**Lab 4: Python Library Pandas**

**Q1. Loading and Inspecting Data:**

* Load various data formats (CSV, Excel, JSON) into Pandas DataFrames.
* Explore DataFrame attributes like shape, columns, dtypes, head, tail, info, describe.
* Practice selecting columns and rows using different methods (indexing, slicing, loc, iloc).

import pandas as pd

df = pd.read\_csv("Iris.csv") print("Shape:", df.shape)

print("Columns:", df.columns) print("Data types:", df.dtypes) print("First 5 rows:\n", df.head()) print("Last 5 rows:\n", df.tail())

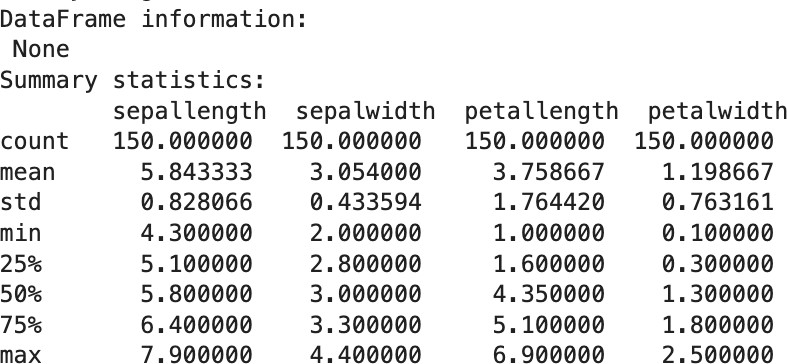
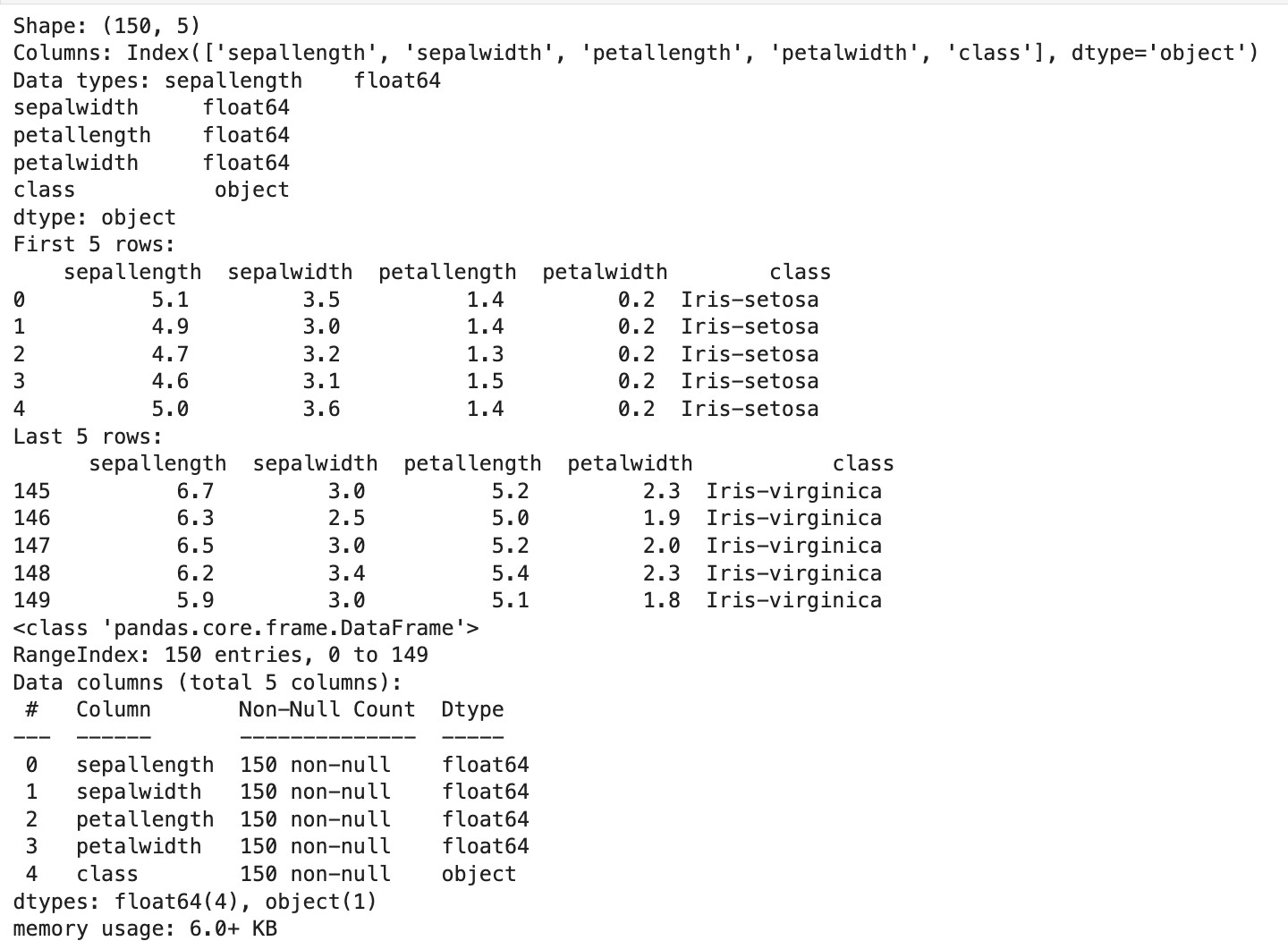
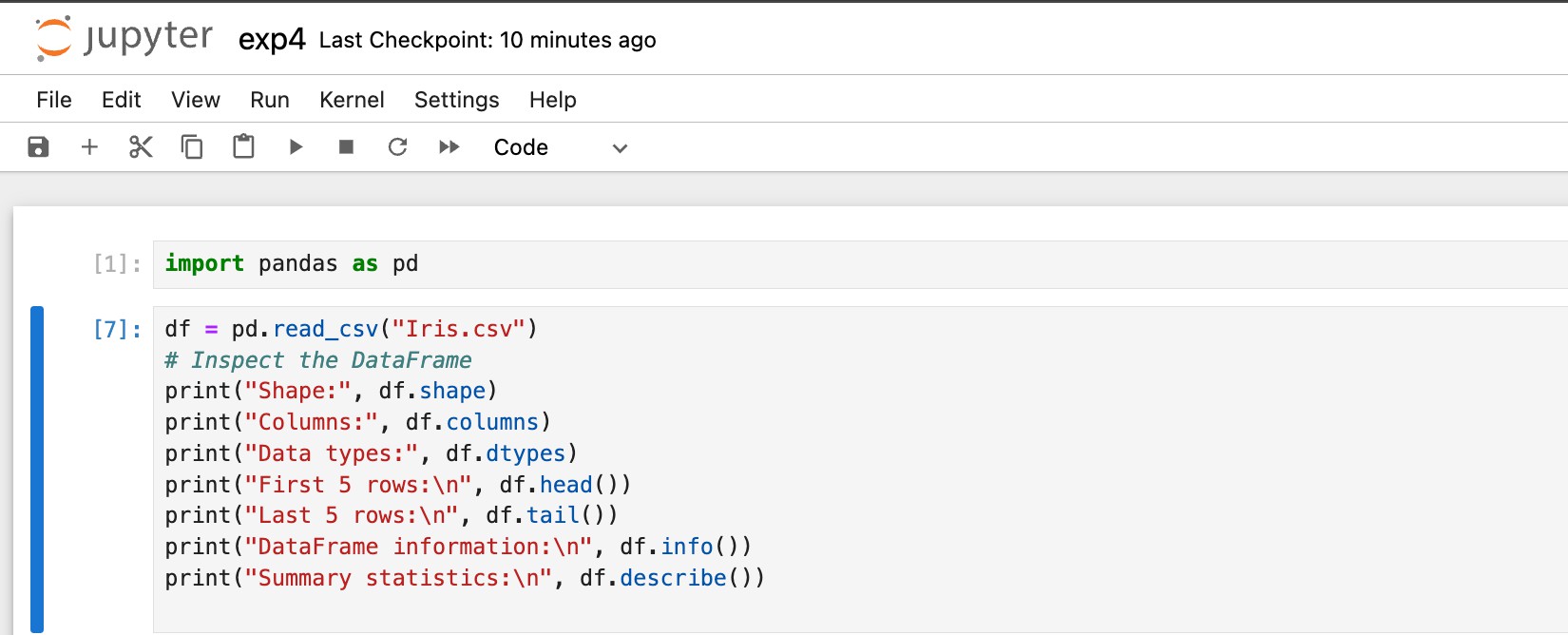
print("DataFrame information:\n", df.info()) print("Summary statistics:\n", df.describe())

sepal\_length\_data = df['sepallength'] first\_row = df.iloc[0]

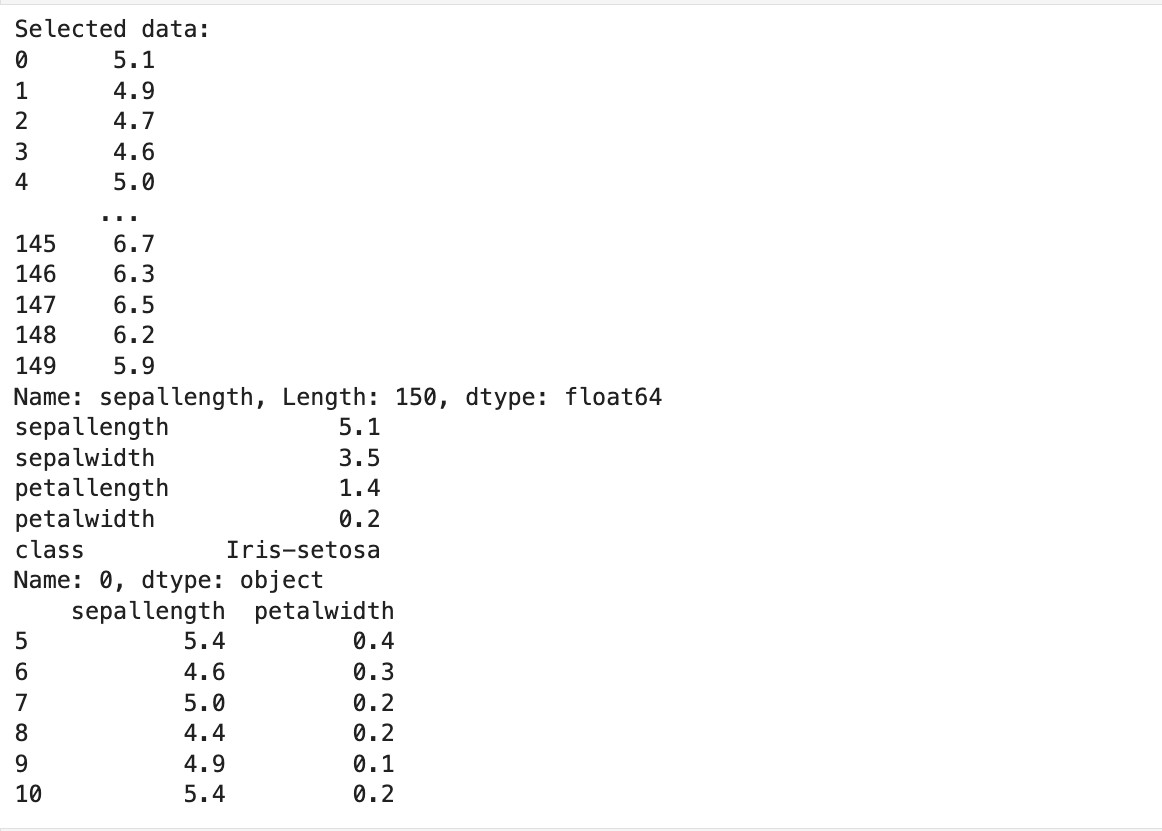
specific\_rows\_and\_columns = df.loc[5:10, ['sepallength', ‘petalwidth']]

**# Print the selected data** print("Selected data:") print(sepal\_length\_data) print(first\_row) print(specific\_rows\_and\_columns)

* **Output:**







**Q2. Data Cleaning and Preparation**

1. Identify missing values using isnull and isna.
2. Handle missing values using fillna, dropna, interpolation.
3. Apply scaling techniques (min-max, z-score) to numerical columns.
4. Create dummy variables for categorical columns.

import pandas as pd

from sklearn.preprocessing import MinMaxScaler, StandardScaler df = pd.read\_csv('Iris.csv')

# # Display the first few rows of the dataset

print("Initial Data:\n", df.head())

**# Identify missing values using isnull and isna** print("\nMissing values with isnull:\n", df.isnull().sum()) print("\nMissing values with isna:\n", df.isna().sum())

# # Handle missing values

numerical\_cols = df.select\_dtypes(include=['float64', 'int64']).columns df[numerical\_cols] = df[numerical\_cols].fillna(df[numerical\_cols].mean()) **# Apply scaling techniques**

min\_max\_scaler = MinMaxScaler()

df[numerical\_cols] = min\_max\_scaler.fit\_transform(df[numerical\_cols])

# # Create dummy variables for categorical columns

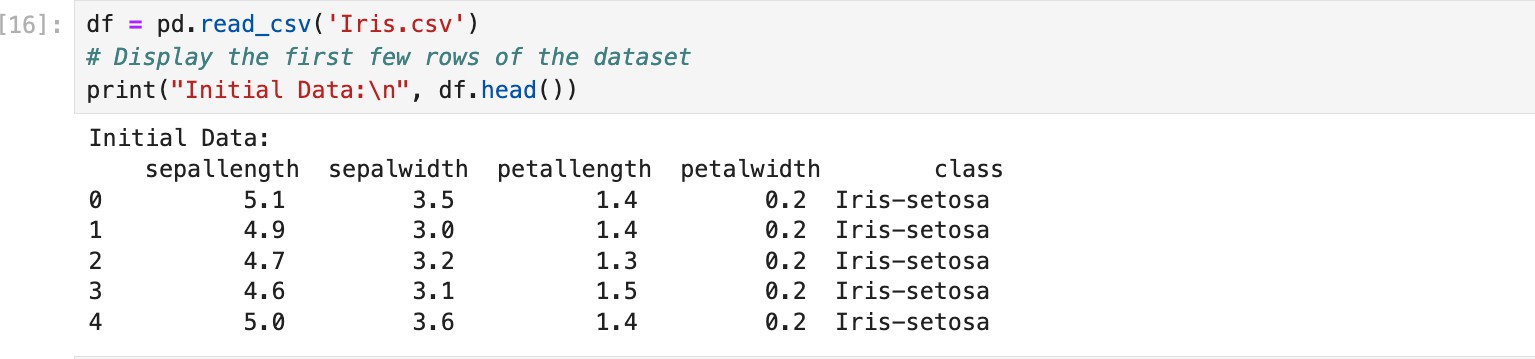
categorical\_cols = df.select\_dtypes(include=['object']).columns

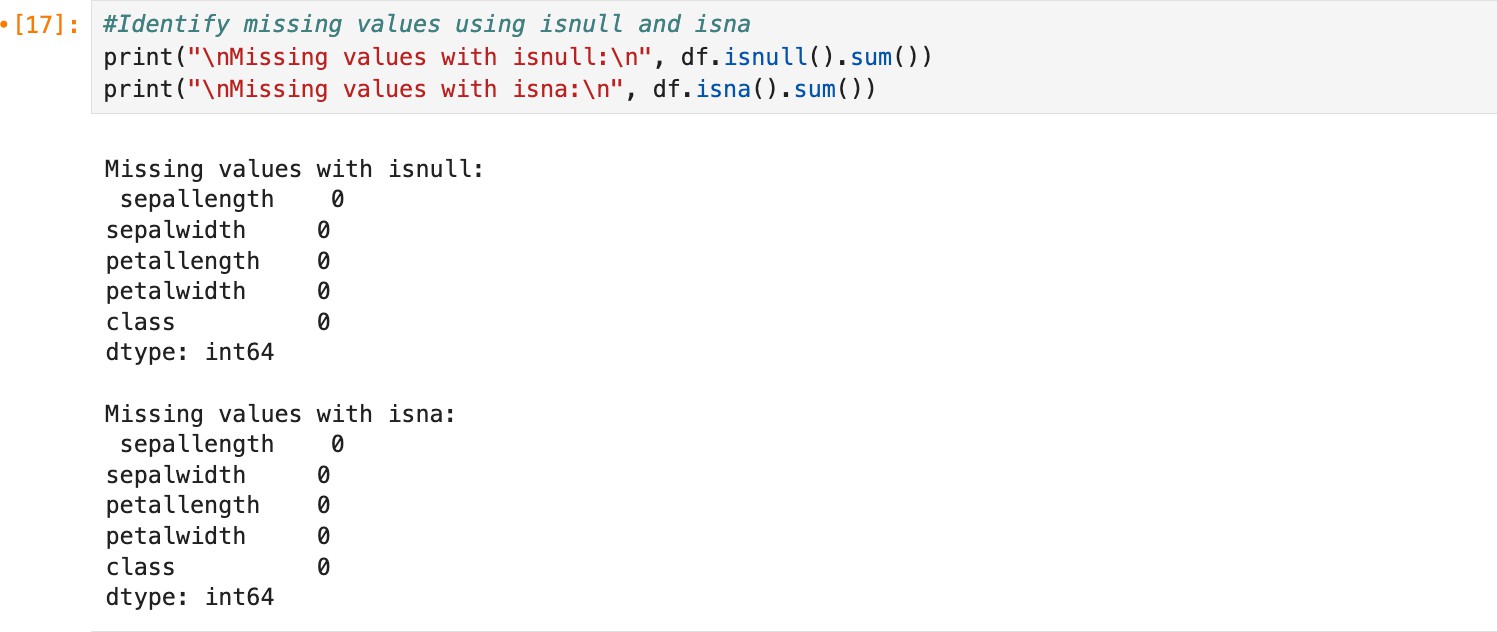
df = pd.get\_dummies(df, columns=categorical\_cols, drop\_first=True)

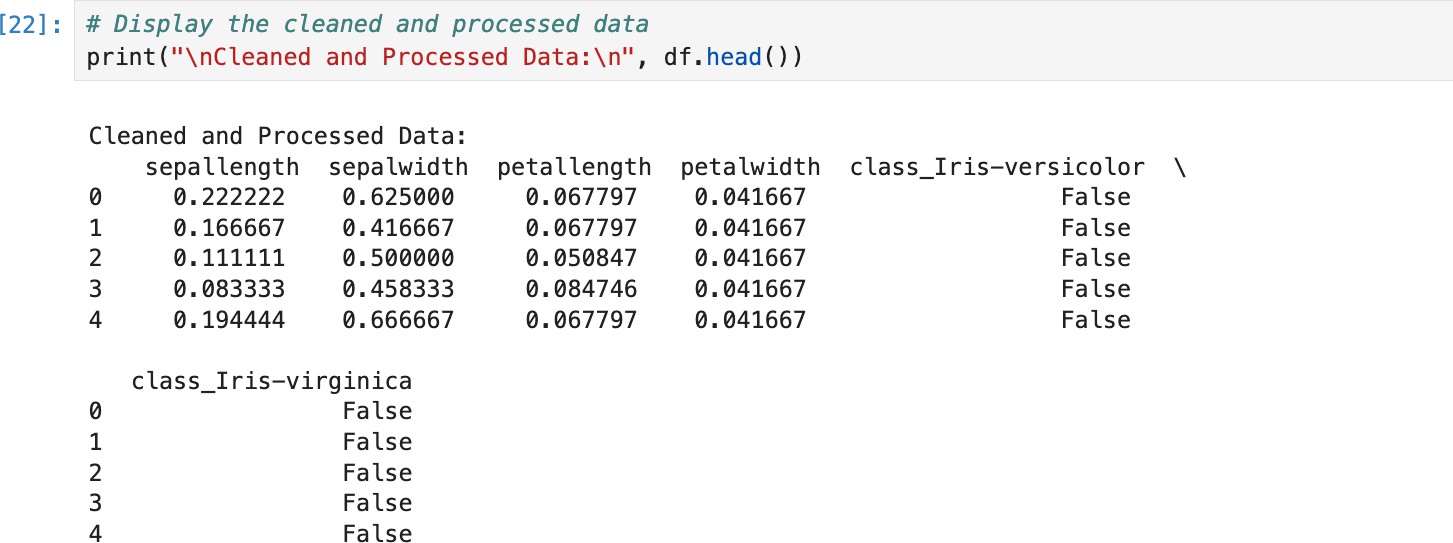
# # Display the cleaned and processed data

print("\nCleaned and Processed Data:\n", df.head()

**Output-**







**Q3. Aggregation and Grouping:**

Calculate summary statistics (mean, median, count, etc.) using groupby.

Create pivot tables for data summarization.

Combine DataFrames using concat, merge, and join.

Practice different join types (inner, outer, left, right).

import pandas as pd

data = {

'Name': ['John', 'Anna', 'Peter', 'Linda', 'James', 'Susan'],

'Department': ['HR', 'Finance', 'IT', 'Marketing', 'HR', 'Finance'],

'Salary': [50000, 60000, 55000, 52000, 49000, 62000],

'Age': [28, 22, 35, 32, 58, 29],

'City': ['New York', 'Paris', 'Berlin', 'London', 'Tokyo', 'New York']

}

df = pd.DataFrame(data)

grouped = df.groupby('Department')

mean\_salary = grouped['Salary'].mean()

median\_salary = grouped['Salary'].median()

count\_by\_dept = grouped['Name'].count()

print("\nMean Salary by Department:\n", mean\_salary)

print("\nMedian Salary by Department:\n", median\_salary)

print("\nCount of Employees by Department:\n", count\_by\_dept)

agg\_stats = grouped['Salary'].agg(['mean', 'median', 'count'])

print("\nAggregated Statistics for Salary by Department:\n", agg\_stats)

pivot\_table = pd.pivot\_table(df, values='Salary', index='Department', columns='City', aggfunc='mean', fill\_value=0)

print("\nPivot Table of Average Salary by Department and City:\n", pivot\_table)

data1 = {

'Name': ['John', 'Anna', 'Peter'],

'Department': ['HR', 'Finance', 'IT'],

'Salary': [50000, 60000, 55000]

}

data2 = {

'Name': ['Linda', 'James', 'Susan'],

'Department': ['Marketing', 'HR', 'Finance'],

'Salary': [52000, 49000, 62000]

}

df1 = pd.DataFrame(data1)

df2 = pd.DataFrame(data2)

concat\_df = pd.concat([df1, df2])

print("\nConcatenated DataFrame:\n", concat\_df)

data\_extra = {

'Name': ['John', 'Anna', 'Peter', 'Linda', 'James', 'Susan'],

'Age': [28, 22, 35, 32, 58, 29],

'City': ['New York', 'Paris', 'Berlin', 'London', 'Tokyo', 'New York']

}

df\_extra = pd.DataFrame(data\_extra)

merged\_inner = pd.merge(df1, df\_extra, on='Name')

print("\nInner Merge on 'Name':\n", merged\_inner)

merged\_left = pd.merge(df1, df\_extra, on='Name', how='left')

print("\nLeft Join on 'Name':\n", merged\_left)

merged\_right = pd.merge(df1, df\_extra, on='Name', how='right')

print("\nRight Join on 'Name':\n", merged\_right)

merged\_outer = pd.merge(df1, df\_extra, on='Name', how='outer')

print("\nOuter Join on 'Name':\n", merged\_outer)

df1.set\_index('Name', inplace=True)

df\_extra.set\_index('Name', inplace=True)

joined\_df = df1.join(df\_extra, how='inner')

print("\nDataFrame Join on 'Name' (inner join):\n", joined\_df)

Output-

