Task 1

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, recall_score
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer

# Load Dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
# NOTE: We are simulating a 'Stroke' dataset using the Pima dataset structure for stability
names = ['feature1', 'feature2', 'feature3', 'feature4', 'feature5', 'mass', 'pedi', 'age', 'class']
df = pd.read_csv(url, names=names)

print("Environment Ready.")
```

```
Environment Ready.
```

```
df.head()
```

	feature1	feature2	feature3	feature4	feature5	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Next steps: Generate code with df | New interactive sheet

```
X = df.drop('class', axis=1)
y = df['class']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```

```
len(X_test)
```

```
192
```

Task 2

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, recall_score

knn_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('knn', KNeighborsClassifier(n_neighbors=5))
])


knn_pipeline.fit(X_train, y_train)


y_pred = knn_pipeline.predict(X_test)


accuracy = accuracy_score(y_test, y_pred)
print("Accuracy Score:", round(accuracy, 2))
```

```
Accuracy Score: 0.68
```

Task 3

```
recall = recall_score(y_test, y_pred, pos_label=1)
print("Recall Score:", round(recall, 2))
```

```
Recall Score: 0.54
```

Task 4

```
dt_model = DecisionTreeClassifier(max_depth=3, random_state=42)
dt_model.fit(X_train, y_train)


y_pred = dt_model.predict(X_test)


accuracy = accuracy_score(y_test, y_pred)
print("Accuracy Score of Decision Tree:", round(accuracy, 2))
```

```
Accuracy Score of Decision Tree: 0.71
```

Task 5

```
importances = dt_model.feature_importances_

max_importance = max(importances)
print("Highest feature importance value:", round(max_importance, 2))
```

```
Highest feature importance value: 0.71
```