LAB No 11

Agglomerative Hierarchical Clustering

In this lab, students will learn how to perform Agglomerative Hierarchical Clustering (AHC), a method used to group similar data points into clusters. The lab involves:

- Understanding hierarchical clustering and dendrograms.
- Performing clustering on datasets using Python (scikit-learn, scipy).
- Visualizing clusters using dendrograms.
- Interpreting the clustering results for practical data analysis.

Objectives:

1. Understand hierarchical clustering concepts and linkage methods.
2. Perform agglomerative clustering on sample datasets.
3. Visualize the clustering process using dendrograms.
4. Analyze cluster assignments and validate results.

Theory

1. Introduction to Hierarchical Clustering

Hierarchical clustering is an **unsupervised learning** method that builds a hierarchy of clusters. It can be:

- **Agglomerative (bottom-up)**:
  Each observation starts as its own cluster, and pairs of clusters are merged step by step until only one cluster remains.
- **Divisive (top-down)**:
  Start with all observations in one cluster and recursively split them into smaller clusters.

2. Agglomerative Hierarchical Clustering

- Start with each data point as a separate cluster.
- Compute a **distance matrix** between all clusters.
- Merge the **two closest clusters** at each step.
- Repeat until all points belong to a single cluster.

Distance Metrics:

- **Euclidean Distance:** Most common for continuous data.
- **Manhattan Distance:** Sum of absolute differences.
- **Cosine Distance:** Measures angular distance for high-dimensional data.

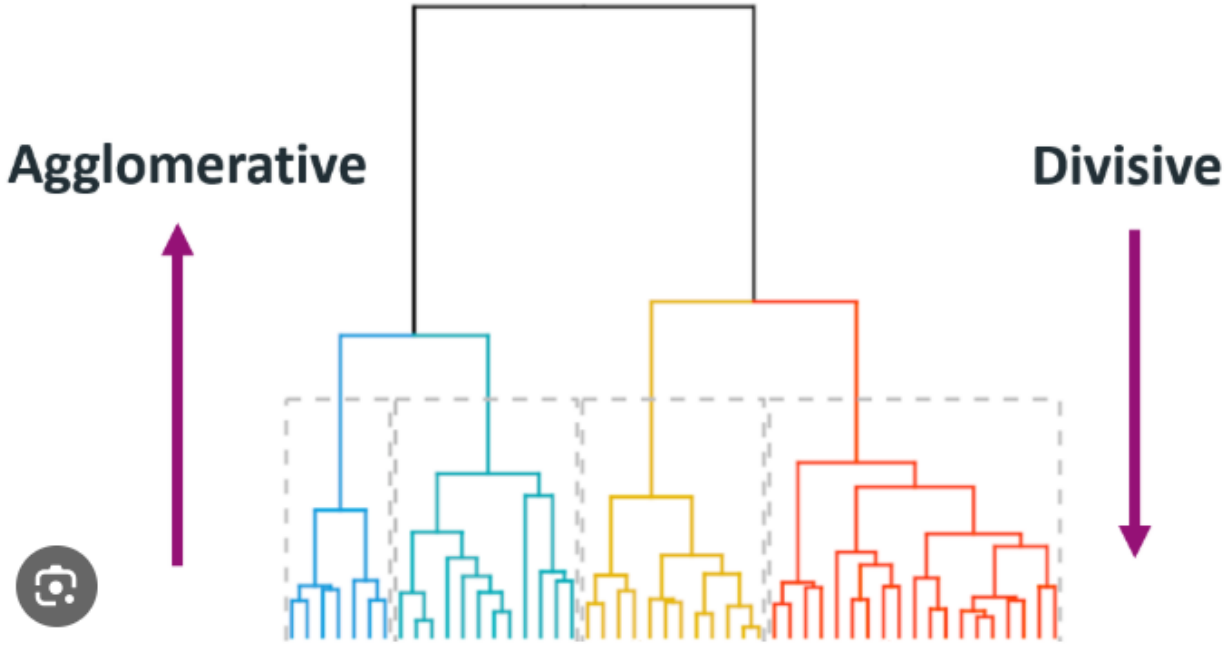
Linkage Methods:

- **Single Linkage:** Distance between closest points of two clusters.
- **Complete Linkage:** Distance between farthest points of two clusters.
- **Average Linkage:** Average distance between all points in two clusters.
- **Ward's Method:** Minimizes variance within clusters.

3. Dendrogram

A **dendrogram** is a tree-like diagram showing the order of cluster merges. It helps to:

- Visualize the hierarchy of clusters.
- Decide the optimal number of clusters by cutting the dendrogram.



4. Applications

- Customer segmentation in marketing.
- Document clustering in NLP.
- Gene expression analysis in bioinformatics.
- Image segmentation.

Python Libraries Required

```
import numpy as np
import pandas as pd
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

Solved Examples

Example 1: Clustering Simple 2D Points

Dataset:

```
data = np.array([[1, 2], [2, 3], [5, 8], [6, 9], [10, 12]])
```

Solution

```
# Step 1: Import libraries
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster
import matplotlib.pyplot as plt

# Step 2: Linkage matrix
Z = linkage(data, method='ward')  # Using Ward's method

# Step 3: Plot dendrogram
plt.figure(figsize=(6,4))
dendrogram(Z)
plt.title("Dendrogram - Example 1")
plt.show()

# Step 4: Form clusters (choose 2 clusters)
clusters = fcluster(Z, t=2, criterion='maxclust')
print("Cluster assignments:", clusters)
```

Output:

```
less                                                    Copy code

Cluster assignments: [1 1 2 2 2]
```

Explanation: The first two points are grouped together; the last three points form the second cluster.

Example 2: Agglomerative Clustering on Random Dataset

Solution

```
from sklearn.datasets import make_blobs
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

# Generate random data
X, _ = make_blobs(n_samples=8, centers=3, random_state=42)

# Linkage
Z = linkage(X, method='complete')
```

```
# Dendrogram
plt.figure(figsize=(6,4))
dendrogram(Z)
plt.title("Dendrogram - Example 2")
plt.show()

# Form clusters
clusters = fcluster(Z, t=3, criterion='maxclust')
print("Cluster assignments:", clusters)
```

Explanation: The dendrogram shows three distinct clusters; cluster labels indicate the group each point belongs to.

Example 3: Agglomerative Clustering on Iris Dataset (subset)

Solution

```
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster
import matplotlib.pyplot as plt

# Load Iris dataset
iris = load_iris()
X = iris.data[:, :2]  # Use only sepal length and width
X = StandardScaler().fit_transform(X)

# Linkage
Z = linkage(X, method='average', metric='euclidean')

# Plot dendrogram
plt.figure(figsize=(8,5))
dendrogram(Z)
plt.title("Dendrogram - Iris Example")
plt.show()

# Form clusters (3 clusters)
clusters = fcluster(Z, t=3, criterion='maxclust')
print("Cluster assignments:", clusters)
```

Explanation:

- Standardization is important to normalize features.
- The dendrogram helps to visualize clusters of similar iris species.
- fcluster assigns each data point to a cluster.

LAB Assignment No. 11

Question 1:

Perform Agglomerative Clustering with Different Linkages.

Task:

Load the "shopping-data.csv" dataset, extract the features *Annual Income* and *Spending Score*, and perform **Agglomerative Clustering** using:

- linkage = "ward"
- linkage = "complete"
- linkage = "average"

Instructions:

1. Perform clustering using AgglomerativeClustering.
2. Plot the clusters using matplotlib.
3. Compare how the cluster structure changes with each linkage method.

Code:

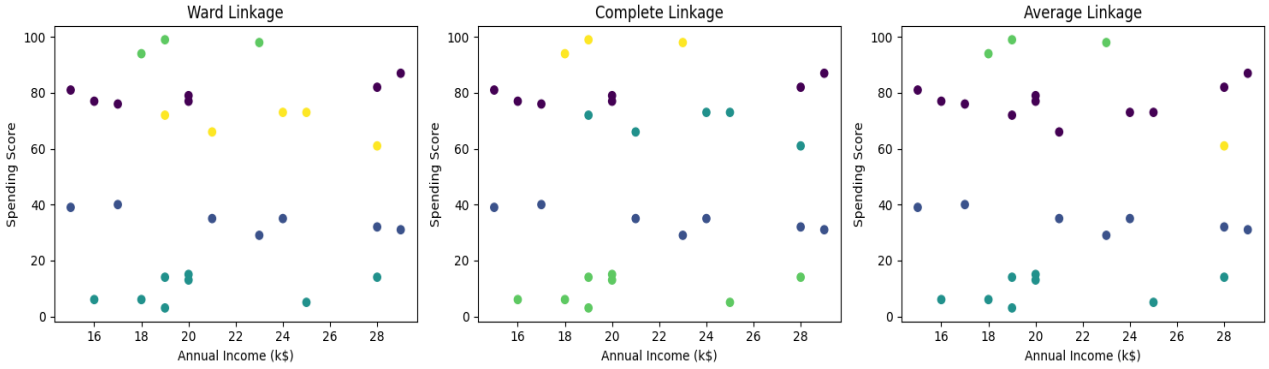
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import AgglomerativeClustering
data = pd.read_csv("shopping-data.csv")
X = data[['Annual Income (k$)', 'Spending Score (1-100)']]


ward = AgglomerativeClustering(n_clusters=5, linkage='ward')
complete = AgglomerativeClustering(n_clusters=5, linkage='complete')
average = AgglomerativeClustering(n_clusters=5, linkage='average')


labels_ward = ward.fit_predict(X)
labels_complete = complete.fit_predict(X)
labels_average = average.fit_predict(X)
```

```
plt.figure(figsize=(15,4))
plt.subplot(1,3,1)
plt.scatter(X.iloc[:,0], X.iloc[:,1], c=labels_ward)
plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score")
plt.title("Ward Linkage")
plt.subplot(1,3,2)
plt.scatter(X.iloc[:,0], X.iloc[:,1], c=labels_complete)
plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score")
plt.title("Complete Linkage")
```

Output:



Question 2:

Draw a Dendrogram and Identify the Optimal Number of Clusters

Task:

Using the same dataset or any synthetic dataset, draw a **dendrogram** using:

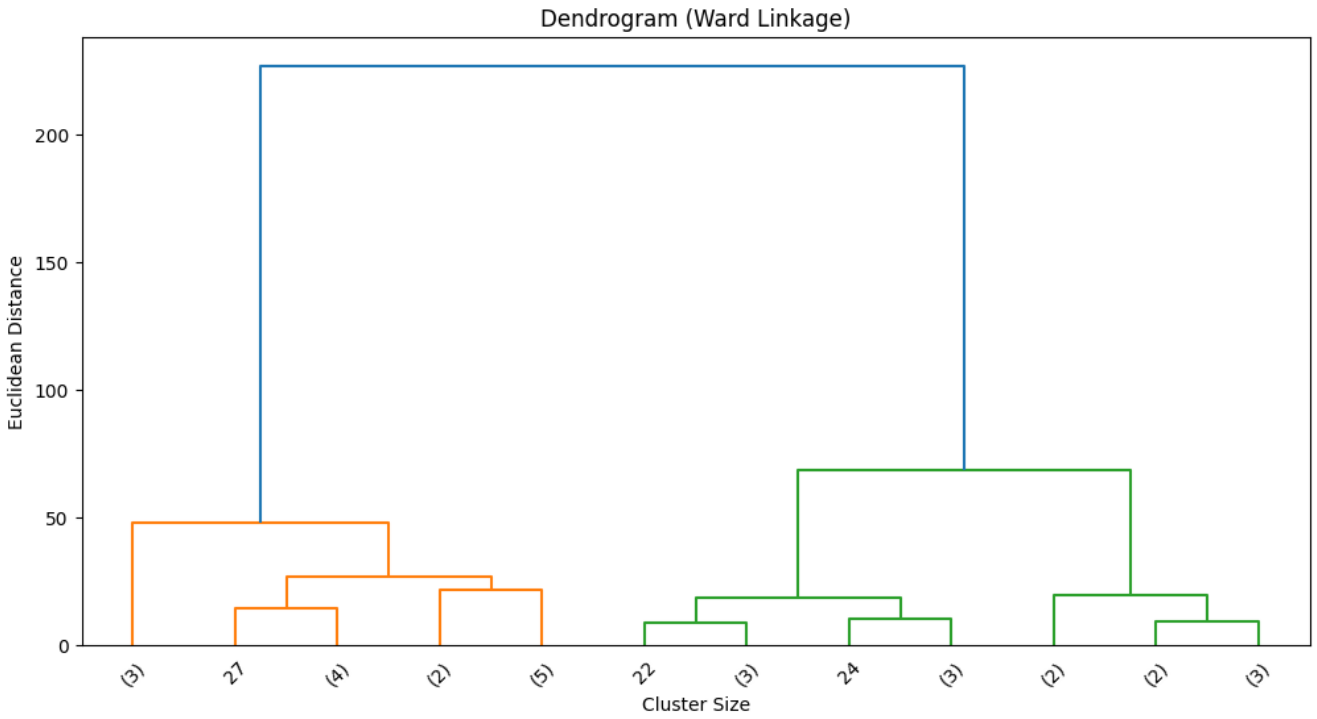
```
from scipy.cluster.hierarchy import dendrogram, linkage
```

Instructions:

1. Fit the data using linkage(method='ward').
2. Plot a dendrogram.
3. From the dendrogram, visually determine:
    - The optimal number of clusters
    - The height at which clusters merge
4. Explain why hierarchical clustering may be preferred over K-Means.

```
import pandas as pd
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
data = pd.read_csv("shopping-data.csv")
X = data[['Annual Income (k$)', 'Spending Score (1-100)']]
linked = linkage(X, method='ward')
plt.figure(figsize=(12,6))
dendrogram(
    linked,
    truncate_mode='lastp',
    p=12,
    leaf_rotation=45,
    leaf_font_size=10
)
plt.title("Dendrogram (Ward Linkage)")
plt.xlabel("Cluster Size")
plt.ylabel("Euclidean Distance")
plt.show()
```

Output:



Question 3: Compare Agglomerative vs Divisive Hierarchical Clustering

Task:
Using a small synthetic dataset (e.g., 10–12 points), perform:

- Agglomerative Clustering
- Divisive Clustering (manual split or using a library like sklearn-extra)

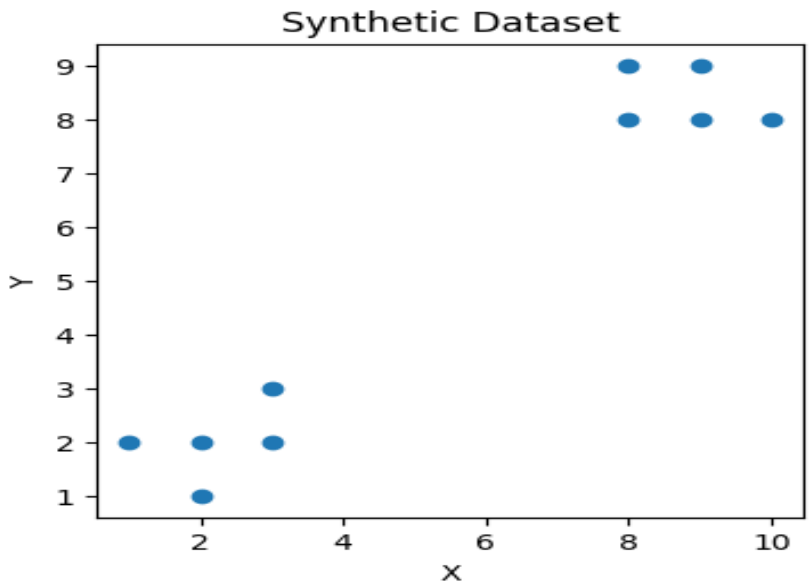
Instructions:

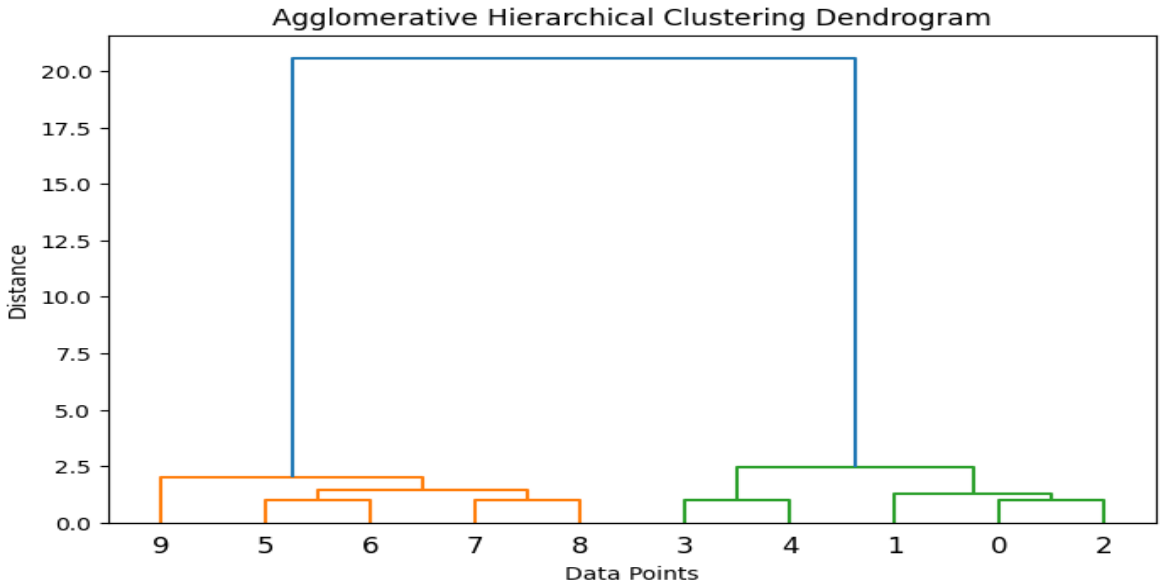
1. Plot dendrograms for both methods.
2. Compare the merge/split patterns.
3. Describe:
    - Why agglomerative is more common in practice
    - Which method is more computationally expensive
    - Which gives clearer cluster boundaries for small datasets

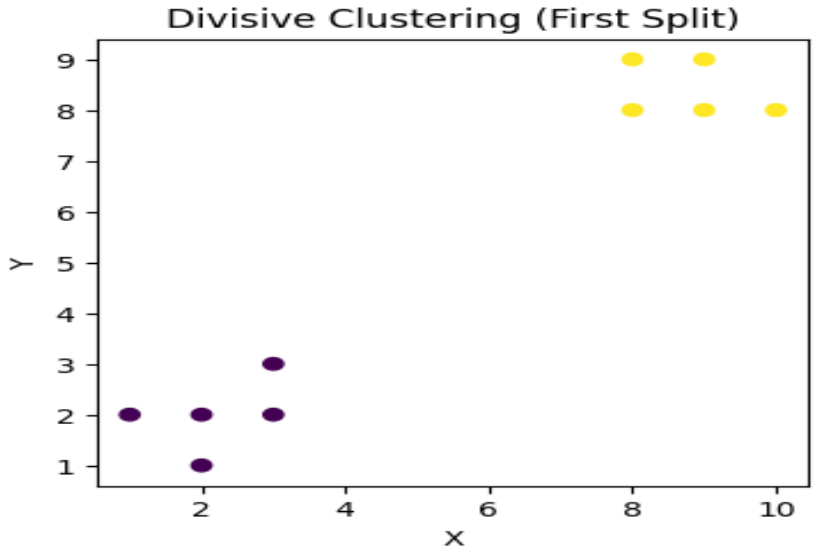
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import KMeans
X = np.array([
   [1,2],[2,1],[2,2],[3,2],[3,3],
   [8,8],[9,8],[8,9],[9,9],[10,8]])
plt.figure(figsize=(4,4))
plt.scatter(X[:,0], X[:,1])
plt.title("Synthetic Dataset")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
Z_agg = linkage(X, method='ward')
plt.figure(figsize=(8,5))
dendrogram(Z_agg)
plt.title("Agglomerative Hierarchical Clustering Dendrogram")
```

```
plt.xlabel("Data Points")
plt.ylabel("Distance")
plt.show()
kmeans = KMeans(n_clusters=2, random_state=42)
labels = kmeans.fit_predict(X)
plt.figure(figsize=(4,4))
plt.scatter(X[:,0], X[:,1], c=labels)
plt.title("Divisive Clustering (First Split)")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

Output:







LAB Assessment

Student Name		**LAB Rubrics**	CLO3 , P5, PLO5
		Total Marks	10
Registration No		**Obtained Marks**	
		Teacher Name	Dr. Syed M Hamedoon
Date		**Signature**	

Laboratory Work Assessment Rubrics

Sr. No.	Performance Indicator	Excellent (5)	Good (4)	Average (3)	Fair (2)	Poor (1)
1	**Theoretical knowledge 10%**	Student knows all the related concepts about the theoretical background of the experiment and rephrase those concepts in written and oral assessments	Student knows most of the related concepts about the theoretical background of the experiment and partially rephrase those concepts in written and oral assessments	Student knows few of the related concepts about the theoretical background of the experiment and partially rephrase those concepts in written and oral assessments	Student knows very little about the related concepts about the theoretical background of the experiment and poorly rephrase those concepts in written and oral assessments	Student has poor understanding of the related concepts about the theoretical background of the experiment and unable to rephrase those concepts in written and oral assessments
2	**Application Functionality 10%**	Application runs smoothly and operation of the application runs efficiently	Application compiles with no warnings. Robust operation of the application, with good recovery.	Application compiles with few or no warnings. Consideration given to unusual conditions with reasonable	Application compiles and runs without crashing. Some attempt at detecting and correcting errors.	Application does not compile or compiles but crashes. Confusing. Little or no error detection or correction.
3	**Specifications 10%**	The program works very efficiently and meets all of the required specifications.	The program works and meets some of the specifications.	The program works and produces the correct results and displays them correctly. It also meets most of the other specifications.	The program produces correct results but does not display them correctly.	The program is producing incorrect results.
4	**Level of understanding of the learned skill 10%**	Provide complete and logical answers based upon accurate technical content to the questions asked by examiner	Provide complete and logical answers based upon accurate technical content to the questions asked by examiner with few errors	Provide partially correct and logical answers based upon minimum technical content to the questions asked by examiner	Provide very few and illogical answers to the questions asked by examiner.	Provide no answer to the questions asked by examiner.
5	**Readability and Reusability 10%**	The code is exceptionally well organized and very easy to follow and reused	The code is fairly easy to read. The code could be reused as a whole or each class could be reused.	Most of the code could be reused in other programs.	Some parts of the code require change before they could be reused in other programs.	The code is poorly organized and very difficult to read and not organized for reusability.

6	**AI System Design 10%**	Well-designed AI models. Code is highly maintainable	Good designed AI models and Little code duplications	Some attempt to make AI models. Code can be maintained with significant effort	Little attempt to design AI models and less understanding of code	Very poor attempt to design AI models and its code
7	**Responsiveness to Questions/ Accuracy 10%**	1. Responds well, quick and very accurate all the time. 2. Effectively uses eye contact, speaks clearly, effectively and confidently using suitable volume	1. Generally Responsive and accurate most of the times. 2. Maintains eye contact, speaks clearly with suitable volume and pace.	1. Generally Responsive and accurate few times. 2. Some eye contact, speaks clearly and unclearly in different portions.	1. Not much Responsive and accurate most of the times. 2. Uses eye contact ineffectively and fails to speak clearly and audibly	. 1. Non Responsive and inaccurate all the times. 2. No eye contact and unable to speak 3. Dresses inappropriately
8	**Efficiency** **10%**	The code is extremely efficient without sacrificing readability and understanding	The code is fairly efficient without sacrificing readability and understanding	Some part of the code is efficient and other part of the code is not understandable and work properly	The code is brute force and unnecessarily long	The code is huge and appears to be patched together
9	**Delivery** **10%**	The program was delivered in time during lab.	The program was delivered in Lab before the end time.	The program was delivered within the due date.	The code was delivered within a day after the due date.	The code was delivered more than 2 days overdue.
10	**Awareness of Safety Guidelines 10%**	Student has sufficient knowledge of the laboratory safety SOPs and protocol and is fully compliant to the guidelines	Student has sufficient knowledge of the laboratory safety SOPs and protocol and is Partially compliant to the guidelines	Student has little knowledge of the laboratory safety SOPs and protocol and is Partially compliant to the guidelines	Student has little knowledge of the laboratory safety SOPs and protocol and is non-compliant to the guidelines	Student has no knowledge of the laboratory safety SOPs and protocol and is non-compliant to the guidelines