Consider the `insurance.csv` data file. This file contains basic demographic information on customers. The goal is to build a regression model to predict insurance premium. **In Python**, answer the following:

1. (3 points) Using the pandas library, read the csv data files and create three a data-frame called `insurance`.
2. (5 points) Change `sex`, `smoker` and `region` from labels to dummy variables.
3. (5 points) Engineer the interactions/features from Chapter 4 lecture notes (the ones from the decision tree).
4. (5 points) Based on the feature selection analysis shown in Chapter 4, it seems that `age`, `bmi`, `children`, `smoker`, and `interaction_4` are the top 5 important variables. Using the top variables as input variables and `charges` as the target variable, split the data into three datasets: `train` (80%) and `test` (20%).
5. (10 points) Using `train` data-frame and the top 5 features, perform a hyper-tuning job on the random forest model. Using the Optuna framework and the following dictionary:

   ```
   params = dict(n_estimators = trial.suggest_int('n_estimators', 100, 2000),
                 min_samples_split = trial.suggest_int('min_samples_split', 5, 30),
                 min_samples_leaf = trial.suggest_int('min_samples_leaf', 5, 30),
                 max_depth = trial.suggest_int('max_depth', 2, 10)
                 )
   ```

   perform the hyper-parameter job with 3 folds. Identify the hyper-parameter combination that produces the minimum mean squared error. Then, use that model to predict the `charges` on the `test` data-frames. Finally, compute the mean squared error of the predictions on the `test` data-frame.
6. (10 points) Using `train` data-frame and the top 5 features, perform a hyper-tuning job on the XGBoost model. Using the Optuna framework and the following dictionary:

   ```
   params = dict(n_estimators = trial.suggest_int('n_estimators', 100, 2000),
                 max_depth = trial.suggest_int('max_depth', 2, 10),
                 min_child_weight = trial.suggest_int('min_child_weight', 2, 20),
                 learning_rate = trial.suggest_float('learning_rate', 0.01, 100, log = True),
                 gamma = trial.suggest_float('gamma', 0, 10),
                 colsample_bytree = trial.suggest_float('colsample_bytree', 0.2, 0.9),
                 subsample = trial.suggest_float('subsample', 0.2, 0.9)
                )
   ```

   perform the hyper-parameter job with 3 folds. Identify the hyper-parameter combination that produces the minimum mean squared error. Then, use that model to predict the `charges` on the `test` data-frames. Finally, compute the mean squared error of the predictions on the `test` data-frame.
7. (3 points) Based on your results from parts 5, and 6, what model would you use to predict `charges`? Be specific.