**Task 1:**

Implemented fundamental building blocks: Defined core classes like Node, Input, Parameter for constructing neural networks.

Created essential operations: Implemented Multiply, Addition, Sigmoid, and BCE nodes to perform basic mathematical operations and calculate the binary cross-entropy loss.

Developed the Linear Layer: Created a Linear class that encapsulates the linear transformation (matrix multiplication and bias addition) and includes forward and backward pass methods for efficient gradient computation.

Demonstrated usage: Provided an example of how to use the Linear layer with sample input data and simulated a backward pass.

**Task 2:**

Built a Logistic Regression Model: Constructed a logistic regression model using the previously defined Linear node, Sigmoid activation, and BCE loss.

Trained the Model: Implemented a training loop with forward and backward passes, parameter updates using SGD, and evaluated the model's accuracy on a test set.

Generated and Visualized Data: Generated synthetic data, split it into training and test sets, and visualized the decision boundary of the trained model.

**Task 3 and 4:**

Introduced Batching: Modified the training loop to process data in batches, improving training efficiency and potentially enhancing generalization.

Investigated Batch Size Effects: Experimented with different batch sizes and analyzed their impact on training loss and model accuracy.

Dynamic Learning Rate Adjustment: Implemented a mechanism to adjust the learning rate based on the batch size to maintain stable training across different batch sizes.

Visualized Results: Plotted the training loss over epochs for various batch sizes to visually compare their convergence behavior.

In essence, these codes demonstrate a progression from basic building blocks to a functional logistic regression model, culminating in an investigation of the impact of batching on training dynamics.