GRADE 7

SECTION A (20 marks)

1. C) To repeat a set of instructions under specific conditions
2. C) Ultrasonic sensor
3. B) Acts as the brain of the robot
4. C) To convert electrical signals into motion
5. B) Breaking down problems into smaller parts
6. B) Microcontroller
7. B) To sort and search data efficiently
8. D) Conditional statement
9. B) To execute a block of code based on a condition
10. B) Finding and fixing errors in the code
11. B) Sensor
12. B) Computer software
13. B) Light intensity
14. C) Automation
15. C) To execute a set of instructions repeatedly
16. C) A 'for' loop can iterate a specific number of times
17. C) To store and manipulate values needed by a program
18. C) They allow for code reuse and organization
19. D) An input to a function
20. C) It enables the reuse of code across multiple programs

SECTION B (5 marks)

1. False
2. True

3. False
4. True
5. False

SECTION C (10 marks)

Question 1: Create an algorithm in Scratch where a sprite moves in a random pattern on the screen without crossing the edges. (5 marks)

Code:

- When green flag clicked
- Forever
- Move (10) steps
- Turn (pick random -180 to 180) degrees
- If on edge, bounce

Mark allocation:

- **Event block:** Start with the "When green flag clicked" block to begin the sprite's random movement when the program starts. **(1 mark)**
- **Control block:** Use the "forever" loop to continuously move the sprite in random directions. **(1 mark)**
- **Motion block:** Insert a "move (10) steps" block to make the sprite move forward. **(1 mark)**
- **Control block for randomness:** Use the "turn" block set to "turn (pick random -180 to 180) degrees" to change the sprite's direction randomly. This ensures the sprite turns in a random direction before each move. **(1 mark)**
- **Sensing block:** Add an "if on edge, bounce" block from the sensing category inside the forever loop. This block automatically turns the sprite if it touches the edge of the stage, preventing it from crossing the boundaries. **(1 mark)**

Question 2: Design a simple game where a sprite jumps over an obstacle when the spacebar is pressed. (5 marks)

Code:

- When green flag clicked

- Forever
- When space key pressed
- Change y by (10)
- Change y by (-10)

Mark allocation:

- **Event block -** Start game: Use the "When green flag clicked" block to start the game. This initialises game settings or prepares the game environment. **(1 mark)**
- **Control block -** Continuous loop: Implement a "Forever" loop to continuously check for user inputs or other game actions, setting up the environment for ongoing game mechanics. **(1 mark)**
- **Event block -** Jump trigger: Employ the "When space key pressed" block to trigger the jumping action when the player presses the spacebar. This is crucial for initiating the jump. **(1 mark)**
- **Motion block -** Ascend: Use "Change y by (10)" within the space key pressed event to make the sprite jump upwards. This step is critical for creating the visible jump effect. **(1 mark)**
- **Motion block -** Descend: Immediately follow the ascend action with "Change y by (-10)" to bring the sprite back down, simulating the gravity effect and completing the jump. **(1 mark)**

SECTION D (15 marks)

Question 1: In a Scratch project, program a robot sprite to navigate through a maze. The robot should turn right whenever it touches the maze walls. Write down the steps in order. (7 marks)

Code:

- When green flag clicked
- Forever
- Move (10) steps
- If <touching colour [maze wall colour]>
- Turn (90 degrees) right
- Wait (0.1) seconds

Mark Allocation:

- **Event block** - Start with the "When green flag clicked" block to initialise the robot's movement when the program begins. **(1 mark)**
- **Control block** - Use a "forever" loop to continuously monitor for wall collisions as the robot moves forward. **(1 mark)**
- **Motion block** - Inside the "forever" loop, insert a "move (10) steps" block to keep the robot moving forward continuously. **(1 mark)**
- **Sensing block** - Employ a "touching colour [colour of maze walls]" block to detect when the robot sprite touches the walls of the maze. **(1 mark)**
- **Control block** - Utilise an "if then" statement within the "forever" loop to check if the robot is touching the maze walls. **(1 mark)**
- **Motion Block** - Within the "if then" condition, include a "turn (90 degrees) right" block to make the robot turn right upon touching a wall. **(1 mark)**
- **Control Block** - To add realism and prevent the robot from getting stuck, optionally insert a "wait (0.1) seconds" block after the turn block to simulate a slight delay in turning, making the movement appear more fluid. **(1 mark)**

Question 2: In a Scratch project, create an animation where a sprite moves in a circle continuously when the 'S' key is pressed. Outline the blocks you would use to implement this feature. (8)

Code:

- When green flag clicked
- Go to x:0 y:0
- Set rotation style [left-right]
- When 'S' key pressed
- Forever
- Move (10) steps
- Turn (15 degrees) right
- Wait (0.01) seconds

Mark allocation:

- **Event block** - Start with the "When green flag clicked" block to initialise the program settings when the program begins. This block also centers the sprite and sets its rotation style. **(2 marks)**
- **Control block** - Use a "forever" loop under the "When 'S' key pressed" block to ensure the sprite continues its motion as long as the condition is active. **(1 mark)**
- **Motion block** - Insert a "move (10) steps" block inside the forever loop to keep the sprite moving continuously. **(1 mark)**
- **Motion block** - Include a "turn (15 degrees) right" block immediately after the move block to curve the movement into a circle. **(1 mark)**
- **Control block** - Add a "wait (0.1) seconds" block after the turn block to control the speed of movement, making the circular motion visible and smooth. **(1 mark)**
- **Event block** - Utilise a "When 'S' key pressed" event to start the circular motion, showing how to trigger actions with keyboard inputs. **(1 mark)**
- **Looks block** - Use "set rotation style [left-right]" to ensure the sprite maintains a consistent visual orientation during movement. **(1 mark)**

GRADE 8

SECTION A (20 marks)

1. (B) To provide a set of instructions for solving a problem
2. (A) Algorithm
3. (C) Identifying and fixing errors in the program
4. (B) To repeat a sequence of instructions
5. (B) Respectful and ethical behavior online
6. (C) If-Else
7. (C) They help the robot interact with its environment
8. (B) Boolean Logic
9. (C) Developing creative solutions to problems
10. (A) A method of adding features to a program one at a time
11. (B) By removing complexity and focusing on the big picture
12. (B) It allows different processes to run simultaneously
13. (B) An Integrated Development Environment (IDE)
14. (B) It provides energy for the robot to function
15. (B) It involves multiple disciplines to build effective solutions
16. (C) Comparison operator
17. (C) Repeating steps a specific number of times
18. (B) A user action that triggers a response
19. (B) Breaking a program into smaller, manageable parts
20. (C) The human-readable instructions that make up a program

SECTION B (5 marks)

1. False
2. False
3. True
4. False
5. False

SECTION C (10 marks)

Question 1: In a Scratch project, create a simple calculator that can add two numbers input by the user and display the result. (5)

Code:

- When green flag clicked
- Ask "Enter first number" and wait
- Set variable [FirstNumber] to (answer)
- Ask "Enter second number" and wait
- Set variable [Result] to ((FirstNumber) + (answer))
- Say (Result)

Mark allocation:

- **Event block:** Start with the "When green flag clicked" block to initialise the program. **(1 mark)**
- **Input block:** Use the "ask and wait" block to get the first number from the user. **(1 mark)**
- **Variable set:** Set the variable [FirstNumber] to the user's first input (answer). **(1 mark)**
- **Input and calculation:** Use another "ask and wait" block to get the second number and immediately calculate the sum by setting [Result] to ((FirstNumber) + (answer)). **(1 mark)**
- **Output:** Use the "say" block to display [Result], showing the sum of the two numbers. **(1 mark)**

Question 2: In a Scratch project, create a countdown timer that starts from 10 and counts down to zero, displaying each number and playing a sound when it reaches zero. (5)

Code:

- When green flag clicked
- Set variable [Countdown] to 10
- Repeat (10)
- Say (Countdown) for 1 second

- Change [Countdown] by -1
- Play sound [alarm]

Mark allocation:

- **Event block:** Use the "When green flag clicked" block to initialise the timer. This starts the countdown when the green flag is clicked. **(1 mark)**
- **Variable initialisation:** Set the [Countdown] variable to 10. This sets the starting number of the countdown. **(1 mark)**
- **Loop structure:** Use a "Repeat (10)" loop to manage the countdown. This loop will ensure the countdown operates correctly from 10 down to 1. **(1 mark)**
- **Display and decrement:** Within the loop, use "Say (Countdown) for 1 second" and "Change [Countdown] by -1" to display each number and decrease the countdown by 1 each time through the loop. **(1/2 mark each – 1 mark in total)**
- **Sound effect:** After the loop, use "Play sound [alarm]" to play a sound once the countdown reaches zero. This indicates the end of the timer. **(1 mark)**

SECTION D (15 marks)

Question 1: Your school is participating in a community clean-up and recycling drive. You have been tasked with designing a simple automated system using Scratch to sort items into two categories: 'Recyclables' and 'Non-Recyclables'. The sorter should ask for the type of material (e.g., plastic, paper, metal, or trash), and then place it in the correct category. (7)

Code:

- When green flag clicked
- Ask "What material are you sorting?" and wait
- If <(answer) = [plastic] or (answer) = [paper] or (answer) = [metal]>
- Say "Recyclable"
- Else
- Say "Non-Recyclable"
- Repeat the process for a new item

Mark allocation:

- **Event block:** Using the "When green flag clicked" to start the program. **(1 mark)**
- **Input/output block:** Implement the "Ask and wait" block to receive input from the user. (**1 mark)**

- **Conditional logic:** Employ the "If" statement to determine if the item is recyclable based on the materials listed. **(2 marks)**
- **Alternate condition:** Use the "Else" block for non-recyclable conditions. **(1 mark)**
- **Loop mechanism:** Utilising a repeat structure to allow continuous sorting until the program is stopped manually. **(2 marks)**

Question 2: You are creating an interactive story in Scratch where the viewer can choose the path the story takes at critical points, leading to different endings. The story is about a character who goes on a quest to find a lost treasure. The viewer decides if the character takes the 'mountain path' or 'forest path'. (8)

Code:

- When green flag clicked
- Ask "Which path should the character take: mountain or forest?" and wait
- If <(answer) = [mountain]>
- Go to backdrop [Mountain]
- Say "You chose the mountain path! Brave choice!"
- Else
- Go to backdrop [Forest]
- Say "You chose the forest path! Watch out for the mysterious creatures!"

Mark allocation:

- **Initial setup (1 mark):** "When green flag clicked" is used to start the interactive story.
- **Path decision (1 mark):** "Ask 'Which path should the character take: mountain or forest?' and wait" captures the viewer's choice.
- **Conditional logic (1 mark):** "If <(answer) = [mountain]>" checks if the mountain path was selected.
- **Backdrop change (1 mark):** "Go to backdrop [Mountain]" changes the scene to the mountain path.
- **Outcome message (1 mark):** "Say 'You chose the mountain path! Brave choice!'" provides feedback for choosing the mountain path.
- **Alternative path logic (1 mark):** The "Else" condition handles the selection of the forest path.
- **Alternative backdrop change (1 mark):** "Go to backdrop [Forest]" changes the scene to the forest path.

- **Alternative outcome message (1 mark):** "Say 'You chose the forest path! Watch out for the mysterious creatures!'" gives feedback for choosing the forest path.

GRADE 9

SECTION A (20 marks)

1. B) To develop problem-solving skills using computer-based techniques
2. C) A step-by-step procedure for solving a problem
3. B) To repeat a sequence of instructions
4. C) Servo motor
5. C) They detect and respond to physical stimuli
6. C) To find and fix errors
7. B) Prototyping
8. B) To make a decision in the program
9. B) Ethical and responsible behavior online
10. C) Actuators
11. A) To store data values
12. B) It facilitates the sharing of ideas and solutions
13. B) Breaking a program into smaller, manageable parts
14. A) A single process running on multiple CPUs
15. B) Programming that relies on users' actions to determine the flow of execution
16. A) They determine the type of operations that can be performed on data
17. C) Evaluating conditions
18. B) A data type representing true/false conditions
19. A) A method where software is developed and delivered in pieces
20. A) Reducing code complexity and focusing on the operations

SECTION B (5 marks)

1. True
2. False
3. False
4. True
5. True

SECTION C (10 marks)

Question 1: Create a quiz game in Scratch where the player is asked: "What is the capital of France?" Provide immediate feedback based on their answer.

Code:

1. When green flag clicked
2. Ask "What is the capital of France?" and wait
3. If <(answer) = [Paris]>
    - Say "Correct!" for 2 seconds
4. Else
    - Say "Incorrect, the correct answer is Paris." for 2 seconds

Mark allocation:

- **Initialisation** "When green flag clicked" to start the quiz. **(1 mark)**
- **Question prompt** "Ask and wait" to present the quiz question to the user. **(1 mark)**
- **Correct answer feedback** Provide feedback if the answer is correct. **(1 mark)**
- **Incorrect answer feedback** Provide feedback if the answer is incorrect. **(1 mark)**
- **Conditional logic** Employ an "If" statement to differentiate responses based on the player's input. **(1 mark)**

Question 2: Develop a number guessing game in Scratch. The player guesses a number between 1 and 10. If they guess correctly, the game congratulates them; otherwise, it tells them the right answer.

Code:

- When green flag clicked
- Set [Secret Number] to (pick random 1 to 10)
- Ask "Guess a number between 1 and 10" and wait
- If <(answer) = (Secret Number)>
- Say "You guessed right!" for 2 seconds
- Else
- Say "Wrong, the correct number was (Secret Number)" for 2 seconds

Mark Allocation:

- **Initialisation :** Use "When green flag clicked" to start the game and set up the initial conditions. **(1 mark)**

- **Set random number:** Set the [Secret Number] to a random number between 1 and 10. **(1 mark)**
- **Input from user :** Use the "Ask and wait" block to receive the user's guess. **(1 mark)**
- **Correct guess feedback:** If the user's guess matches the secret number, provide positive feedback. **(1 mark)**
- **Incorrect guess feedback :** If the guess is incorrect, inform the user of the correct number. **(1 mark)**

SECTION D (15 marks)

Question 1: Create a Scratch project where the robot monitors two variables: temperature and humidity. If the temperature exceeds 30 degrees Celsius or the humidity exceeds 70%, the robot should alert the user. (7 marks)

Code:

- When green flag clicked
- Set [Temperature] to (pick random 20 to 40)
- Set [Humidity] to (pick random 50 to 90)
- If <(Temperature) > 30 or (Humidity) > 70>
- Say "Alert: Check environmental conditions!"
- Else
- Say "Conditions are normal."

Mark Allocation:

- **Initialisation :** Using "When green flag clicked" to start the simulation. **(1 mark each)**
- **Variable setting:** Setting the [Temperature] and [Humidity] with random values to simulate varying environmental conditions. **(1 mark each = 2 marks in total)**
- **Conditional logic:** Implementing an "If" statement to check if the temperature or humidity exceeds the set thresholds. **(2 marks)**
- **Alert message:** Displaying an alert message if conditions exceed safe limits. **(1 mark)**
- **Normal condition message:** Displaying a normal condition message if thresholds are not exceeded. **(1 mark)**

Question 2: The school library requires a simple program to manage book checkouts and returns. Users should be able to update the status of books as either checked out or

returned. Design a Scratch project that facilitates the tracking of books by allowing users to input a book ID and select the status of the book (checked out or returned). The program should confirm the action for the user in a concise manner.

Code:

- When green flag clicked
- Ask "Enter book ID and status (C/R):" and wait
- If <(answer) contains [C]>
- Say "Book ID (answer before [space]) checked out"
- If <(answer) contains [R]>
- Say "Book ID (answer before [space]) returned"
- Else
- Say "Invalid status entered. Please enter C for checkout or R for return after the Book ID."

Mark allocation:

- **Program initialisation (1 mark):** Start the program with "When green flag clicked."
- **Combined input for efficiency (1 mark):** Use one "Ask and wait" block to collect both the book ID and the checkout or return status in a single input.
- **Check-out logic (2 marks):** Implement an "If" block to detect 'C' in the answer and confirm the book has been checked out.
- **Return logic (2 marks):** Implement a separate "If" block to detect 'R' in the answer and confirm the book has been returned.
- **Error handling (1 mark):** Use an "Else" block for error handling to provide feedback on incorrect entries.
- **Efficient parsing of the input (1 mark):** Extract the book ID from the input by using the text manipulation feature (answer before [space]).