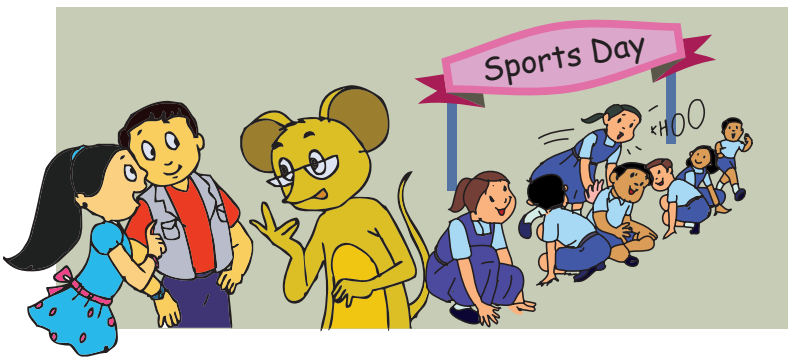


Advanced Scratch Programming



In this lesson you will learn:
How to capture the input from the user.
How to write programs using variables and lists.



Jyoti and Tejas are planning to create a game called “Logic Bingo” using Scratch programming language. They start discussing about it with Moz.

Moz: Why do you call it “Logic Bingo”?

Jyoti: In the game there are some

conditions. The players have to find these conditions using logical reasoning before getting to the answer. There are only four chances to win the game. So we named the game “Logic Bingo”.

Moz: So what is this game?

Tejas: For sports day in our school, each student can participate in two games from a list of six games.

Team games	Individual games
Foot ball Basket ball Kho kho	Athletics Swimming Gymnastics

Jyoti: Rules for combinations of games are as follows:

- Participation in two team games is not allowed (invalid choice).
- Participation in two individual games is allowed (valid choice).
- Participation in a team game and an individual game is valid.

Tejas: We want to convert this into a Scratch project. Students have to find the correct combination of games in four chances.

Jyoti: We will not reveal the rules for valid combinations to students. We will display only the names of the games. They have to guess the rules and find a valid choice.

Moz: Good project. So how do you start?

Tejas: The main steps in our project are:

Step 1: Display the games.

Step 2: Repeat the following sequence four times.

- i. Ask the student to enter the choice for first game. Receive the answer.
- ii. Ask the student to enter the choice for second game. Receive the answer.
- iii. Determine whether the two games are a valid combination.
- iv. Display whether the combination is valid or invalid.
- v. Keep score of valid choices.

Step 3: Display the final score.

Jyoti: We do not know the instructions for sub-steps in Step 2.

Moz: Let us start with sub-steps i and ii of Step 2. This is called receiving input from the keyboard.

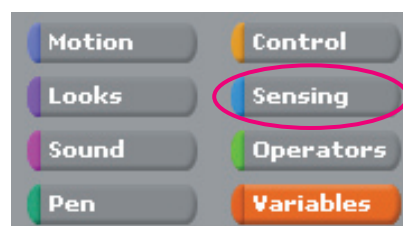
Receiving input from the keyboard

Moz: In which block do you find instructions to sense a mouse click?

Tejas (thinks): Sensing. Oh! So words entered using the keyboard is “sense the keyboard input”. Isn't it?

Moz: Correct.

Jyoti: Let us check Sensing.

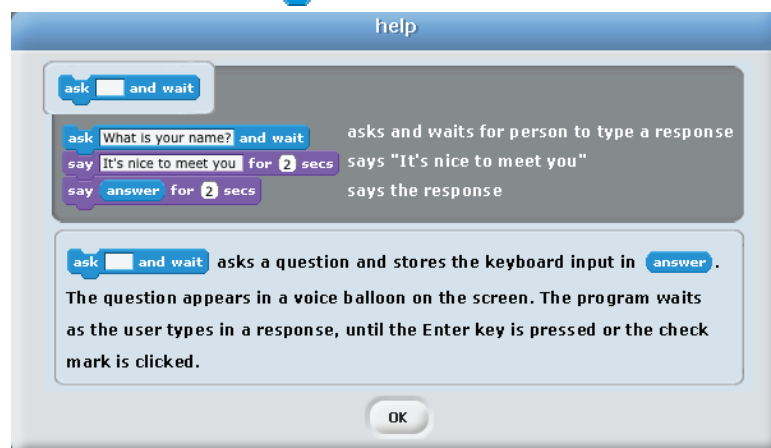


Look at this instruction



Moz: To learn how to use this instruction, right click on the instruction to get help.

Tejas right clicks on the instruction **ask** **and wait** and gets help for it.



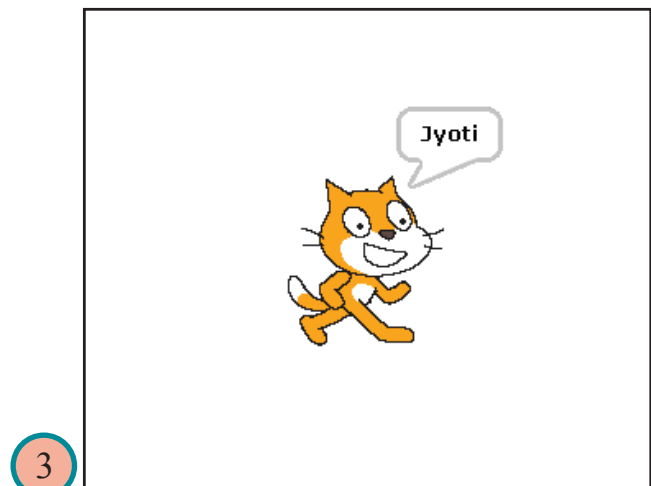
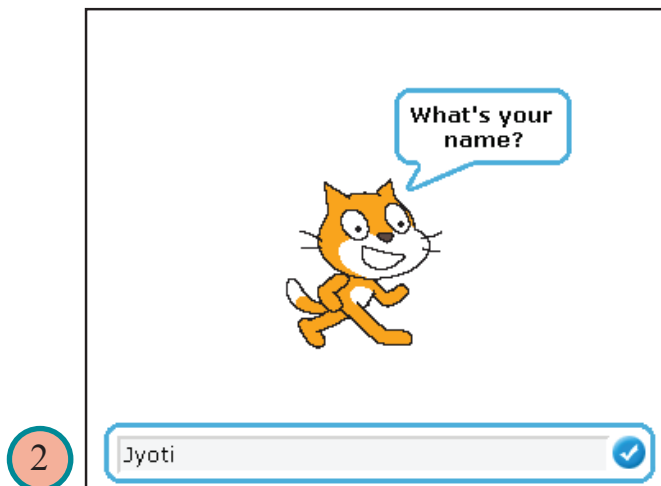
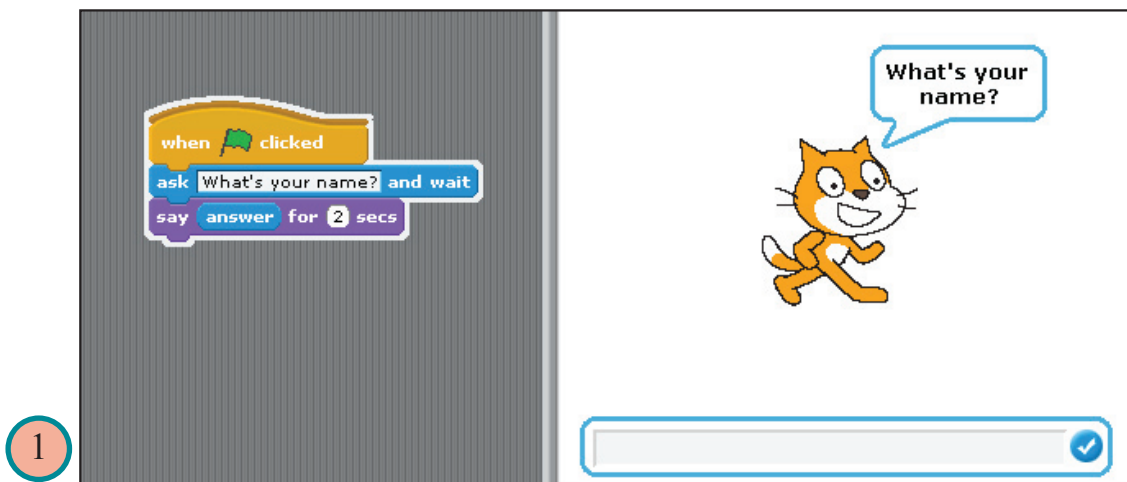
Using Help option:

To learn about the functionality of an instruction, right click on the instruction. This gives you a help option. Help option displays how to use the instruction with an example.

SKILLS

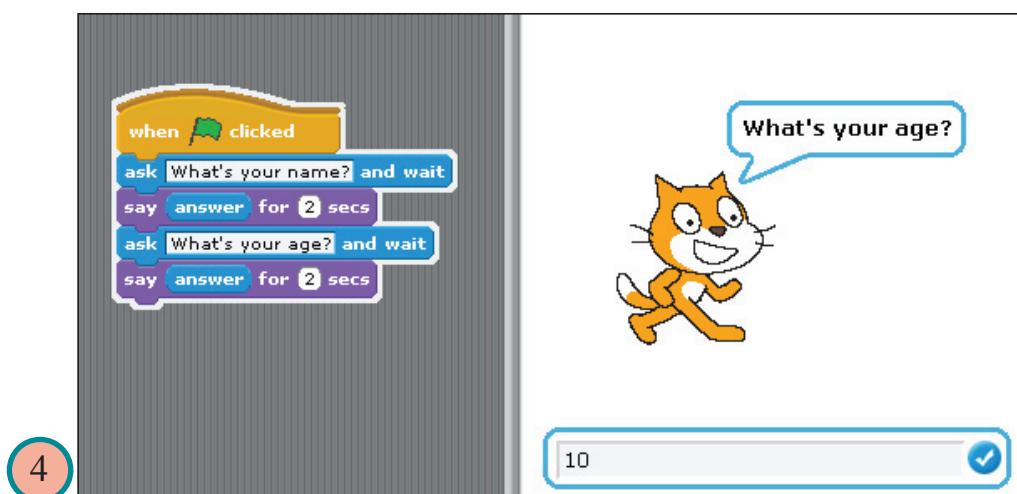
Moz: Build a small block using the Sensing instructions and execute it.

Sensing input and its execution steps



Tejas: This is good. We can ask a question and also capture the answer given by the user.

Jyoti: Let us enter another question using the ask instruction.



Keyboard input

- *ask and wait* prompts users to type input using the keyboard.
- *answer* stores the keyboard input.

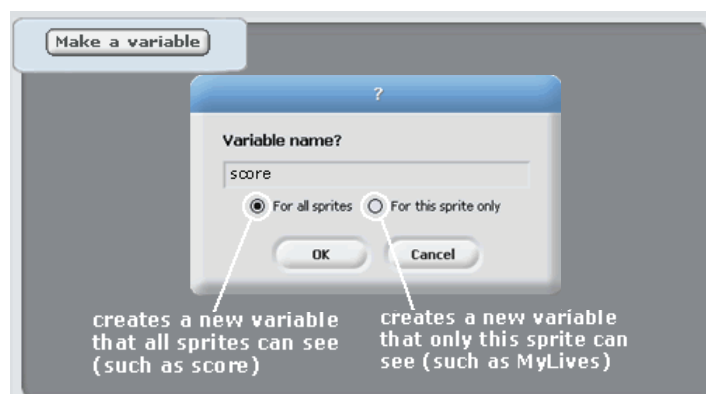


Creation of Variables

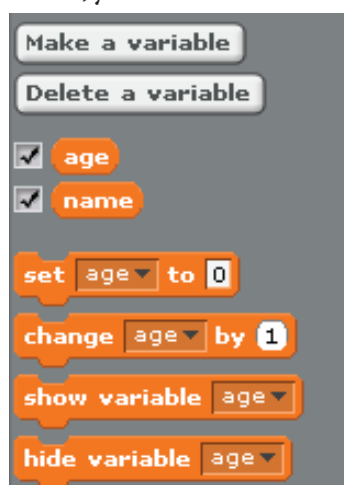
Tejas: What happens to the input entered for the first question. Is it lost?

Jyoti: I wish the computer can save the answer somewhere.

Moz: Sure it can. Check out the **Variables block**.



Tejas and Jyoti create Variables called *name* and *age*.



Variables

The variable blocks allow you to create variables and use them in a program. The variables can store numbers or **strings** (sequence of characters). A variable can be created such that:

- Only one of the Sprites can use it or all Sprites can use the variable.

Info



Letters, words or sentences are called strings in Computer language.

Assigning values to a Variable

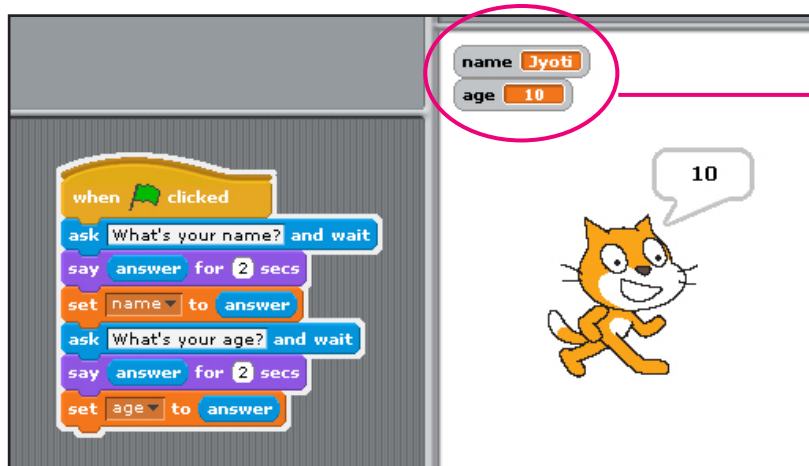
Tejas: Can we store a word in this variable?

Moz: Yes. When a word is saved in a variable, it is called a **string variable**.

Jyoti: I want to save answer in the string variable *name*.

Moz: Check out the *help* for *answer* under the Sensing block. If you want to save the current answer, you can store it in a variable.

Moz: Scratch allows you to save answer in a variable that you have already created.

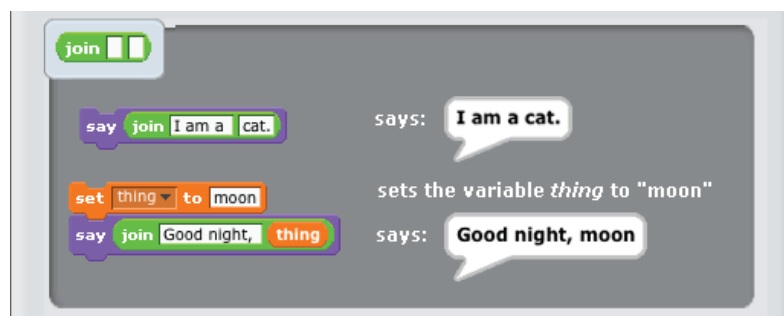


The variables are set to the *answer* entered by the user.

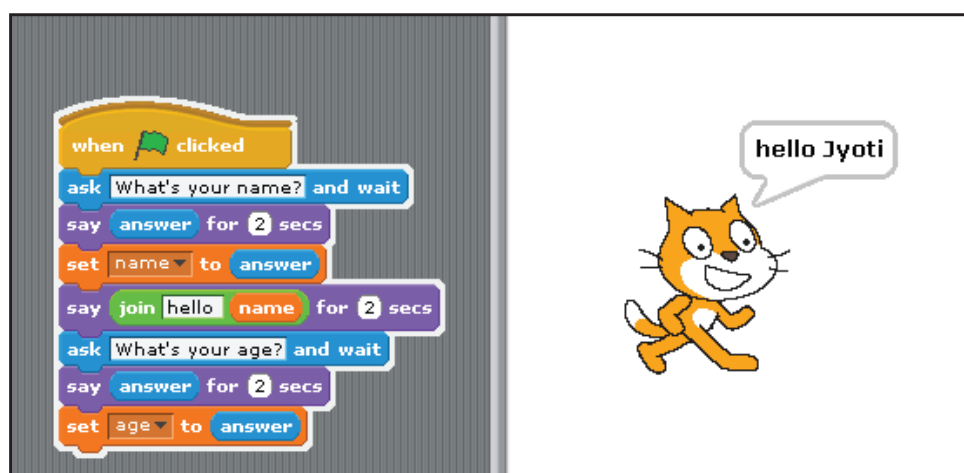
Operators (A symbol that represents a specific action. For example, a plus sign (+) is an operator that represents addition.)

Tejas: Suppose the name typed is “Jyoti” then I want to say “Hello Jyoti”. How do I get this?

Moz: You want to join two words and display it. **Join** is a string operator. Check out these instructions in **Operators block**.



Jyoti checks the help for string operator *join*. Then she adds the *join* instruction to her scratch program and executes it.



Tejas: How do we compare strings? We will need it for our project, to compare the user's answer with ours.

Moz: How do you compare if two numbers are equal?

Jyoti: Using '=' (equal to) operator.

Moz: You can compare strings for equality also, just like numbers. Where do you find the comparison instruction?

Tejas: In Control block.

Tejas and Jyoti write the following blocks to see how operators and control statements can be used.



Script for the Cat Sprite



Script for the Football Sprite



Stage after execution of program

Conditional statements:

- *if* and *if-else* in the control block are used to check for a condition.
- If the condition is true the set of instructions after *if* are executed.
- Otherwise, the set of instructions after *else* are executed.

Moz: Explore the other string operators too. You may get some more ideas to show comparisons.

Tejas and Jyoti explore the following string operators:

join [] []

Concatenates (combines) strings.

length of []

Reports the number of letters in a string.

letter [] **of** []

Reports the letter at the specified position in a string.

Logic Bingo by Tejas:

Tejas: Let us create a game called “Logic Bingo”. There are six Sprites for the six games. Cat Sprite will ask a question and capture the response. An Instruction Sprite will give the instructions of the game.



Moz: Have you decided on the variables that you will require for the game?

Jyoti: Since the player enters two games of his or her choice, we need two variables, *game1* and *game2*, to save the choices.

Moz: Good. What else do you need?

Tejas: We can decide if a game combination is valid or invalid by comparing the types of the two games. So each time the player enters a choice, we should also save the type of the game. Later, we can do the comparison and decide if the choices are valid or invalid. So we need two variables, *team* and *individual*.

Moz: Good. So next what do you do?

Tejas: Import the sprites and position them on the stage.

Jyoti: We have to also create the variables.

Moz: Good. Summarize the steps.

Steps for creating Variables

Step 1: Click on Make a variable to create a new variable.

Step 2: Type in the name of the variable in the pop-up box and click OK. The new variable *game1* is created.

Steps for creating a game Logic Bingo

1. Import Sprites and position them on Stage:

Select Sprites from the available Sprites or paint the Sprites required for the game.

Six Sprites for six games and *Cat* Sprite as referee of the game. One Sprite which has instructions of the game.



2. Create the required variables:

Four variables are required. Two to save the names of games and two for the types of games.

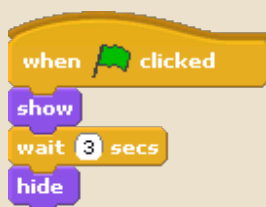
Moz: What is the next step? How does the program start?

Jyoti: The program starts as follows.

How does the game start?

3. Hide the Sprites and show the instructions of playing the game.

4. Hide the instructions and show the Sprites.



Script for
Instruction
Sprite



Script for both
Cat Sprite and the
game Sprites

LOGIC BINGO
You can participate in any 2
games. Guess the correct
combination and enter your 2
choices of games.

Stage after running the Script

Moz: How do you take responses from the player?

Tejas: First, the Referee Sprite asks a question. Next, the Player responds. Then the response is saved in *game1*.

Jyoti: We have to also save the type of game. We can use Broadcast. Referee Sprite can Broadcast the name of the game. Then the corresponding game sprite will receive and save the type of game in *choice1*.

Tejas: Let us use *join* while broadcasting. For example if the first game is football then we can join it with '1'. So the Broadcast will go as *football1*.



Moz: Very good.

Tejas: If the choice of game is team game, then *choice1* = team. If the choice is individual type of game then *choice1* = individual.

Jyoti: We can again use *join* to display the type of game.

say join The first game selected is choice1 for 2 secs

Moz: Good.

Jyoti: We can use the same sequence and logic for the second choice of game too.

Moz: Correct. Go ahead and write the Scripts for referee and game Sprites.

5. Referee - Cat Sprite block:

Referee gives an instruction “Enter the name of the game you want to play on sports day”

Player responds by entering the name of the game.

Save the name of the game in the variable game1.

Broadcast game1.

6. Game - Sprite:

The game that has been chosen receives the Broadcast.

Type of game is also saved by the chosen game Sprite in a variable choice1.

7. Referee - Cat Sprite:

Receives the Broadcast and says the type of game (choice1) so that the player knows type of the first game that he/she chose to play on sports day.

```
ask Select the game that you want to play on sports day and wait
set game1 to answer
say game1 for 2 secs
broadcast join game1 1 and wait
say join The first Game selected is a choice1 for 2 secs
```

Scripts for referee Sprite for choice1



Stage for choice1 of the player
for the game **Logic Bingo**

```
when I receive swimming1
say I am game1.
set choice1 to individual game
```

Individual game Sprite
Script, if choice1 is
swimming

```
when I receive football1
say I am game1.
set choice1 to team game
```

Team game Sprite
Scripts, if choice1 is
football

Moz: Now that you have one choice, what should be your program for the next one?

Tejas: Program for *choice2* will be the same as for *choice1*, except that player's response will be saved in variable *game2*.

```
ask Select one more game that you want to play on sports day and wait
set game2 to answer
say game2 for 2 secs
broadcast join game2 2 and wait
say join The second game selected is a choice2 for 2 secs
```

Scripts for referee Sprite for choice2



Stage of the player for choice2 response

```
when I receive swimming2
say I am game2.
set choice2 to individual game
```

Individual game Sprite Script,
if choice2 is swimming

```
when I receive football2
say I am game2.
set choice2 to team game
```

Team game Sprite Scripts,
if choice2 is football

Jyoti: Now we have to build the program for comparing *choice1* and *choice2*.

Moz: What would the referee Sprite do now?

Tejas: Referee Sprite compares the combination of selected games and says whether it is valid or invalid.

Moz: Good. Suppose someone enters their *choice1* again in *choice2*?

Jyoti: We can handle that easily. We will first write the logic to check duplicate. The full comparison logic is as follows:

1. If the same game is selected twice then the combination is not valid.
If $\text{game1} = \text{game2}$, then say "Duplicate selection of games – invalid combination"
2. If the two games selected are team games then the combination is not valid.
If $\text{choice1} = \text{team game}$ and
If $\text{choice2} = \text{team game}$, say "invalid game combination".
3. Rest of the combinations are correct.
 $\text{choice1} = \text{team}$, $\text{choice2} = \text{individual}$, then valid combination.
 $\text{choice1} = \text{individual}$, $\text{choice2} = \text{individual}$, then valid combination.

Moz: That is a very good comparison sequence. First you are making sure there are no duplicates. Next, since there is only one more invalid combination, you are checking for it. Rest of the combinations are valid.

Tejas: We will now complete the game.

Final stage of Logic Bingo

Referee Sprite script to display the following choices:

```

if game1 = game2
say You have selected the same game twice. Try again
endif
if choice1 = team
if choice2 = team
say Your choice of games is not valid. Try again.
endif
else
say Your choice of games is valid. Go ahead and play the game. All the best.
endif

```



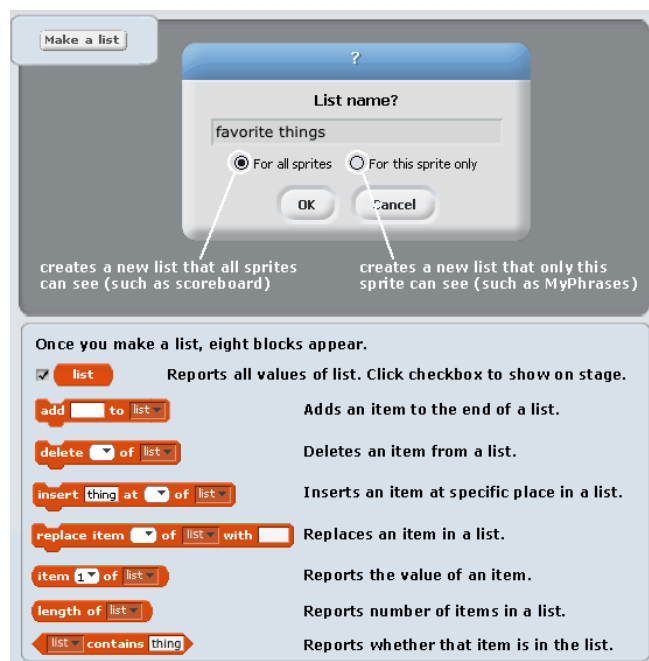
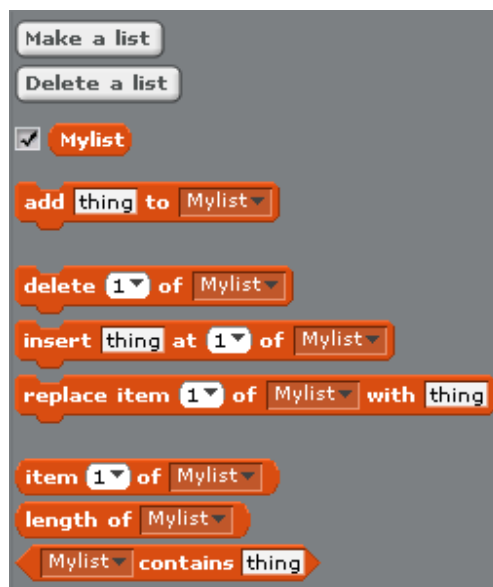
Moz: Have you noticed there is one more option under the Variables block: **Make a list** ?

Tejas: Is it something like the lists we make for purchasing things from the grocery store?

Moz: Yes, it is similar to lists that we normally use.

Jyoti: We will click on it and make a new list. Let us name it as *Mylist*.

Moz: When you create a list you get eight different blocks under list. If you click on the help for **Make a list** you can get the functions of all these eight blocks.



Lists (arrays)

- Use list blocks to store a list of numbers and strings.

Tejas: Let us write a program to get four items from the user, which can be taken while going on a picnic.

Tejas and Jyoti explore the various commands for list and write the following program.



There are some blocks in Scratch such as **x position**, **mouse down?**, **variable** which are designed to fit in the input area of other blocks. These blocks are called **Reporters**. Reporters with rounded ends (such as **variable**, **mouse x**) report numbers or strings, and fit in blocks with rounded or rectangular holes (such as **say Hello!**). Reporters with pointed ends (such as **mouse down?**) report values (true or false) and fit inside blocks with pointed or rectangular holes (such as **wait until** or **say Hello!**).

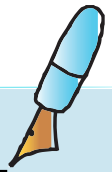
Info

Moz: Good. Today you have learned to take input from user, use list and variables. Write more programs and have fun! Chin Chinaki...

Learning Outcome

After you have studied this lesson, you will be able to:

- Write programs using commands under the Operators (like, join) and Variables (variable, list) blocks.
- Provide input through keyboard for a Scratch program.



1. Gargi has written a program to capture the input from the user. She has created a variable named *Input*. What will be the final value of this variable after the execution of the program?

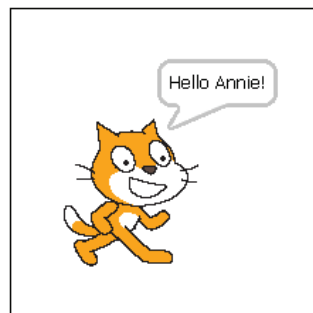
- Hello!
- User name
- User's friend's name
- answer



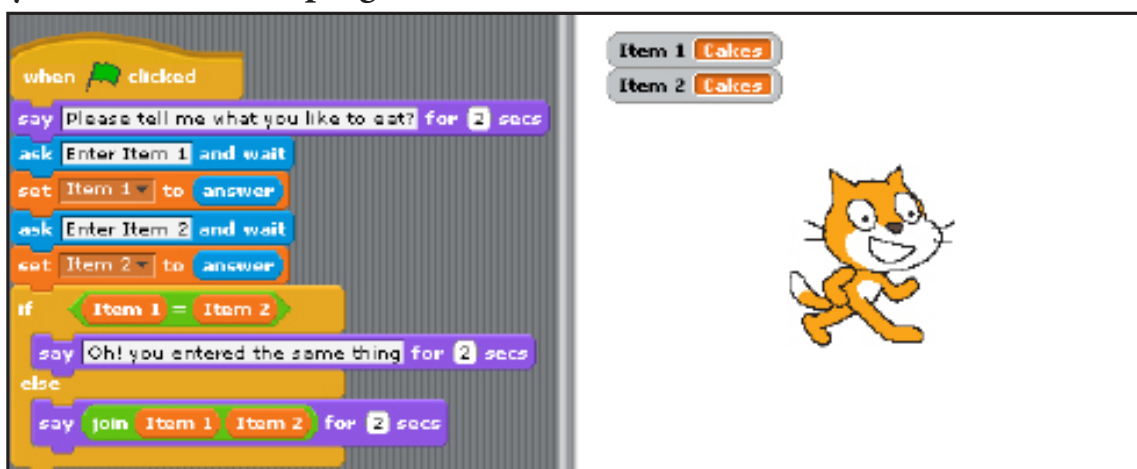
2. The following is a small program. The last line of the program is missing. When executed the user gave the name as Annie. If you want the sprite to say “Hello Annie”, what should be the last line of the program?



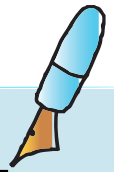
- say Hello Annie for 2 secs
- say join Hello answer for 2 secs
- say Hello + Annie for 2 secs



3. A small program which compares two strings is given below. The user has entered the name of two items he likes to eat. Can you tell what the Sprite will say at the end of this program?



- Oh! You entered the same thing
- Cakes Cakes
- Enter Item 2
- Enter Item 1



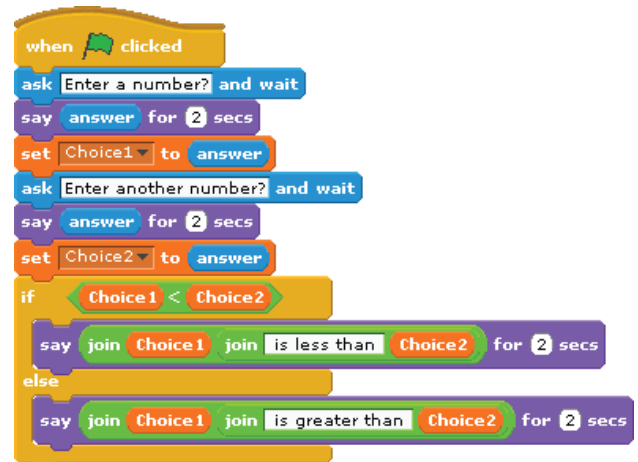
4. In the following program there are three variables: Choice1, Choice2 and Game. When the program runs fully, what will be the final value of the variable: Game?

- Basketball,
- 0
- Football
- 2



5. Here is a program to compare numbers. What will the Sprite say when the numbers entered by the user are 250 and 500. Circle the correct option.

- Choice1 is less than Choice2
- 250 is less than 500
- Choice1 is greater than Choice2
- 500 is greater than 250



6. Here is a program which calculates the length of the name entered by a user. But the instructions are jumbled up. Rearrange the instructions by numbering them.

say length of answer for 2 secs

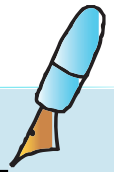
say Hello! for 2 secs

set Name to answer

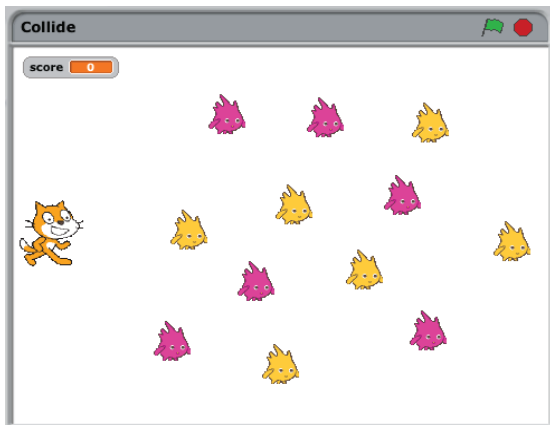
ask 'Whats your name?' and wait

when clicked

1



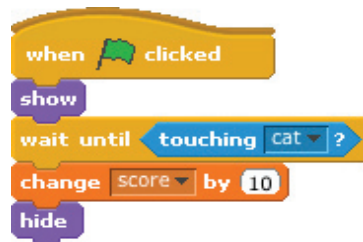
7. The stage of the game *Collide* (which has Cat and Gobos as sprites), and the scripts for the cat, yellow Gobos and the pink Gobos are given below.



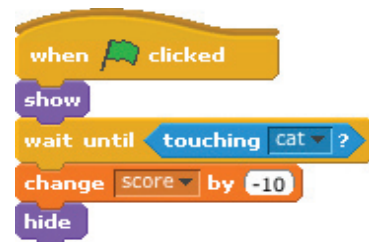
Cat Script



Script for yellow Gobo

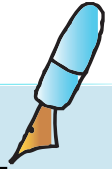


Script for pink Gobo



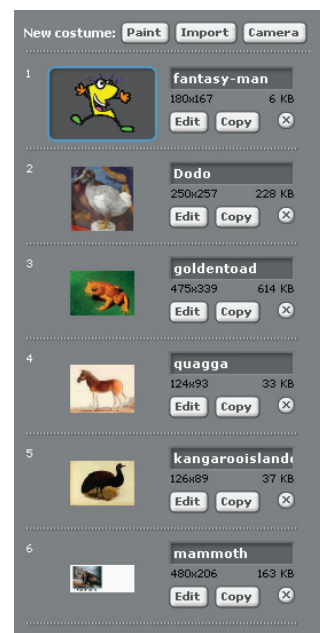
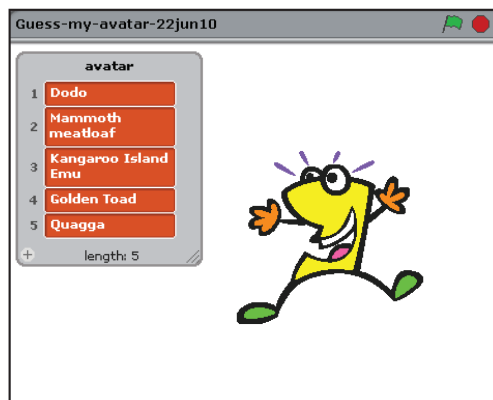
Using the scripts given above for the game *Collide* answer the following:

- initializes the _____ of the Sprite and the variable _____.
- What is the Goal of the game?
 Help the Cat _____ the mine field of Gobos with the _____.
- What are the Rules of the game?
 Collect _____ Gobos to earn points, Avoid _____ Gobos to lose points.
- What happens when you earn points?
 Maximize _____ while playing the game.



8. You are given a Sprite “fantasy man”, costumes for the Sprite which are “extinct birds and animals” and a list “avatar” with the names of the costumes and jumbled up instructions for the Sprite. Arrange the instructions into a Scratch block by:

- Using the Sprite and costumes given.
- Arranging the jumbled up Scripts such that the following actions are repeated:
 - The fantasy man asks the user to guess the next avatar that he would change to from the list of extinct birds and animals given on the stage.
 - The user guesses the next avatar and enters the corresponding number from the list on the stage.
 - Fantasy man changes to a random costume.
 - If the guess matches the costume that the fantasy man changes to then he makes a positive statement, else provides the name of the costume that he changed to.



when clicked 1

say join Sorry! The magician has changed me to item next-avatar of avatar for 2 secs 2

set next-avatar to pick random 1 to 10 3

switch to costume next-avatar

if answer = next-avatar 4

ask Guess and enter the number of what my next avatar will be? and wait 5

say join You are right.. I am currently item next-avatar of avatar for 2 secs 6

switch to costume fantasy-man

say Look at the list and guess what will be my next avatar? 7

Scratch block of fantasy man

1

repeat 10



ACTIVITY

Level V | Lesson 4

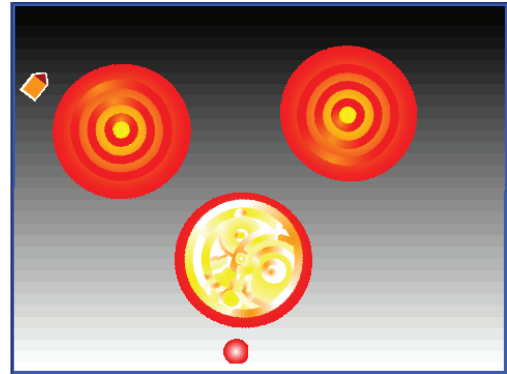
1. Open the following Scratch projects and do the activities.

i. **Bouncing Music Balls:** Use the arrow keys to move the small balls. When the small balls hit the big ball, music is produced. Follow these steps to start the project:

Scratch ---> Projects ---> Simulation ---> Bouncing Music Balls

Activities to do:




- Press H to read the help on how to play.
- Use arrow keys and + and - keys on the keyboard and observe what happens.

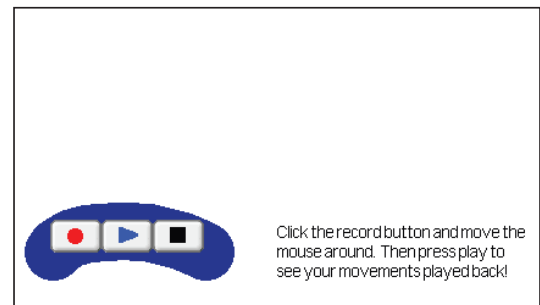


ii. **Mouse Recorder:** This project lets you record and play back movements of the mouse. It uses lists to store x and y positions of the mouse. Follow these steps to start the project:

Scratch ---> Projects ---> Interactive Art ---> Mouse Recorder

Activities to do:

- Press record button (). Move the mouse around to record the movements of the cursor. Press stop (). Press play () to see the recording.
- Explore the Scripts to see how list is used in the program.

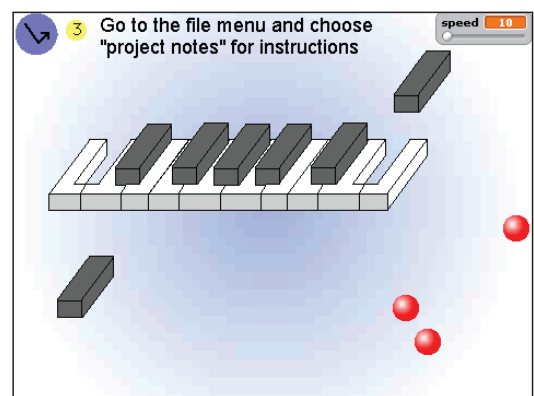


iii. **Piano Machine:** Balls bounce on the piano keys to make music. To start the project follow these steps:

Scratch ---> Projects ---> Music and Dance ---> Piano Machine

Activities to do:

- Click and drag any piano keys to a new position for different music.
- Click on the number icon to change the number of balls in motion.
- Change the program and create your own piano.





ACTIVITY

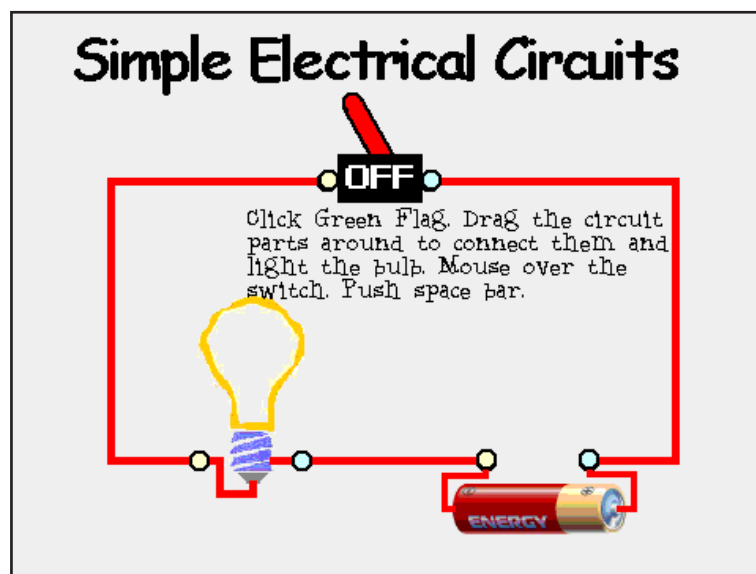
Level V | Lesson 4

iv. **Simple Electrical Circuits:** Place all the components in the circuit to light up the bulb.

Scratch ---> Projects ---> Simulations ---> SimpleElectricalCircuits

Activities to do:

- Position the circuit components and light up the bulb.
- Change the program to introduce another bulb and light up both the bulbs.





Project

Do project 4.1, 4.2, 4.3 from lesson 8.



Explore!

1. Explore what are boolean operators?
2. Find out the functions of commands under Operators block like , .

- This lesson is best taught in front of a computer running Scratch.
- The purpose of this lesson is to teach students how to use the instruction blocks related to conditional statements, variables, lists, operators and keyboard input, available in Scratch. Students are already familiar with this programming language and understand that they can make the computer do something using this activity.
- Begin by revising the functions of different instruction blocks already taught. You can ask them to write a small project to refresh their memory of what they already know. Tell the students that they will now learn to use additional instruction blocks to write a variety of interesting projects.
- Start Scratch and open an existing project (e.g. Examples --->Speak up ---> Global warming) that includes sensing and variable instructions. At this time, do not show the Script. Ask them to observe the animation. Now show the Script area and ask them to observe if they find any new instruction blocks, and of what colour. Ask them to which instruction block set do they belong. Let them guess its functions, if they can. To do this, you can use the strategy of think pair share, where students can partner with their classmate and do this activity.
- Now explain the functions of different sensing options using the example mentioned in the book. Right click on the different block and open the help screen and ask them to read their functions. Do it for a couple of instructions and ask the students to explore the remaining instruction blocks themselves. Similarly teach the functions of different variable options. Give a demonstration of the Script included in the lesson to show how to use the different instruction blocks.
- Emphasize that it is important to plan and organise the different activities you want to do using Scratch. Ask the students to write the Script for the Scratch project in their notebook. Now ask them to exchange it with their partner and program using Scratch for each other's activity.
- Summarize the lesson and give the students activities to practice.



Further Reading:

<http://scratch.mit.edu/>