Question: Write a single python file to perform the following tasks:

(a) Get dataset "`from sklearn.datasets import load_iris`". This dataset has 4 features. Split the dataset into two sets: 20% of samples for training, and 80% of samples for testing.
NOTE 1: Please use "`from sklearn.model_selection import train_test_split`" with "`random_state=N`" and "`test_size=0.8`".
NOTE 2: The offset/bias column is not needed here for augmenting the input features.

(b) Generate the target output using one-hot encoding for both the training set and the test set.

(c) Using the same training and test sets generated above, perform a polynomial regression (utilizing "`from sklearn.preprocessing import PolynomialFeatures`") from orders 1 to 10 (adopting the weight-decay L2 regularization with regularization factor λ=0.0001) for classification (based on the one-hot encoding) and compute the number of training and test samples that are classified correctly.
NOTE 1: The offset/bias augmentation will be automatically generated by `PolynomialFeatures`.
NOTE 2: If the number of rows in the training polynomial matrix is less than or equal to the number of columns, then use the dual form of ridge regression (Lecture 6). If not, use the primal form (Lecture 6).

Instructions: please submit a single python file with filename It should contain a function `A2` that takes in an integer "`N`" random_state as input and returns the following outputs in the following order:

- `X_train`: training numpy feature matrix with dimensions (number_of_training_samples x 4). (1%)
- `y_train`: training target numpy array (containing values 0, 1 and 2) of length number_of_training_samples. (1%)
- `X_test`: test numpy feature matrix with dimensions (number_of_test_samples x 4). (1%)
- `y_test`: test target numpy array (containing values 0, 1 and 2) of length number_of_test_samples. (1%)
- `Ytr`: one-hot encoded training target numpy matrix (containing only values 0 and 1) with dimension (number_of_training_samples x 3). (1%)
- `Yts`: one-hot encoded test target numpy matrix (containing only values 0 and 1) with dimension (number_of_test_samples x 3). (1%)
- `Ptrain_list`: list of training polynomial matrices for orders 1 to 10. Ptrain_list[0] should be polynomial matrices for order 1 (size number_of_training_samples x 5), Ptrain_list[1] should be polynomial matrices for order 2 (size number_of_training_samples x 15), etc. (1.5%)
- `Ptest_list`: list of test polynomial matrices for orders 1 to 10. Ptest_list[0] should be polynomial matrices for order 1, Ptest_list[1] should be polynomial matrices for order 2, etc. (1.5%)
- `w_list`: list of estimated regression coefficients for orders 1 to 10. w_list[0] should be estimated regression coefficients for order 1, w_list[1] should be estimated regression coefficients for order 2, etc. (2%)
- `error_train_array`: numpy array of training error counts (error count = number of samples classified incorrectly) for orders 1 to 10. error_train_array[0] is error count for polynomial order 1, error_train_array[1] is error count for polynomial order 2, etc. (2%)
- `error_test_array`: numpy array of test error counts (error count = number of samples classified incorrectly) for orders 1 to 10. error_test_array[0] is error count for polynomial order 1, error_test_array[1] is error count for polynomial order 2, etc. (2%)

Please use the python template provided to you. Do not comment out any lines. Remember to rename both and using your student matriculation number. For example, if your matriculation ID is A1234567R, then you should submit "A2_A1234567R.py" that contains the function "A2_A1234567R". Please do NOT zip/compress your file. Please test your code at least once.

Because of the large class size, **points will be deducted if instructions are not followed**. The way we would run your code might be something like this:

```
>> import A2_A1234567R as grading
>> N = 5
>> X_train, y_train, X_test, y_test, Ytr, Yts, Ptrain_list, Ptest_list,
w_list, error_train_array, error_test_array = grading.A2_A1234567R(N)
```