```python
################################################################################
# IMPORTS
################################################################################
import numpy as np
import os
import matplotlib.pyplot as plt
import matplotlib as mpl
##############################################################################
# Variable declaration
##############################################################################
isPlotReqd = True
isPlotPdf = True
##############################################################################
# Settings for plot
##############################################################################
if (True == isPlotReqd):
    if (True == isPlotPdf):
        mpl.use('pdf')
        fig_width  = 3.487
        fig_height = fig_width / 1.618
        rcParams = {
            'font.family': 'serif',
            'font.serif': 'Times New Roman',
            'text.usetex': True,
            'xtick.labelsize': 8,
            'ytick.labelsize': 8,
            'axes.labelsize': 8,
            'legend.fontsize': 8,
            'figure.figsize': [fig_width, fig_height]
           }
        plt.rcParams.update(rcParams)
##############################################################################
# CODE STARTS HERE
##############################################################################
n_samples = 10000
n_bins = 100
phi = np.random.uniform(0, np.pi, n_samples)
y_hat = np.sin(phi)
y_hat_mean = np.mean(y_hat)
y_hat_var = np.var(y_hat)
print('UNIFORM Phi: Mean of y_hat = ', y_hat_mean)
print('UNIFORM Phi: Variance of y_hat = ', y_hat_var)
#######################################################################
# Plot of histogram of y_hat: UNIFORM Phi
#######################################################################
if (True == isPlotReqd):
    #######################################################################
    # Configure axis and grid
    #######################################################################
    fig = plt.figure()
    ax = fig.add_subplot(111)
    fig.subplots_adjust(left=.15, bottom=.16, right=.99, top=.97)

    ax.set_axisbelow(True)
    ax.minorticks_on()
    ax.grid(which='major', linestyle='-', linewidth='0.5')
    ax.grid(which='minor', linestyle="-.", linewidth='0.5')
    y_linspace = np.linspace(0, 1, n_samples, endpoint=False)
    y_pdf = 2./(np.pi*np.sqrt(1-y_linspace**2))
    ax.hist(y_hat, bins=n_bins, density=True, label='Monte Carlo')
    ax.plot(y_linspace, y_pdf, 'r', label='Analytical PDF')

    ax.set_xlabel(r'$\hat{y}$', fontsize=8)
    ax.set_ylabel(r'$f_{\hat{Y}(\hat{y})}$', fontsize=8)

    plt.legend()
    if (True == isPlotPdf):
        if not os.path.exists('./generatedPlots'):
            os.makedirs('generatedPlots')
        fig.savefig('./generatedPlots/q1_unif_phi_mc.pdf')
    else:
        plt.show()
################################################################################
# 1.d Gaussian phi
################################################################################
phi_norm = np.random.normal(0, 1, n_samples)
y_hat_norm = np.sin(phi_norm)
y_hat_mean_norm = np.mean(y_hat_norm)
y_hat_var_norm = np.var(y_hat_norm)
print('NORMAL Phi: Mean of y_hat = ', y_hat_mean_norm)
print('NORMAL Phi: Variance of y_hat = ', y_hat_var_norm)
#######################################################################
# Plot of histogram of y_hat: NORMAL Phi
#######################################################################
if (True == isPlotReqd):
    ###########################################################################
    # Configure axis and grid
    ###########################################################################
    fig = plt.figure()
    ax = fig.add_subplot(111)
    fig.subplots_adjust(left=.15, bottom=.16, right=.99, top=.97)

    ax.set_axisbelow(True)
    ax.minorticks_on()
    ax.grid(which='major', linestyle='-', linewidth='0.5')
    ax.grid(which='minor', linestyle="-.", linewidth='0.5')

    ax.hist(y_hat_norm, bins=n_bins, density=True, label='Monte Carlo: Normal Phi')

    ax.set_xlabel(r'$\hat{y}$', fontsize=8)
    ax.set_ylabel(r'$f_{\hat{Y}(\hat{y})}$', fontsize=8)

    plt.legend()
    if (True == isPlotPdf):
        if not os.path.exists('./generatedPlots'):
            os.makedirs('generatedPlots')
        fig.savefig('./generatedPlots/q1_norm_phi_mc.pdf')
    else:
        plt.show()
```