```python
import numpy as np
from lib import *


def part_a():
    # get the data
    X, y = generate_data()

    # perform linear regression
    weights = linear_regression(X, y)
    # measure error
    y_pred = X @ weights
    error = mean_squared_error(y, y_pred)
    print("Least squares error on original data:", error)
    # plot, remember to use original X to plot even if you augment
    plot_compare(X, y, y_pred, figname="ols_original.png")


    """YOUR CODE HERE"""
    # PART A: Implement the augmenting function in lib and
    # follow a similar procedure as above to plot the regression
    # for different degrees of polynomials as specified in the problem.
    augmented_X_2 = polynomial_augmentation(X, 2)
    augmented_X_4 = polynomial_augmentation(X, 4)
    augmented_X_6 = polynomial_augmentation(X, 6)
    weights_2 = linear_regression(augmented_X_2, y)
    weights_4 = linear_regression(augmented_X_4, y)
    weights_6 = linear_regression(augmented_X_6, y)
    y_pred_2 = augmented_X_2 @ weights_2
    y_pred_4 = augmented_X_4 @ weights_4
    y_pred_6 = augmented_X_6 @ weights_6
    error_2 = mean_squared_error(y, y_pred_2)
    error_4 = mean_squared_error(y, y_pred_4)
    error_6 = mean_squared_error(y, y_pred_6)
    print("Least squares error on augmented data with degree 2:", error_2)
    print("Least squares error on augmented data with degree 4:", error_4)
    print("Least squares error on augmented data wiht degree 6:", error_6)
    plot_compare(X, y, y_pred_2, figname="ols_augmented_2.png")
    plot_compare(X, y, y_pred_4, figname="ols_augmented_4.png")
    plot_compare(X, y, y_pred_6, figname="ols_augmented_6.png")
    """YOUR CODE ENDS"""


def part_b():
    X, y = generate_data()
    """YOUR CODE HERE"""
    # PART B: Implement the rbf augmenting function in lib and then follow
    # the same procedure as before to produce a graph of the fitted data.

    # decide on three means and variances that best represent the data
    # (look at the plot from part a to see gaussian shapes!)
    means = [-2, 1, 3]
    variances = [3, 2, 0.5]
    X_rbf = rbf_augmentation(X, means, variances)
    weights = linear_regression(X_rbf, y)
```

```
    y_pred = X_rbf @ weights
    error = mean_squared_error(y, y_pred)
    print("Least squares error on augmented data with Gaussian function:",
     error)
    plot_compare(X, y, y_pred, figname="ols_augmented_guassian.png")
    """YOUR CODE ENDS"""


def main():
    part = input('Which part to run?\n')
    if part == 'a':
        part_a()
    elif part == 'b':
        part_b()
    else:
        print("Invalid part.")
        exit(1)

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