```
# Assessed exercises 6

# Import packages
from pandas import Series, DataFrame
import pandas as pd
import numpy as np
import numpy.random as npr

# This week we will use the san-francisco-2018 dataset to test the code. It
# contains salary information for over 40,000 workers in San Francisco. I have
# taken a subset of 500 of the entries from this dataset, and removed some of t
# the entries to create NaN entries. Download the dataset, load it in and have
# a look at the first few entries to see what it looks like.
data = pd.read_csv('san-francisco-2018.csv',index_col='Name')
# Q1 Write a function that will calculate the column means for a given DataFrame
# df and returns the DataFrame with the column means removed (subtracted)
def exercise1(df):
   return df - df.mean()  # it returns (original values - column mean)

# Suggested test
# Take a small subset of the data, and drop the categorical data
dataQ1 = data.drop(['Job Title','Status'],axis=1).iloc[80:100]
exercise1(dataQ1)
# This should return a DataFrame with 20 rows and 4 columns, with the difference
# from the mean for each measurement, for each person

# Q2 Write a function that computes summary statistics for a DataFrame df. The
# function should return a DataFrame with the summary statistics, mean, standard
# deviation, minimum, maximum, the index for the first minimum and the index
# for the first maximum. The outputted DataFrame should have 6 rows, one for
# each piece of information listed above, labelled 'mean', 'std', 'min', 'max',
# 'minloc', 'maxloc' (in this exact order). The columns should be the same as
# those in the original DataFrame (in the same order). You can assume that df
# does not contain categorical data
def exercise2(df):
    # get 'mean', 'std', 'min', 'max','minloc', 'maxloc' from each column in df. Usin
    # combine them into a matrix
    data = np.vstack((df.mean().T, df.std().T, df.min().T, df.max().T, df.idxmin().T,
    # create a dataframe with data, required indexes and column names.
    df1 = pd.DataFrame(data,index = ['mean','std','min','max','minloc','maxloc'],colu
    return df1

# Suggested test
# Remove the categorical data. Once you've completed Q4 you should have a
# generalisable way to remove categorical data
dataQ2 = data.drop(['Job Title','Status'],axis=1)
exercise2(dataQ2)
# This should give you back a DataFrame with 6 row and 4 columns, containing
# the mean, std, min, max, minloc, maxloc (in that order) for Base Pay, Overtime
# Pay, Other Pay and Benefits. minloc and maxloc should contain the name of the
# person with the min/max Base Pay, Overtime Pay, etc.


# Q3 Write a function that will return the index of the 3 highest entries for
# each of the columns in a DataFrame df. The function should return a DataFrame,
# where the rows are the columns of the DataFrame df, and the columns labelled
# '1st', '2nd' and '3rd' contain the index label of the 3 highest entries in
# the given column. Again, you can assume that df does not contain categorical
# data
def exercise3(df):
    # fill all NaN data at first, then the data array can be sorted.
    df = df.fillna(0)
    # get column labales
    col = df.columns
    # create a list to store the 1st,2nd,3rd heightest values.
    temp = list()
    for i in range(len(col)):
        # sort each column in descending and get the first three one.
        L = sorted(df[col[i]],reverse = True)
        temp.append(L[0:3])
    # convert temp to array and use it to create a dataframe
    temp = np.array(temp)
    df3 = pd.DataFrame(temp,index = col, columns = ['first','second','third'])
    return df3
# Suggested test
# We can use the same data from Q2
exercise3(dataQ2)
# This should return a DataFrame with 4 rows and 3 columns, where their rows are
# Base Pay, Overtime Pay, Other Pay and Benefits, and the columns 1st, 2nd and
# 3rd. The entries should be the names of the employees with the highest, second
# highest and third highest Base Pay, Overtime Pay, etc. Look at the DataFrame
# dataQ2 to ensure the function is returning the correct information.

# Q4 In this question you need to write a function to replace all of the NaNs
# in a DataFrame df with the mean of the column for numeric data and the mode
# of the column for categorical data. If a column of categorical data has more
# than one mode you should use the first one. The function should return df with
# the missing values replaced as outlined above. This function must work for any
# DataFrame, so you cannot use the column names inside the function. You can
# assume that a column won't have both numerical and categorical data in it.
# Hint: You'll need to figure out how to select columns based on their data type
# and then use drop to make the replacements for numeric and categorical data
# separately.
def exercise4(df):
    col = df.columns
    for i in range(len(col)):
        # get the first item in each column and identify whether it is string type
        # string type means this is a categorical columns
        if (type(df[col[i]][0]) == type('str')):
            temp = df[col[i]].mode()
            df[col[i]] = df[col[i]].fillna(temp[0])
        # others are numeric columns
        else:
            df[col[i]] = df[col[i]].fillna(df[col[i]].mean())
    return df

# Suggested test
# Take a subset of the data, look at the data and see that there are NaN entries
dataQ4 = data.iloc[150:180]
exercise4(dataQ4)
# This should return a DataFrame with the same dimensions and values as dataQ4,
# with all of the NaN entries filled.
# This function could now be used to replace all of the NaNs for the entire
# DataFrame, or any DataFrame for that matter
```