Question 1

What is the optimal value of alpha for ridge and lasso regression? What will be the changes in the model if you choose double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

Optimal value of lambda for Ridge Regression = **10**
Optimal value of lambda for Lasso Regression = **0.001**
After doubling the lambda value for both ridge and lasso regression are:
Ridge:
r2_score for train using ridge regression 0.94 to 0.93
r2_score for test using ridge regression 0.92 remains same
Lasso:
r2_score for train using lasso regression 0.92 to 0.90
r2_score for test using lasso regression 0.92 to 0.91
The most predictor variables after doubling the alpha values are:

- GrLivArea
- OverallQual_8
- OverallQual_9
- Neighborhood_Crawfor
- Functional_Typ
- Exterior1st_BrkFace
- OverallCond_9
- TotalBsmtSF
- CentralAir_Y
- OverallCond_7

```
## To interpret the ridge coefficients
ridge_coeffs = np.exp(betas['Ridge'])
ridge_coeffs.sort_values(ascending=Fal
```

```
]:  GrLivArea               1.085212
    OverallQual_8           1.073766
    OverallQual_9           1.066059
    Neighborhood_Crawfor    1.065026
    Functional_Typ          1.063942
    Exterior1st_BrkFace     1.059093
    OverallCond_9           1.056856
    TotalBsmtSF             1.050231
```

Question 2
You have determined the optimal value of lambda for ridge and lasso regression during the assignment. Now, which one will you choose to apply and why?

The selection between the lasso and ridge regression depends on specific usecase dataset.
If you need feature selection, go for Lasso regression because it can set some coefficients to zero, effectively selecting a subset of features.
If your features are highly correlated, Ridge might be more suitable since it tends to distribute the coefficients among correlated features rather than selecting one and ignoring others.
I prefer to use Lasso regression because I got the optimal value 0.001 with nearly equal r2_score value for both train and test datasets.

Question 3
After building the model, you realised that the five most important predictor variables in the lasso model are not available in the incoming data. You will now have to create another model excluding the five most important predictor variables. Which are the five most important predictor variables now?

After dropping the five most predictor variables (OverallQual_9,GrLivArea,OverallQual_8,Neighborhood_Crawfor,Functional_Typ) using lasso,The r2_score values are:
r2_score for train using lasso regression 0.9115497897032407
r2_score for test using lasso regression 0.9221954603884809
we get the following new top 5 predictors:-

2ndFlrSF
Exterior1st_BrkFace
1stFlrSF
Neighborhood_Somerst
MSSubClass_70

```
## To interpret the lasso coeffic
lasso_coeffs = np.exp(betas['Lass
lasso_coeffs.sort_values(ascendin
```

```
]: 2ndFlrSF                 1.105869
   Exterior1st_BrkFace      1.091758
   1stFlrSF                 1.068318
   Neighborhood_Somerst     1.063768
   MSSubClass_70            1.059417
   TotalBsmtSF              1.054665
   Neighborhood_NridgHt     1.049874
   CentralAir_Y             1.046425
```

Question 4

How can you make sure that a model is robust and generalisable? What are the implications of the same for the accuracy of the model and why?

Making sure a model is robust and generalizable is essential for its success on new data. A robust and generalizable model does well not only on the data it was trained with but also on new, unseen data.
To make sure the model is robust and generalisable ,

1. Model should take care of overfitting and underfitting.
2. Model should not be too complex.
3. Cross-Validation: Split the data into multiple parts and train/test the model multiple times with different splits. Because to ensure the model's performance is consistent across different subsets of data.
4. Hyperparameter Tuning: Find the best parameters for the model using methods to optimize the model's performance.
5. Using regularization techniques like lasso,ridge regression by adding penalty or cost function to the model.

From the perspective of accuracy, a model that is too complex will have very high accuracy on the training data. To make our model more robust and generalizable, we need to reduce its complexity, which will increase bias and decrease variance. This increase in bias means that the model's accuracy will decrease somewhat.