1. Less than
 2. Greater than
 3. Equals to

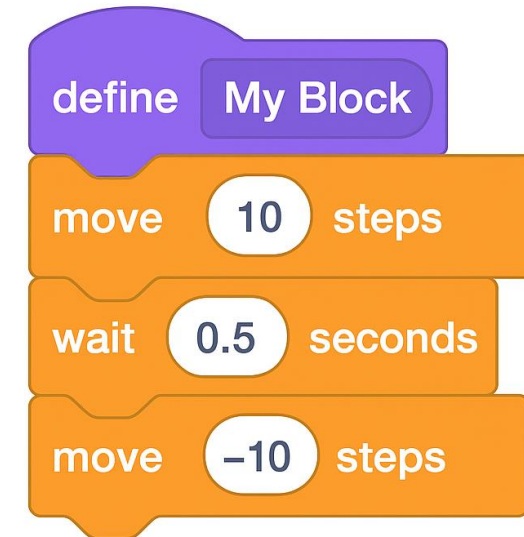


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

What is a procedure?

- A procedure is a small section of code that we can use multiple times in a program.
- Procedures save time for programmers, so they don't need to keep writing the same set of commands
- Procedures are written using the “My Blocks” category in Scratch



Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

What is an interaction?

- Interaction is when one part changes or affects any other part of program.
- For Example, in Scratch, If we have two sprites and when one sprite touches the other sprite, the other sprite disappears or performs any action.

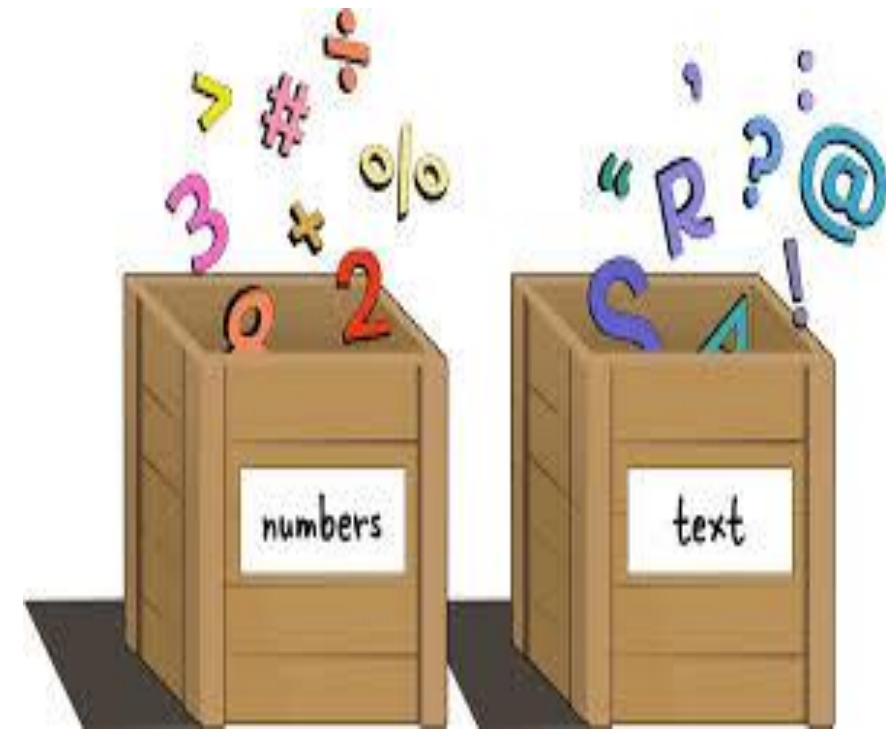


Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

Data Types

- A data type is the kind of data a variable holds—like numbers, words, or true/false values.
- Data types tell the computer what kind of information is being used or stored in a program.
- There are three main data types in Scratch
 1. Integer: stores numeric values such as whole numbers
 2. Character: stores individual letters, characters or symbols
 3. String: combination of characters, letters, numbers or symbols



Unit:1 Computational Thinking and Programming

1.2 Programming Constructs

Data Types

Activity: Guess the data type

Unplugged activity 3

The Scratch blocks below all require an input of certain data types.
For each block, discuss with your partner which data type the block uses.

- 1
- 2
- 3
- 4
- 5

Unit:1 Computational Thinking and Programming

1.3 Sub-Routines

What is a Sub-Routine?

- A sub-routine is a section of code that is separate from the main code and performs a specific task.
- It can be used multiple times in an algorithm or code
- It makes the code easier to understand
- It reduces the repetition of code

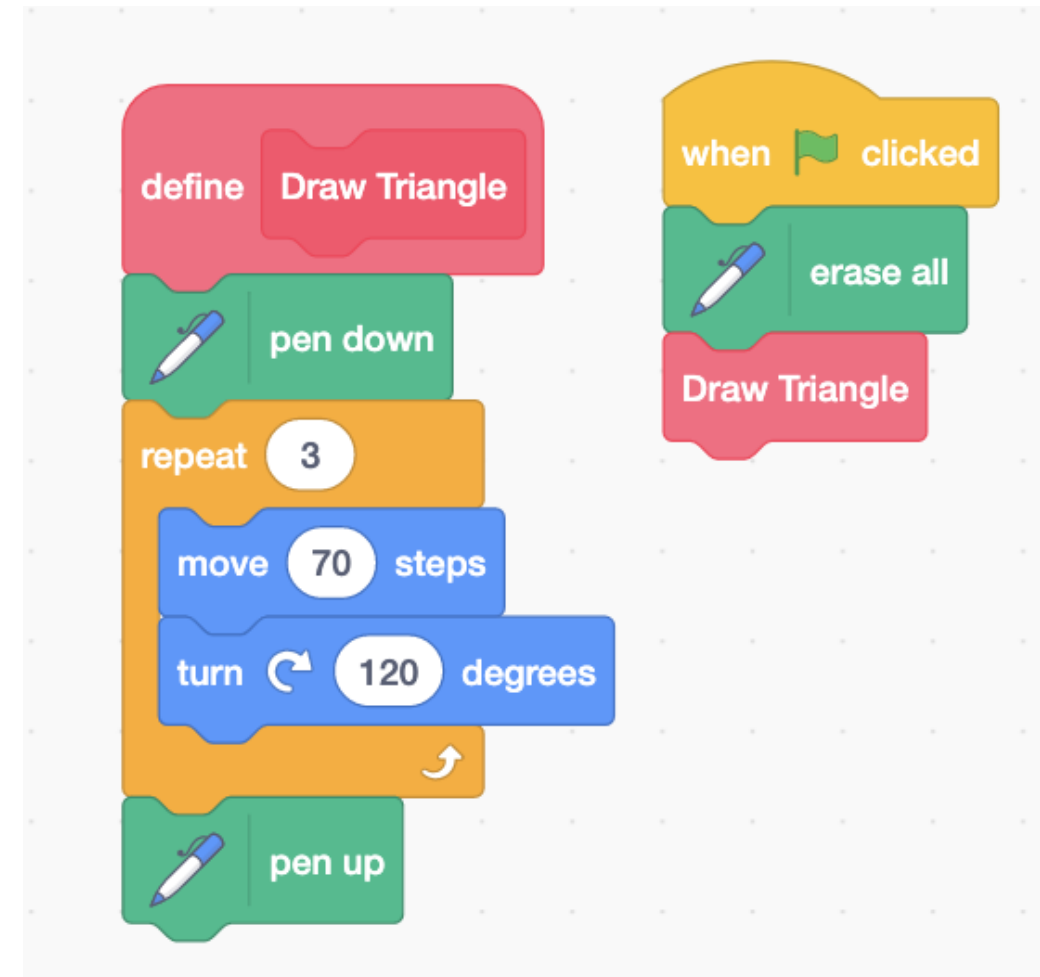


Unit:1 Computational Thinking and Programming

1.3 Sub-Routines

Defining a Sub-Routine

- To use a sub routine in an algorithm, we need to first define it.
- To define a sub-routine, we give it a name and instructions in it
- To use a sub-routine in a code, we call it.
- A sub-routine can be called many times in a code



Unit:1 Computational Thinking and Programming

1.4 Planning Programs

- Programmers need to plan their programs before writing a code.
- The task needs to be analyzed for a logical solution
- Decomposition of the problem is done in the planning stage
- A complex task can be broken down into small, manageable tasks.
- Project plan can include mind maps, flowcharts, sketches and annotations etc.
- After the plan is made, the programmers start working on a prototype

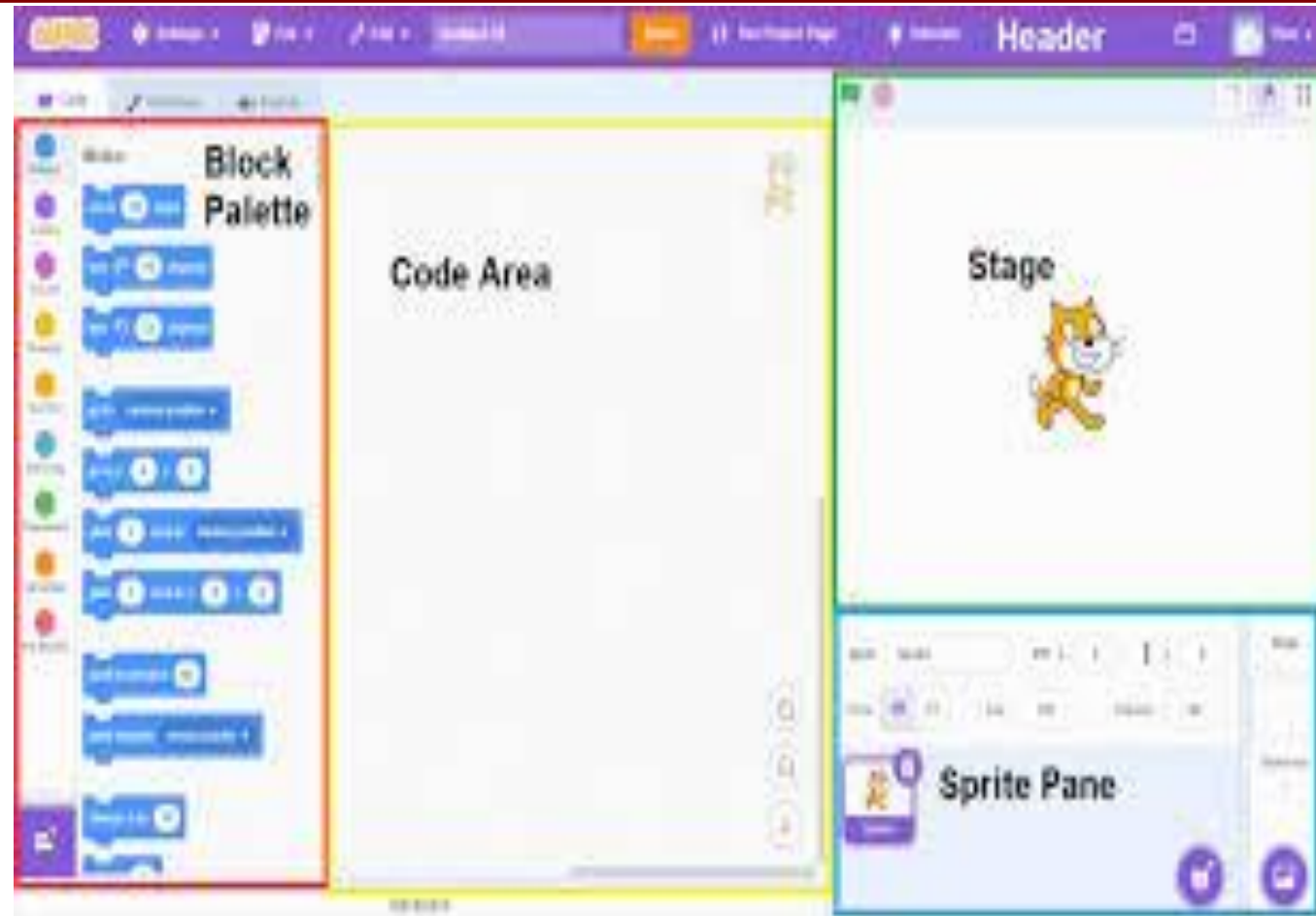


Unit:1 Computational Thinking and Programming

1.4 Planning Programs

What is an interface?

- Interface is what the user sees when they run the program. It is important because it tells the user what they need to do.
- The user interacts with the computer through the interface



Unit:1 Computational Thinking and Programming

1.4 Planning Programs

What is prompt?

- Prompt is the message on the screen that shows the program is waiting for input.
- In Scratch, a prompt is a way to ask the user for input during a program.
- When you use a prompt, the program shows a message asking the user a question, then waits for them to type an answer.



Unit:1 Computational Thinking and Programming

1.5 Evaluating and Testing Programs

- Program evaluation means checking how well your program works after you've finished making it.
- It's like testing and reviewing your project to make sure:
 1. It does what it's supposed to do
 2. It is easy to use
 3. There are no bugs or errors
 4. It could be improved in any way



Unit:1 Computational Thinking and Programming

1.5 Evaluating and Testing Programs

Why is program evaluation required?

When Do We Evaluate?

You evaluate your program:

- After building it
- Before sharing or submitting it
- Sometimes even while testing small parts during coding

What Do We Look at During Evaluation?

1. **Functionality** – Does the program work correctly?
2. **Usability** – Is it easy for others to use?
3. **Design** – Does the interface look good and make sense?
4. **Efficiency** – Is it simple and clear, not too messy or repetitive?



EVALUATE

Unit:1 Computational Thinking and Programming

1.5 Evaluating and Testing Programs

What are success criteria?

Success criteria are the clear steps or goals that show a program has been done correctly and completely.

Success criteria tell us what a good project should include or be able to do.

It tells us what a program should do to be successful.



Unit:1 Computational Thinking and Programming

1.5 Evaluating and Testing Programs Code Improvement & Testing Programs

- Code improvement means making your code better. Code improvement is making your code cleaner, faster, or easier to understand.
- Program testing means running your code to find and fix any mistakes or problems.

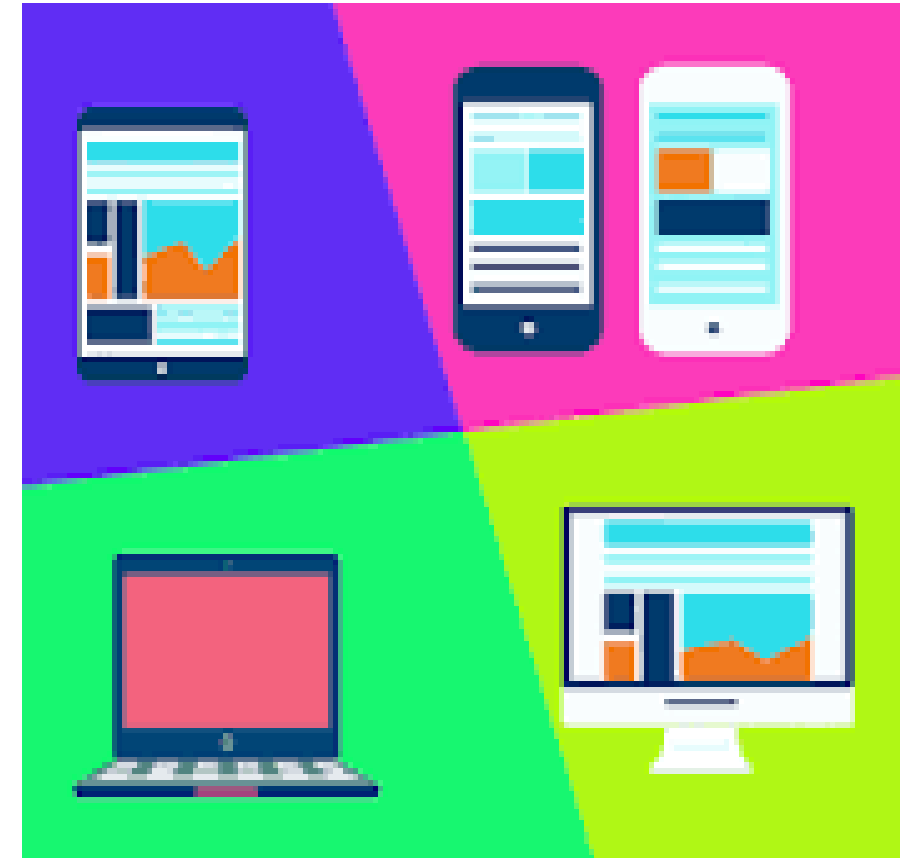


Unit:1 Computational Thinking and Programming

1.6 Using Variables with a physical device

What are computing devices?

- Computing devices are electronic machines or tools that can receive data (input), process it, store it, and produce results (output) using a set of instructions called a program.
- Examples of Computing Devices are desktop computers, Laptops, Tablets, Smartphones, Smartwatches, Game consoles, Robots and Smart TVs.
- A process is a task or action that a computing device is doing by following instructions in a program.



Unit:1 Computational Thinking and Programming

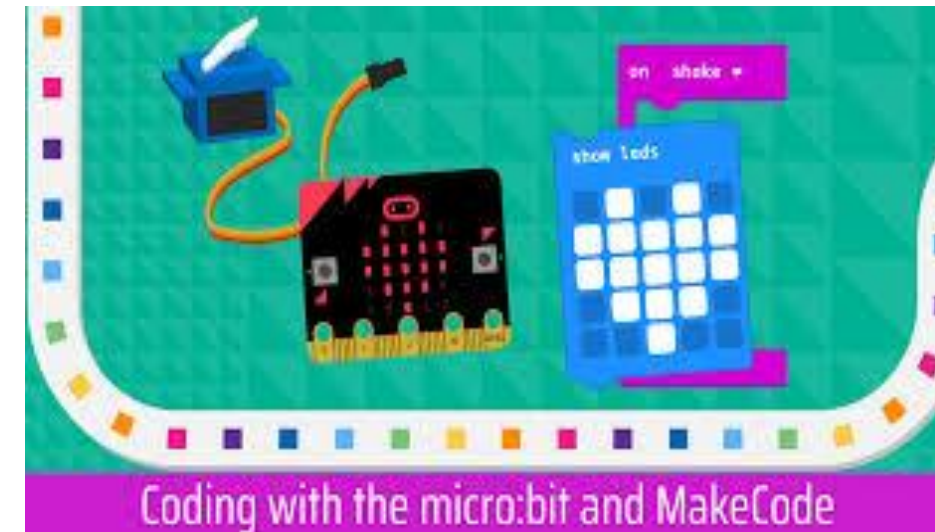
1.6 Using Variables with a physical device

What are Makecode and Microbit?

MakeCode is a free, online coding platform created by Microsoft. It lets you write code using blocks or JavaScript to control devices like the micro:bit. It's great for beginners and fun to use.

A micro:bit is a small programmable computer (also called a microcontroller).

It has LED lights, Buttons, Sensors (like temperature and motion) and can connect to other devices



Unit:1 Computational Thinking and Programming

End of unit Exit ticket

Planning Flowcharts

Q1. What is the purpose of a flowchart in programming?

Programming Constructs

Q2. Match each construct with its meaning:

Construct	Description
A. Sequence	Making a choice
B. Selection	Doing steps in a specific order
C. Repetition	Doing something again



Unit:1 Computational Thinking and Programming

End of unit Exit ticket

Sub-Routines

Q3. What is a sub-routine, and why is it useful?

Planning Programs

Q4. Circle all tools you can use to plan a program:

Flowchart, Python, Pseudocode, Spreadsheet,Storyboard

Evaluating and Testing Programs

Q5. Why is it important to test a program before using it?

Unit:1 Computational Thinking and Programming

End of unit Exit ticket

Variables & Devices

Q6. You're using a microcontroller to measure temperature.

Which variable name is best?

Circle one:

- a) X1**
- b) tempSensorReading**
- c) helloWorld**
- d) dog**