6. Write a program to implement k-nearest neighbour algorithm to classify the iris dataset. Print both correct and wrong predictions. Java/python ML library classes can be used for this problem.

```
from sklearn.dataset import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrices import classification_report
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrices import confusion_matrix
from sklearn.metrices import accuracy_score
iris_dataset = load_iris()

print("\nIRIS FEATURES \TARGET NAMES:\n", iris_dataset, target_names)

for i in range(len(iris_dataset.target_names)):
    print("\n [{0}] : [{1}]".format(i, iris_dataset.target_names[i]))


x_train, x_test, y_train, y_test = train_test_split(iris_dataset["data"],
iris_dataset["target"], random_state=0)

classifier = KNeighborsClassifier(n_neighbors = 8, p = 3, metric = "euclidean")

classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
```

```
cm = confusion_matrix(y_test, y-pred)
print(" Confusion matrix is as follows \n", cm)
print("Accuracy metrices")
print(Classification_report(y_test, y-pred))
print(" Correct prediction ", accuracy-score(y-test, y-pred))
print(" wrong prediction ", (1-accuracy-score(y-test, y-pred)))
```

Output

```
IRIS FEATURES\TARGET NAMES:

['setosa', 'versicolor' 'virginica']

[0] : [setosa]
[1] : [versicolor]
[2] : [virginica]

KNeighborsClassifier(algorithm = "auto", leaf_size = 30, metric =
        'euclidean', metric_params = None, n_jobs = None,
        n_neighboos = 8, p = 3, weights = 'uniform')

Confusion matrix is as follows:
[[13  0  0]
 [0  15  1]
 [0   0  9]]

Accuracy metrics

            prediction    recall    f1-score    support
    0          1.00        1.00       1.00         13
    1          1.00        0.94       0.97          6
    2          0.90        1.00       0.95          9

accuracy                              0.97         38
macro avg       0.97       0.98       0.97         38
weighted avg    0.98       0.97       0.97         38
```

Jouita, 4MT17CS044

correct prediction : 0.9736842105263158

wrong prediction: 0.02631578947368418