This assignment is designed to help you understand how the backpropagation training algorithm for multi-layer feedforward neural networks works. It has three parts.

**Part 1: Gradient descent**

The first part of the assignment is a simple exercise to help you understand the idea of gradient descent. You will use gradient descent to find the minimum of a simple function. Of course, the function must be continuous and differentiable, so that you can compute a derivative.

First, graph the function: \( f(x) = (x - 1)^2 \). What is its derivative?

The following simple C program uses gradient descent to find the minimum of this function, that is, the value of \( x \) for which \( y = f(x) \) has its minimum value.

In the program, the variable alpha is the learning rate. Run this program, and experiment with changing alpha to see what happens. How does changing alpha affect convergence? What would be a reasonable test for convergence? Then modify the program so that it finds the minimum of another function (your choice). Write a one page description of your results, including simple graphs of the functions.

```c
/* Let's find the minimum of \( f(x) = (x - 1)^2 \) using gradient descent */
#include <stdio.h>

float alpha=0.25, x=0.1, y;
int iter=0;

float f(float x) {
    return (x-1)*(x-1);
}
float derivative(float x) {
    return 2 * (x-1);
}

void main(void) {
    do {
        iter++;
        y = f(x);
        printf( "%5d x=%6.3f y=%6.3f\n", iter, x, y);
        x -= alpha * derivative(x);
    } while (iter < 20);
}
```

Part 2: Backpropagation training

The second part of the assignment is to use the backpropagation algorithm to train a feedforward neural network to classify three types of Iris (Iris setosa, Iris versicolor, and Iris Virginica) based on four different attributes (sepal length, sepal width, petal length, and petal width). The training examples for this classification problem can be downloaded from the class website. For this problem, your network will have four input nodes (one for each attribute) and between one and three output nodes, depending on how you encode the output. (For example, you will use three output nodes if there is a distinct node for each type of Iris.) It is up to you to choose the number of hidden nodes that works best. One hidden layer is sufficient.

To make this assignment easier, you do not have to implement the backpropagation algorithm yourself. You are allowed to use code from any source. On the class links page, I will put some links to code that is available on-line. You can use this or any other code. I am not recommending any particular code. It is up to you to find something that works. However, I suggest using simple code that you can easily understand. You are required to indicate the source of your code when you turn in your assignment.

There are 150 examples in the Iris data file available on the class website. Use some of these examples as a training set, and the others as a test set. Recall that an epoch is one pass through the training set. It is normal to take between 1000 and 2000 epochs to train your neural network.

In your report for this part of the assignment, please include a graph of a “learning curve” that plots the error on the y-axis against the training epoch as the x-axis. Also, if you can, draw a picture of the network with its learned weights. Write a one-page description of your results and what you learned. Did you notice any evidence of over-training (i.e., over-fitting)? How can you avoid over-fitting?

Part 3: Applications of neural networks

For the last part of the assignment, write a one-page report that describes an application of neural networks to some practical problem. It is easy to find out about many applications of neural networks, either on the web or from books in the library.