GRADE 4

SECTION A (20 marks)

1. C) A step-by-step procedure for solving a problem or performing a task

2. B) Scanner

3. B) HTML

4. A) Central Processing Unit

5. B) To detect changes in the environment

6. B) Monitor

7. A) 0 and 1

8. B) CPU

9. A) To perform specific tasks

10. B) Writing programmes for devices

11. C) Python

12. B) HyperText Markup Language

13. C) Monitor

14. A) When green flag clicked

15. B) To repeat a sequence of instructions

16. B) A storage location for data

17. B) Makes a decision

18. C) Increased efficiency

19. A) Finding and fixing errors in the code

20. B) Random Access Memory

SECTION B (5 marks)

1. True
2. False
3. True

4. False
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. Detail the steps and Scratch blocks you would use. (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. Write down the steps in order. (5 marks)

Code:

- When green flag clicked
- When space key pressed
- Change y by (10)
- Wait (0.1) seconds
- Change y by (-10)

Mark allocation:

- **Event block** - Begin with an event block to start the game. Use the "When green flag clicked" block to initiate the sprite's movement and obstacle appearance. **(1 mark)**
- **Event block for jump** - Use another event block "When space key pressed" to trigger the jump action. This block will be used to make the sprite jump when the player presses the spacebar. **(1 mark)**
- **Motion block** - Inside the "When space key pressed" block, use a "change y by (10)" for the upward movement and then "change y by (-10)" to come back down. This simulates the jump. **(1 mark each – 2 marks in total)**
- **Control block** - Add a "wait" block inside the jump sequence to control the timing of the jump, ensuring it's visible and realistic. **(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
- If-then
- If <touching colour [maze wall colour]>
- Turn (90 degrees) right
- Move (10) steps
- 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 color [color 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 marks)

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 (0.1) 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 5

SECTION A (20 marks)

1. B) To process instructions
2. C) Python
3. B) Random Access Memory
4. C) Keyboard
5. C) A set of instructions to solve a problem
6. C) Writing programs
7. C) A sequence that repeats several times
8. C) To allow the robot to interact with its environment
9. B) When sprite clicked
10. A) To send messages between sprites
11. B) Input
12. B) It checks for a condition and executes a sequence if the condition is true
13. B) Bytes
14. A) Fixing errors in a programme
15. C) CPU
16. B) It repeats the blocks inside it until a specified condition becomes true
17. B) To measure distance
18. B) Network card
19. B) A change in the sprite's appearance
20. B) Control block

SECTION B (5 marks)

1. True
2. False
3. False

4. False
5. False

SECTION C (10 marks)

Question 1: Write a simple programme using Scratch blocks that will count from 1 to 10 and say each number out loud. (5 marks)

Code:

- When green flag clicked
- Repeat 10
- Change number by 1
- Say (number)
- Wait 2 seconds Mark allocation:
- mark)

Mark allocation:

- **Event block -** Begin with the "When green flag clicked" block to start the counting programme. **(1 mark)**
- **Control block -** Use a "repeat" loop to iterate through numbers from 1 to 10. This tests understanding of loops and their setup. **(1 mark)**
- **Variable block -** "Change number by 1" increases the number each time through the loop, showing an understanding of variables. **(1 mark)**
- **Looks block -** Use a "say" block to vocalise each number, showing application of actions within loops**. (1 mark)**
- **Control block -** "Wait 2 seconds" after saying each number, demonstrating the ability to manage timing within loops. **(1 mark)**

Question 2: Design a sequence of steps to instruct a robot to draw a square. (5 marks)

Code:

- Start
- Repeat 4 times
- Move forward 100 units
- Turn right 90 degrees
- End

Mark allocation:

- **Algorithm Structure** - Begin with "Start" to initialise the drawing sequence. This assesses understanding of starting and ending a program. **(1 mark)**
- **Control Block** - Use a "Repeat" loop to execute the square drawing steps 4 times. This evaluates knowledge of repetition in programming. **(1 mark)**
- **Motion Instructions** - Specify moving forward 100 units and turning right 90 degrees within the loop. Tests precise movement and turning instructions. **(2 marks)**
- **Completion** - Properly closing the algorithm with "End" to signify the end of the program sequence. **(1 mark)**

SECTION D (15 marks)

Question 1: You are to program a light sensor to turn on a light when it gets dark. List the steps and logic you would use in Scratch to program this behaviour. (7)

Code:

- When green flag clicked
- Forever
- If <light sensor detects dark>
- Turn on light
- Else
- Turn off light
- End if

Mark allocation:

- **Event block -** Start with the "When green flag clicked" to initiate sensor monitoring. **(1 mark)**
- **Control block -** Use a "forever" loop to continuously check light levels. **(1 mark)**
- **Sensing block -** Employ an "if" block to check for darkness using the light sensor. **(1 mark)**
- **Action block -** "Turn on light" command within the conditional block for darkness**. (1 mark)**
- **Else block -** Use the "else" block to handle the condition when it is not dark. **(1 mark)**
- **Action block -** "Turn off light" command within the "else" block to deactivate the light when it's light. **(1 mark)**
- **End if -** Proper use of "end if" to close the conditional structure, ensuring logical flow and responsiveness to light conditions. **(1 mark)**

Question 2: Design a simple game in Scratch where a character moves left and right to catch falling apples. If an apple reaches the bottom of the screen without being caught, the game ends. (8)

Code:

- When green flag clicked
- Forever
- If <left arrow key pressed>
- Change x by -10
- If <right arrow key pressed>
- Change x by 10
- If <apple touching edge>
- 'Game Over' for 2 seconds

Mark allocation:

- **Event block** - Start with the "When green flag clicked" to begin the game, setting the stage for gameplay. **(1 mark)**
- **Control block** - Use a "forever" loop to continuously check for key presses and apple position, handling continuous input and game logic. **(1 mark)**
- **Motion blocks** - Specify movement changes for the left and right arrow keys, enabling the character to move horizontally across the screen. **(1x4 mark = 4 marks)**
- **Sensing block** - Implement a condition to detect if an apple touches the edge (bottom) of the screen, triggering the end of the game. **(1 mark)**
- **Game over handling** - Include logic to display 'Game Over' and stop the game when an apple reaches the bottom of the screen without being caught, effectively managing game termination. **(1 mark)**

GRADE 6

SECTION A (20 marks)

1. B) To convert electrical energy into mechanical motion
2. A) Understanding Boolean logic
3. A) To store data that can change during program execution
4. C) Forever
5. B) A numerical label assigned to each device connected to a computer network
6. A) Converts high-level language into machine language
7. D) Using internet-based services for storage and applications
8. B) The basic unit of information in computing and digital communications
9. B) A microcontroller
10. A) Software that is free to use, modify, and distribute
11. A) To provide a platform for building, testing, and debugging software
12. C) 3D graphics programming
13. A) User Interface; it refers to the graphical layout of an application
14. B) Converting data into a secure format for protection
15. B) A set of rules governing the exchange or transmission of data
16. C) Printer
17. A) It allows devices to perform tasks that typically require human intelligence
18. B) Software that protects a network from unauthorised access
19. B) It determines the most relevant results to display in response to a query
20. B) Modem

SECTION B (5 marks)

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

SECTION C (10 marks)

Question 1: Create a Scratch program where a sprite continuously changes its color as it moves across the screen. Detail the steps and Scratch blocks you would use. (5 marks)

Code:

- When green flag clicked
- Forever
- Move (10) steps
- Change colour effect by (25)
- Wait (0.5) seconds

Mark allocation:

- **Event block** - Start with the "When green flag clicked" block to initialise the sprite's actions when the program starts. **(1 mark)**
- **Control block** - Use the "Forever" loop to continuously execute the enclosed instructions. **(1 mark)**
- **Motion block** - Insert a "Move (10) steps" block to make the sprite move forward continuously. **(1 mark)**
- **Looks block** - Use the "Change colour effect by (25)" to dynamically change the sprite's color as it moves. **(1 mark)**
- **Control block for timing** - Add a "Wait (0.5) seconds" block within the forever loop to control the pace of color changes and movement. **(1 mark)**

Question 2: Design a Scratch project where a sprite moves across the screen and changes its size when a key is pressed. (5)

Code:

- When green flag clicked
- Forever
- If key (space key) pressed then change size by 10%
- Move (10) steps
- If on edge, bounce

Mark allocation:

- **Event block** - Start with the "When green flag clicked" block to activate the sprite's movement at the beginning of the program. **(1 mark)**
- **Control block** - Use the "Forever" loop to keep checking and executing the movement and size change commands. **(1 mark)**

- **Conditional block** - Insert "If key (space key) pressed then Change size by 10%" to allow interactive size changes through user input. **(1 mark)**
- **Motion block** - Use "Move (10) steps" to maintain continuous motion across the screen. **(1 mark)**
- **Control block for boundary handling** - Add "If on edge, bounce" to handle the sprite's behavior at the screen edges, making it turn around instead of getting stuck. **(1 mark)**

SECTION D (15 marks)

Question 1: In a Scratch project, you are tasked to simulate a robot navigating through a maze. The robot should start at one end of the maze and move to the exit, avoiding walls. The robot should turn 90 degrees to the right when it encounters a wall. Order the blocks needed to create this simulation. Assign one mark for each correctly ordered code block. (7 marks)

Code:

- When green flag clicked
- Forever
- Move (10) steps
- If <touching [colour black]>
- Move (-10) steps
- Turn right (90 degrees)
- End
- (Use 'turn right (180 degrees)' if needed to handle dead ends or additional logic)

Mark Allocation:

- **Event block:** Start with the "When green flag clicked" block to initiate the program. This ensures that when the green flag is clicked, the robot starts its maze navigation. **(1 mark)**
- **Control block**: Use a "forever" loop to keep the robot moving and checking for walls continuously, simulating constant navigation through the maze. **(1 mark)**
- **Motion block**: Include a "move (10) steps" block inside the forever loop to move the robot forward step by step. This creates the forward motion through the maze corridors. **(1 mark)**
- **Sensing block:** Use an "if <touching [colour black]>" block to detect if the robot has bumped into a wall (black colour represents walls). This allows the robot to react when hitting obstacles. **(1 mark)**

- **Motion block:** Insert a "move (-10) steps" block inside the ‘if’ condition to make the robot step back after hitting a wall, preventing it from getting stuck inside the wall. **(1 mark)**
- **Motion block:** Add a "turn right (90 degrees)" block inside the same ‘if’ condition to make the robot turn to find a new path after encountering a wall. This simulates decision-making at walls. **(1 mark)**
- **Control block:** Use "end" to properly close the ‘if’ condition structure, ensuring logical flow and preventing script errors. (1 mark)
- **Integration and logical flow:** Ensure all these blocks are ordered correctly and logically to allow the robot to navigate the maze without freezing or moving through walls. This mark assesses the student’s ability to integrate multiple Scratch concepts for problem-solving. **(1 mark)**

Question 2:

Design a Scratch project where a character moves horizontally across the screen to catch falling apples. If an apple reaches the bottom edge of the screen without being caught, the game ends. (8 marks)

Code:

• When green flag clicked

• Forever

• If <left arrow key pressed>

• Change x by (-10)

• If <right arrow key pressed>

• Change x by (10)

• If <apple touching edge>

• End game or stop all

• (Move (10) steps block used separately in apple sprite to make apples fall continuously)

Mark Allocation:

• **Event block:** Start with the "when green flag clicked" block to begin the game. This ensures the program starts only when triggered by the player. **(1 mark)**

• **Control block:** Use a "forever" loop to continuously check for keyboard inputs and game conditions, allowing real-time control and updates. **(1 mark)**

• **Sensing & control:** Inside the forever loop, use an "if <left arrow key pressed>" block to detect player input for moving left. **(1 mark)**

• **Motion block:** Include a "change x by (-10)" block inside the left key ‘if’ block to move the character left on the x-axis. **(1 mark)**

• **Sensing & control:** Use another "if <right arrow key pressed>" block to detect player input for moving right. **(1 mark)**

• **Motion block:** Place a "change x by (10)" block inside the right key ‘if’ block to move the character right. **(1 mark)**

• **Sensing block:** Include an "if <apple touching edge>" block (placed in the apple sprite’s script) to detect if the apple has hit the bottom edge of the screen. This determines if the game should end. **(1 mark)**

• **Integration**: The apple sprite separately uses "move (10) steps" inside its own forever loop to continually fall downwards. Proper sequencing of character movement and collision checks with the edge ensures a functional game. **(1 mark)**