```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.linear_model import Perceptron
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report

def load_data():
    iris = load_iris()
    return iris.data, iris.target, iris.target_names

def train_perceptron(X_train, y_train):
    clf = Perceptron(max_iter=1000, eta0=0.1, random_state=0)
    clf.fit(X_train, y_train)
    return clf

def evaluate_model(clf, X_test, y_test, target_names):
    y_pred = clf.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    print(f"\nPerceptron Accuracy: {acc:.2f}")
    print("\nClassification Report:\n", classification_report(y_test, y_pred,
target_names=target_names))
    return y_test, y_pred

def visualize(y_test, y_pred):
    plt.scatter(range(len(y_test)), y_test, marker='o', label='Actual')
    plt.scatter(range(len(y_pred)), y_pred, marker='x', label='Predicted')
    plt.title('Perceptron - Iris Classification')
    plt.xlabel('Sample Index')
    plt.ylabel('Class')
    plt.legend()
    plt.grid(True)
    plt.show()

def main():
    print("=== Perceptron Based Iris Classification ===")
    X, y, target_names = load_data()

    # Standardize features for faster convergence
    scaler = StandardScaler()
    X = scaler.fit_transform(X)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

    clf = train_perceptron(X_train, y_train)
    y_test, y_pred = evaluate_model(clf, X_test, y_test, target_names)
    visualize(y_test, y_pred)

if __name__ == "__main__":
    main()
```