higher education & training

Department:
Higher Education and Training
REPUBLIC OF SOUTH AFRICA

MARKING GUIDELINE

NATIONAL CERTIFICATE (VOCATIONAL)

INTRODUCTION TO ROBOTICS
NQF LEVEL 3
(Second paper)

3 December 2024

This marking guideline consists of 10 pages.

Copyright reserved

QUESTION 1

1.1

- Robots can autonomously collect litter, reducing the need for manual rubbish collection.
- Smart waste management robots can sort recyclables from general waste more efficiently.
- Surveillance robots can monitor public spaces for illegal dumping and notify authorities.
- Street-cleaning robots can sweep and sanitise sidewalks and public areas, improving hygiene.
- Robots can clean and disinfect high-touch surfaces, reducing the spread of diseases.
- AI-powered robots can optimise waste collection routes, increasing operational efficiency.
- Robots can monitor air quality and identify sources of pollution in urban areas.
- Drone-based robots can survey and inspect hard-to-reach areas for cleanliness issues.
- Robots can provide real-time data on waste generation patterns to inform better urban planning.

Any 5 bullets will suffice. (Any 5 x 1) (5)

1.2

Criteria	3 Marks	2 Marks	1 Mark	0 Marks
Planning	Planning present and well thought out. Flow diagram used. ✓✓✓	Planning evident but lacks detail or flow.	Minor planning evident.	No Planning Evident
Pin and variable definitions		Pins and variables correctly declared and instantiated. ✓✓	Incorrect pin and variable declarations.	Missing pin and variable declaration
Initialisation and Setup	Correct setup and initialisation of all components. ✓✓✓	Setup and initialisation done with errors.	Incorrect setup and initialisation.	Missing setup and initialisation
Code logic and operations	Code runs as expected with no errors. ✓✓✓	Code runs but has minor logical errors.	Code runs but has major errors.	Code does not run
Comments		Clear and extensive comments used. ✓✓	Incorrect comments.	no comments

Copyright reserved

Use the above rubric to mark the students work. Below is an example code:

```
#include <Servo.h>

const int trigPinApproach = 9; // Trigger pin for the approach sensor
const int echoPinApproach = 10; // Echo pin for the approach sensor
const int trigPinFill = 11; // Trigger pin for the fill level sensor
const int echoPinFill = 12; // Echo pin for the fill level sensor
const int servoPin = 6; // Pin for the servo motor
const int ledPin = 7; // Pin for the LED indicator
const int buzzerPin = 8; // Pin for the buzzer

Servo dustbinLid; // Create a Servo object to control the dustbin lid

const long openDuration = 5000; // Duration (in milliseconds) to keep the
lid open
const int fullThreshold = 20; // Fill level threshold to activate the LED
and buzzer

void setup() {
  // Initialize pins
  pinMode(trigPinApproach, OUTPUT);
  pinMode(echoPinApproach, INPUT);
  pinMode(trigPinFill, OUTPUT);
  pinMode(echoPinFill, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);

  // Attach the servo to its pin
  dustbinLid.attach(servoPin);

  // Set the initial position of the lid (closed)
  dustbinLid.write(0);


  Serial.begin(9600); // Optional - start serial communication for debugging
}

void loop() {
  // Check if someone is approaching
  long distanceApproach = measureDistance(trigPinApproach, echoPinApproach);
  if (distanceApproach < 20) { // Adjust the distance threshold as needed
    openLid();
    delay(openDuration);
    closeLid();
  }

  // Monitor the fill level
  long distanceFill = measureDistance(trigPinFill, echoPinFill);
  if (distanceFill < fullThreshold) {
    digitalWrite(ledPin, HIGH);
    digitalWrite(buzzerPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
    digitalWrite(buzzerPin, LOW);
  }

  // Wait before the next iteration
  delay(500);
}

// Function to measure distance using an ultrasonic sensor
long measureDistance(int trigPin, int echoPin) {
  // Clear the trigger pin
  digitalWrite(trigPin, LOW);
```

Copyright reserved

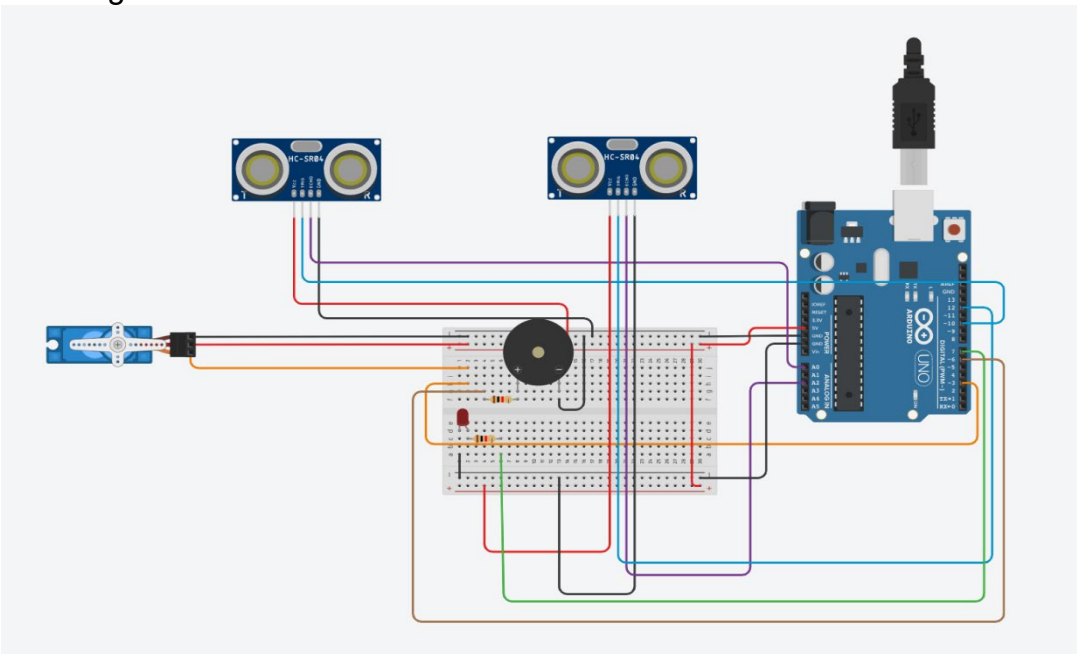
```
62.   delayMicroseconds(2);
63.
64.   // Set the trigger pin high for 10 microseconds
65.   digitalWrite(trigPin, HIGH);
66.   delayMicroseconds(10);
67.   digitalWrite(trigPin, LOW);
68.
69.   // Read the echo pin
70.   long duration = pulseIn(echoPin, HIGH);
71.
72.   // Calculate distance in cm
73.   long distance = duration * 0.034 / 2;
74.
75.   return distance;
76. }
77.
78. // Function to open the dustbin lid
79. void openLid() {
80.   dustbinLid.write(90); // Set servo to 90 degrees (open position)
81. }
82.
83. // Function to close the dustbin lid
84. void closeLid() {
85.   dustbinLid.write(0); // Set servo to 0 degrees (closed position)
86. }
```

(13)

1.3

Criteria	6 Marks	4 Marks	2 Mark	0 Marks
Arduino wiring and connections		Correct pins used for components. Safe wiring and powering of Arduino. ✓✓✓✓	Some wiring issues or incorrect pin connections.	Incorrect or unsafe wiring and pin connections.
Component wiring and connections		All components wired correctly and safely for operation✓✓✓✓	Some components wired incorrectly or with issues.	Incorrect or unsafe wiring of com
Operation of artefact	Prototype works correctly and meets all criteria. ✓✓✓✓✓✓	Prototype works with minor issues.	Prototype has significant issues or only partially meets criteria.	Prototype does not work or fails to meet criteria.

The use of pin numbers may vary, but the project should resemble the following:



(14)

1.4 **Artefact name and description**

- *2 marks one example and an explanation/ description.*
  - Example: "Automated Dustbin"✓ - This device is designed to automatically open its lid when someone approaches and monitor its fill level using ultrasonic sensors. ✓ It alerts the maintenance team with an LED and buzzer when it is full.

Basic operations

- *2 marks for a one example and an explanation/ description* Example:

- **Activating the dustbin✓:** Approach the dustbin to trigger the ultrasonic sensor. ✓
- **Lid operation✓:** The servo motor will open the lid automatically and keep it open for a pre-set duration before closing. ✓
- **Monitoring fill level✓:** The second ultrasonic sensor continuously measures the fill level. If the dustbin reaches its full capacity, the LED and buzzer will activate to alert for emptying. ✓

(any one example)

Troubleshooting

- *2 marks for a one example and an explanation/ description* Example:

- **Issue:** Lid does not open when approached. ✓
- **Solution:** Ensure the servo motor is connected properly and is receiving power. Check for any mechanical obstructions or wiring issues. ✓
- **Issue:** The fill level alert does not activate. ✓
- **Solution:** Inspect the ultrasonic sensor connections and ensure it is correctly aligned and functioning. Test the LED and buzzer separately to verify they are working. ✓

(any one example)

Frequently asked questions (FAQs)

- *2 marks for a one example and an explanation/ description* Example:

- **Q: What should I do if the ultrasonic sensors are giving incorrect readings? ✓**
- **A:** Check for obstructions or dirt on the sensors and ensure they are properly aligned. Adjust the sensor placement if needed. ✓
- **Q: How can I modify the duration for which the lid remains open? ✓**
- **A:** Adjust the delay time in the Arduino code responsible for controlling the servo motor's open/close timing. ✓

(any one example) (8)

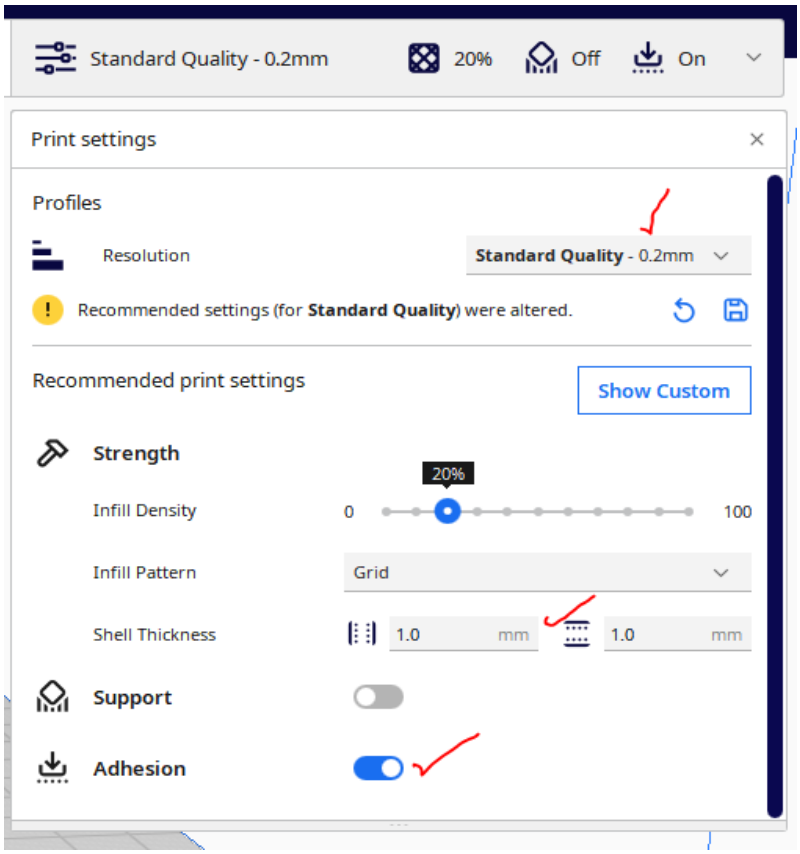
[40]

QUESTION 2

2.1

- The resolution to 0.2mm.✓
- The wall thickness to 1mm (top and bottom)✓
- Enabling a brim adhesion layer.✓
- .gcode file saved.✓

The screenshot should appear as follows:



(4)

2.2 In the code from 2.3, learners should have the following:

```
1.  String html = "<html><body><h1>Temperature Monitoring</h1>";
2.  html += "<p id='temp'>Temperature: " + String(tempCelsius) + " °C</p>";  //
    Use your actual variable for temp
3.  html += "<p>Temperature in F: " + String((tempCelsius * 9/5) + 32) + "
    °F</p>";
4.  html += "<script>";
5.  html += "function reloadPage() {";
6.  html += "    window.location.reload();";
7.  html += "}";
8.  html += "setInterval(reloadPage, 10000);";
9.  html += "</script></body></html>";
10. request->send(200, "text/html", html);
```

Marks awarded for:

- New line html += … ✓
- Placing string in a ✓
- Correct formula applied to temperature variable✓
- Line 8 updated to have 10000 instead of 5000.✓ (4)

2.3

Criteria	5 Marks	3 Marks	2 Marks	0 Mark
Planning		Planning present and well thought out. Flow diagram used. ✓✓✓	Planning evident but lacks detail or flow.	Little to no planning evident.
Initialisation and Setup	Correct setup and initialisation of all components. ✓✓✓✓✓	Minor errors in the initialisation of one of the components.	Setup and initialisation done with errors.	Incorrect or missing setup and initialisation.
Code logic and operations	Code runs as expected with no errors. ✓✓✓✓✓	Code runs but has minor logical errors.	Code runs but with significant logical errors.	Code does not run or has major errors.
Comments			Clear and extensive comments used. ✓✓	Little to no comments or incorrect comments.
Web library setup and utilisation	Correctly setup web server that responds to a local request. ✓✓✓✓✓	Correctly setup web server with minor issues on request handling.	Some errors in web server setup with issues in request handling	Web server not setup or major faults in setup.

(20)

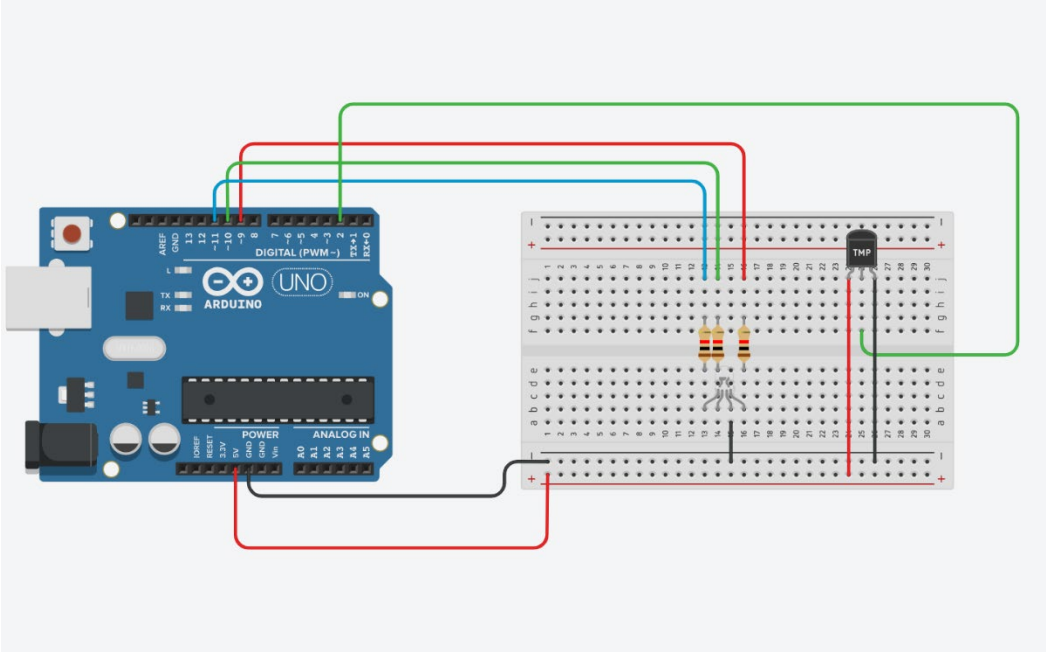
```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <DHT.h>

// Replace with your network credentials
const char* ssid = "Your_SSID";  // Enter your Wi-Fi SSID
const char* password = "Your_PASSWORD";  // Enter your Wi-Fi password

ESP8266WebServer server(80);

// Define the DHT sensor type and pin
#define DHTPIN D4  // DHT sensor connected to GPIO2 (D4)
#define DHTTYPE DHT11  // Use DHT11 sensor
DHT dht(DHTPIN, DHTTYPE);

// Define RGB LED pins
#define REDPIN D1
#define GREENPIN D2
#define BLUEPIN D3

void setup() {
  Serial.begin(115200);

  // Initialize the DHT sensor
  dht.begin();

  // Set up RGB LED pins
  pinMode(REDPIN, OUTPUT);
  pinMode(GREENPIN, OUTPUT);
  pinMode(BLUEPIN, OUTPUT);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
```

```
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to Wi-Fi");

  // Define server routes
  server.on("/", handleRoot);

  // Start the server
  server.begin();
}

void loop() {
  server.handleClient();
}

void handleRoot() {
  float temperature = dht.readTemperature();

  // Set LED colour based on temperature
  if (temperature < 15) {  // Cold
    analogWrite(REDPIN, 0);
    analogWrite(GREENPIN, 0);
    analogWrite(BLUEPIN, 255);
  } else if (temperature >= 15 && temperature <= 25) {  // Moderate
    analogWrite(REDPIN, 0);
    analogWrite(GREENPIN, 255);
    analogWrite(BLUEPIN, 0);
  } else {  // Hot
    analogWrite(REDPIN, 255);
    analogWrite(GREENPIN, 0);
    analogWrite(BLUEPIN, 0);
  }

  // Prepare HTML response
  String html = "<html><body><h1>Temperature Monitoring</h1>";
  html += "<p id='temp'>Temperature: " + String(temperature) + " °C</p>";
  html += "<p>Temperature in F: " + String(temperature * 9 / 5 + 32) + "
°F</p>";
  html += "<script>";
  html += "function reloadPage() {";
  html += "    window.location.reload();";
  html += "}";
  html += "setInterval(reloadPage, 10000);";
  html += "</script></body></html>";

  server.send(200, "text/html", html);
}
```

Copyright reserved

2.4

Criteria	4 Marks	2 Mark	0 Marks
Arduino wiring and connections	Correct pins used for components. Safe wiring and powering of Arduino. ✓✓✓✓	Some wiring issues or incorrect pin connections.	Incorrect or unsafe wiring and pin connections.
Component wiring and connections	All components wired correctly and safely for operation✓✓✓✓	Some components wired incorrectly or with issues.	Incorrect or unsafe wiring of com
Operation of artefact	Prototype works correctly and meets all criteria. ✓✓✓✓	Prototype has significant issues or only partially meets criteria.	Prototype does not work or fails to meet criteria.



(12)
[40]

TOTAL: 80

Copyright reserved