Practical No.1

Aim:- To create datasets from raw text files and import data (TXT) into SAS using SAS Studio.

Theory:- SAS reads data from many external sources like Text files. To use these files inside SAS, we must import them or read them using a data step.

Procedure:-

- Create .txt file with data
- Open SAS Studio
- Left panel → Server Files and Folders
- Choose the path where you save the .txt file
- Right-click → Upload
- Browse → Select .txt → Upload
- After uploading add code with the path of your file.

Code:-

Output:-

Total rows: 4 Total columns: 3 — Rows 1-4

	RollNo	Name	Marks
1	101	Yash	18
2	102	Tushar	19
3	103	Amit	20
4	104	Neha	18

Practical No.2

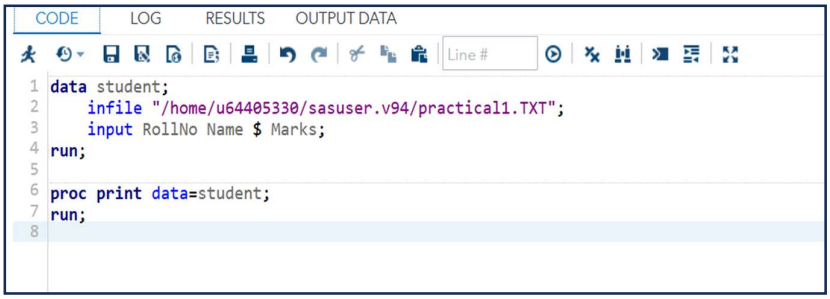
Aim:- To explore and display the contents of a SAS dataset using the PROC PRINT procedure.

Theory:- PROC PRINT is one of the most basic and powerful SAS procedures used to:

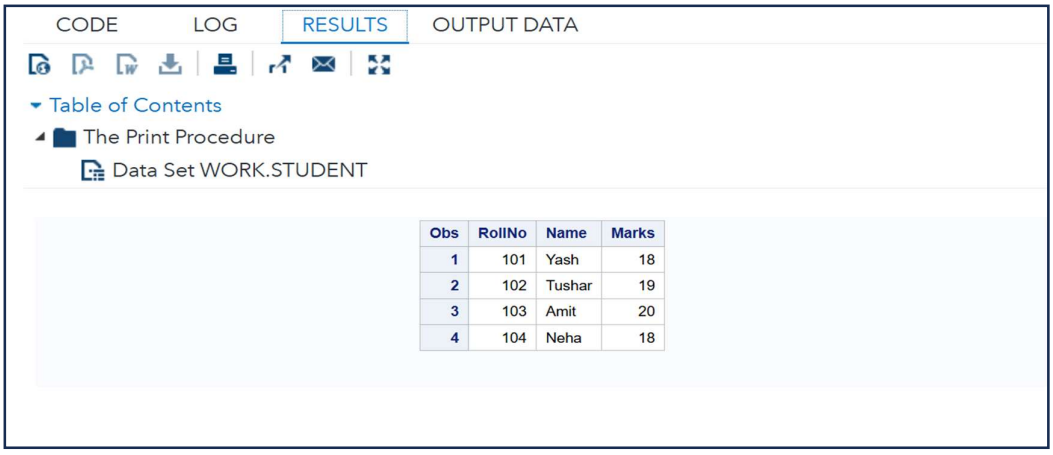
- Display all observations and variables in a dataset
- Show selected variables
- Apply labels
- Display only specific rows
- Sort results temporarily
- Check imported or created data

By default, PROC PRINT displays all rows and all columns.

Code:-



Output:-



Obs	RollNo	Name	Marks
1	101	Yash	18
2	102	Tushar	19
3	103	Amit	20
4	104	Neha	18

Practical No.3

Aim:- To filter and display SAS datasets using conditional statements: IF, IF-THEN, and WHERE.

Theory:- SAS provides several ways to filter data:

1. **IF statement**
   - Used **inside a DATA step**
   - Filters observations **while creating a dataset**
   - Syntax:
   - if condition then <action>;
2. **IF-THEN statement**
   - Used to **apply conditions and create new variables**
   - Example: Assign grades based on marks
3. **WHERE statement**
   - Used **in procedures (PROC)** or DATA steps
   - Filters observations **temporarily**, without creating a new dataset

Code for IF statement :-

CODE LOG RESULTS OUTPUT DATA

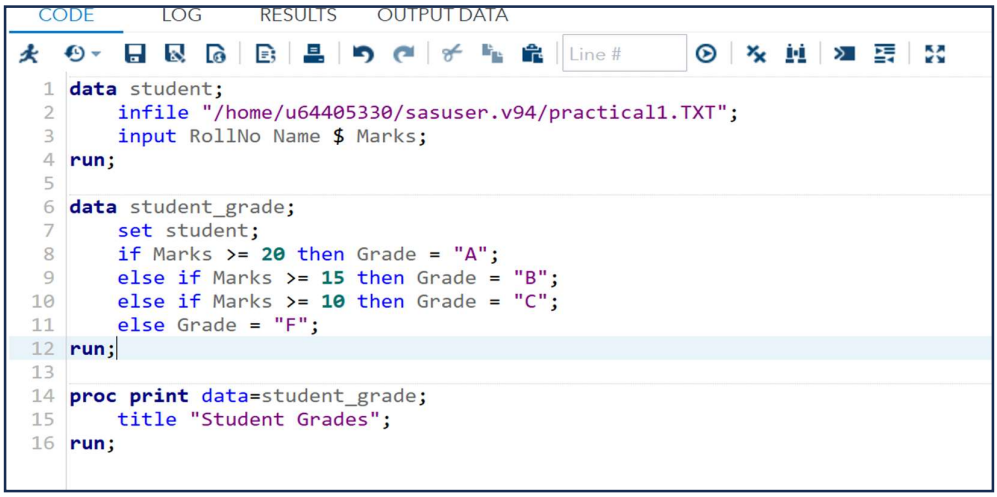
```
1  data student;
2      infile "/home/u64405330/sasuser.v94/practical1.TXT";
3      input RollNo Name $ Marks;
4  run;
5
6  data passed;
7      set student;
8      if Marks >= 15;  /* Only keep students with marks >= 60 */
9  run;
10
11 proc print data=passed;
12     title "Students Passed (Marks >= 15)";
13 run;
14
```

Output of IF statement:-

Students Passed (Marks >= 15)

Obs	RollNo	Name	Marks
1	101	Yash	18
2	102	Tushar	19
3	103	Amit	20
4	104	Neha	18

Code for IF-THEN statement :-

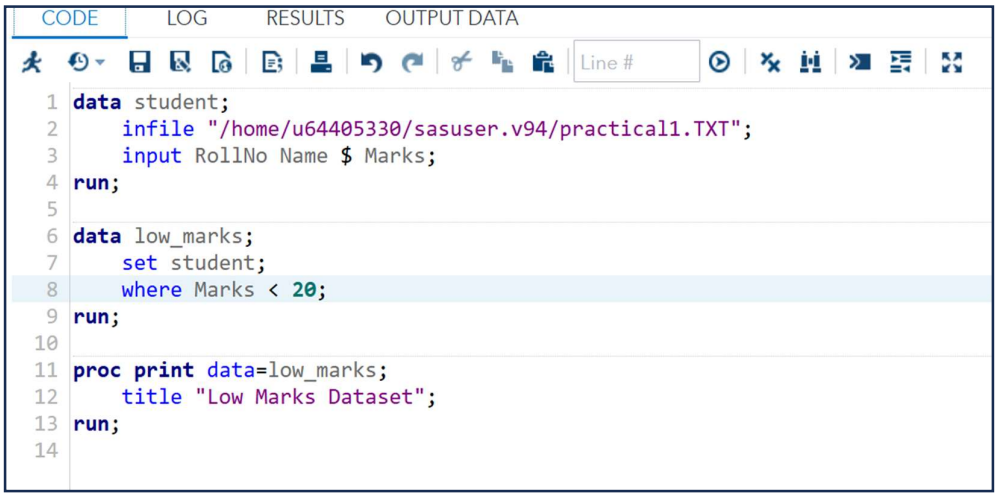


Output of IF-THEN statement:-

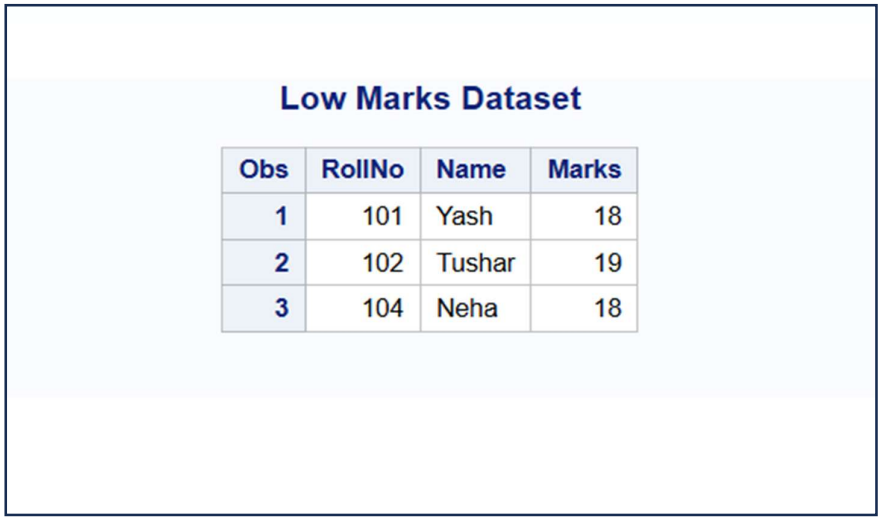
Student Grades

Obs	RollNo	Name	Marks	Grade
1	101	Yash	18	B
2	102	Tushar	19	B
3	103	Amit	20	A
4	104	Neha	18	B

Code for WHERE statement :-



Output of WHERE statement:-



Low Marks Dataset

Obs	RollNo	Name	Marks
1	101	Yash	18
2	102	Tushar	19
3	104	Neha	18

Practical No.4

Aim:- To sort a SAS dataset based on one or more variables in ascending or descending order using PROC SORT

Theory:-

- PROC SORT is used to **arrange data** in order.
- Sorting is often required **before using BY statements in PROC steps**.
- By default, PROC SORT sorts in **ascending order**.
- To sort in descending order, use the DESCENDING keyword.

Code of dataset in ascending order:-

CODE LOG RESULTS OUTPUT DATA

```
1  data student;
2      infile "/home/u64405330/sasuser.v94/practical1.TXT";
3      input RollNo Name $ Marks;
4  run;
5
6  proc sort data=student out=student_sorted;
7      by Marks;  /* Ascending order by Marks */
8  run;
9
10 proc print data=student_sorted;
11     title "Dataset Sorted by Marks (Ascending)";
12 run;
13
```

Output:-

Dataset Sorted by Marks (Ascending)

Obs	RollNo	Name	Marks
1	101	Yash	18
2	104	Neha	18
3	102	Tushar	19
4	103	Amit	20

Code of dataset in descending order:-

CODE LOG RESULTS OUTPUT DATA

```
1  data student;
2      infile "/home/u64405330/sasuser.v94/practical1.TXT";
3      input RollNo Name $ Marks;
4  run;
5
6  proc sort data=student out=student_sorted_desc;
7      by descending Marks;  /* Descending order by Marks */
8  run;
9
10 proc print data=student_sorted_desc;
11     title "Dataset Sorted by Marks (Descending)";
12 run;
13
14
```

Output:-

Dataset Sorted by Marks (Descending)

Obs	RollNo	Name	Marks
1	103	Amit	20
2	102	Tushar	19
3	101	Yash	18
4	104	Neha	18

Practical No.5

Aim:- To combine multiple SAS datasets using the MERGE statement and create a single dataset with all variables.

Theory:-

- **MERGE** is used to combine datasets **horizontally** (side by side).
- Datasets must be **sorted by the common variable(s)** before merging.
- The BY statement specifies the **key variable(s)** used to match observations.
- Only datasets with the same **BY variable** can be merged correctly.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
/* data 1 */
data marks;
    input RollNo Name $ Marks;
    datalines;
101 Rahul 78
102 Sneha 85
103 Amit 92
104 Priya 65
;
run;

/* data 2 */
data attendance;
    input RollNo DaysPresent;
    datalines;
101 180
102 190
103 175
104 160
;
run;

proc sort data=marks;
    by RollNo;
run;

proc sort data=attendance;
    by RollNo;
run;

data student;
    merge marks attendance;
    by RollNo;
run;

proc print data=student;
    title "Combined Dataset (Marks + Attendance)";
run;
```

Output:-

Combined Dataset (Marks + Attendance)

Obs	RollNo	Name	Marks	DaysPresent
1	101	Rahul	78	180
2	102	Sneha	85	190
3	103	Amit	92	175
4	104	Priya	65	160

Practical No.6

Aim:- To select or exclude specific variables from a dataset using KEEP and DROP statements in SAS.

Theory:-

- **KEEP** → Select only the variables you want to **retain** in the new dataset.
- **DROP** → Exclude variables you do **not want** in the new dataset.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data student;
2      infile "/home/u64406284/sasuser.v94/practical1.TXT";
3      input RollNo Name $ Marks;
4  run;
5
6  proc print data=student;
7  run;
8
9  proc print data=student(keep=Name Marks);
10     title "Print Only Name and Marks";
11 run;
12
13 proc print data=student(drop=RollNo);
14     title "Print Dataset Without RollNo";
15 run;
16
```

Output:-

Obs	RollNo	Name	Marks
1	101	Yash	18
2	102	Tushar	19
3	103	Amit	20
4	104	Neha	18

Print Only Name and Marks

Obs	Name	Marks
1	Yash	18
2	Tushar	19
3	Amit	20
4	Neha	18

Print Dataset Without RollNo

Obs	Name	Marks
1	Yash	18
2	Tushar	19
3	Amit	20
4	Neha	18

Practical No.7

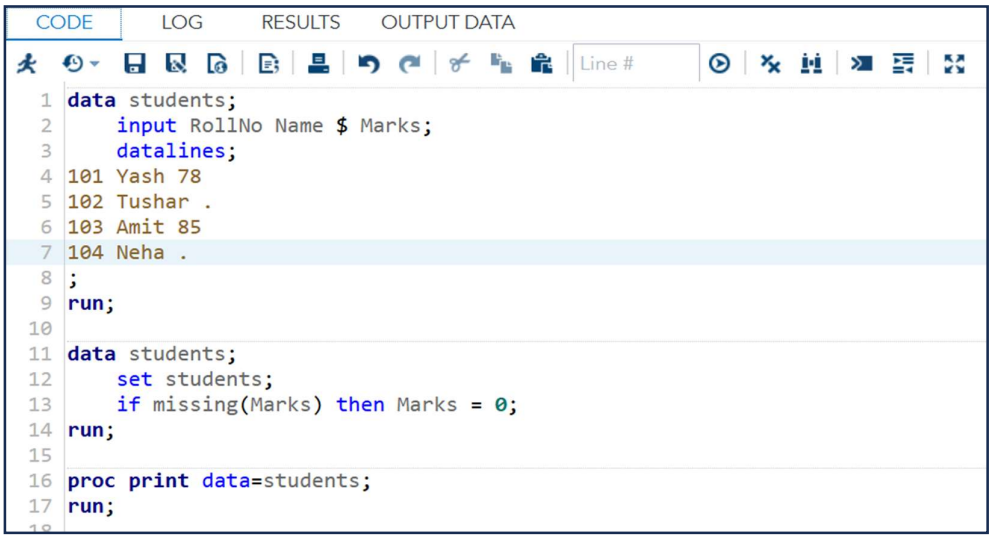
Aim:- To learn how to detect and handle missing values in SAS using the MISSING function and missing-value operators.

Theory:- In SAS, missing values are represented as:

- **(.)** for numeric variables
- **(")** (blank) for character variables

To check whether a variable contains a missing value, SAS provides:

Code:-



Output:-

Obs	RollNo	Name	Marks
1	101	Yash	78
2	102	Tushar	0
3	103	Amit	85
4	104	Neha	0

Practical No.8

Aim:- To perform text manipulation using the SUBSTR and SCAN functions in SAS.

Theory:- Text manipulation is an important task in SAS when working with character data. Two commonly used functions are:

1. SUBSTR (Substring Function)

- Used to extract a portion of a character string.
- Example: substr("Rohini", 1, 3) returns "Roh".

2. SCAN (Word Extraction Function)

- Used to extract a specific word from a text string based on delimiters.
- Default delimiters include space, comma, period, underscore, etc.
- Example: scan("Rohini_Patil", 1, "_") returns "Rohini".

:

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data student;
2      length Name $20;
3      infile "/home/u64406284/sasuser.v94/practical1.TXT";
4      input RollNo Name $ Marks;
5  run;
6
7  data text_output;
8      set student;
9
10     Part_Substr = substr(Name, 1, 5);          /* Extract first 5 characters */
11     First_Name  = scan(Name, 1, '_');          /* Extract first name */
12 run;
13
14 proc print data=text_output;
15 run;
```

Output:-

Obs	Name	RollNo	Marks	Part_Substr	First_Name
1	Yash_chavan	101	18	Yash_	Yash
2	Tushar_Gaikwad	102	19	Tusha	Tushar
3	Amit_Jadhav	103	20	Amit_	Amit
4	Neha_Kashyap	104	18	Neha_	Neha

Practical No.9

Aim:- To create new variables in SAS using arithmetic calculations and data transformations.

Theory:- In SAS, new variables can be created inside a DATA step using mathematical expressions or transformations. These new variables do not change the original dataset but are added to the output dataset.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
data student;
    input RollNo Name $ Marks1 Marks2;
    datalines;
101 Yash 45 40
102 Neha 38 42
103 Amit 50 48
104 Tushar 46 44
;
run;

data student_calc;
    set student;

    Total_Marks = Marks1 + Marks2;              /* Addition */
    Percentage  = (Total_Marks / 100) * 100;    /* Percentage calculation */
    Result      = ifc(Percentage >= 40, "Pass", "Fail");  /* Conditional transformation */
run;

proc print data=student_calc;
run;
```

Output:-

Obs	RollNo	Name	Marks1	Marks2	Total_Marks	Percentage	Result
1	101	Yash	45	40	85	85	Pass
2	102	Neha	38	42	80	80	Pass
3	103	Amit	50	48	98	98	Pass
4	104	Tushar	46	44	90	90	Pass

Practical No.10

Aim:- To reshape data from long format to wide format using PROC TRANSPOSE in SAS.

Theory:- PROC TRANSPOSE in SAS is used to rotate or pivot a dataset:

- **Long → Wide:-** Convert rows into columns.
- **Wide → Long:-** Convert columns into rows.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
data marks_long;
    input Name $ Subject $ Marks;
    datalines;
Amit Math 78
Amit Science 85
Amit English 90
Neha Math 92
Neha Science 95
Neha English 94
;
run;

proc transpose data=marks_long out=marks_wide;
    by Name;                  /* Group by student name */
    id Subject;               /* Make Subject names new columns */
    var Marks;                /* Values to transpose */
run;

proc print data=marks_wide;
run;
```

Output:-

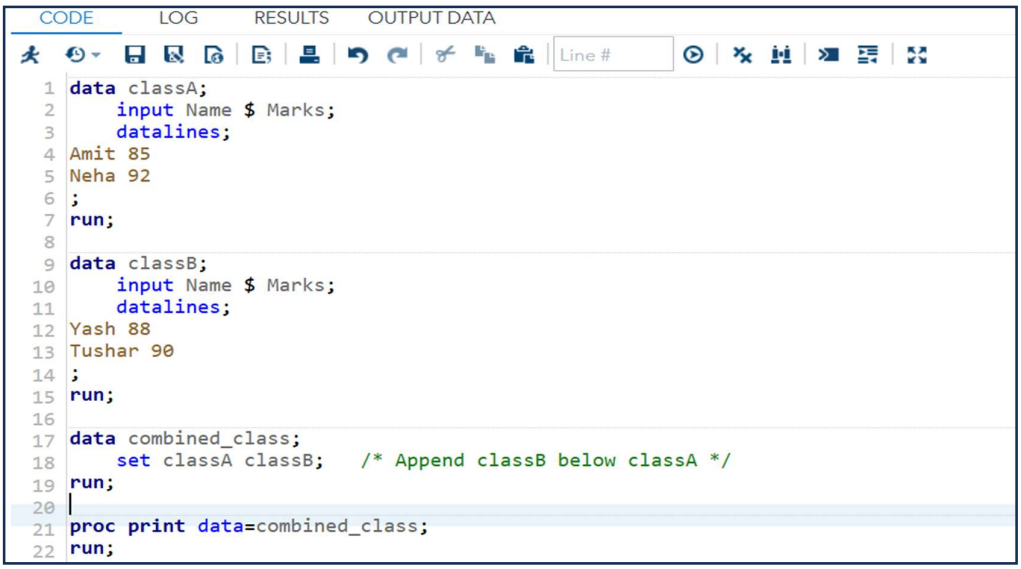
Obs	Name	_NAME_	Math	Science	English
1	Amit	Marks	78	85	90
2	Neha	Marks	92	95	94

Practical No.11

Aim:- To combine two or more datasets vertically (one below another) using the SET statement in SAS.

Theory:- Vertical combination means **adding rows** from multiple datasets to form **one bigger dataset**. In SAS, this is done using the **SET statement**:

Code:-



Output:-

Obs	Name	Marks
1	Amit	85
2	Neha	92
3	Yash	88
4	Tushar	90

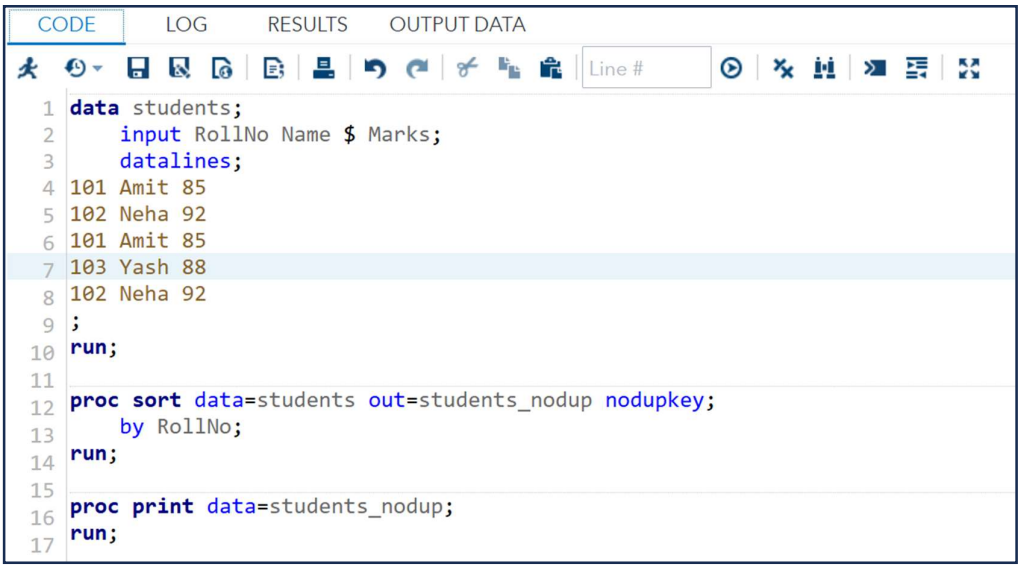
Practical No.12

Aim:- To identify and remove duplicate observations from a SAS dataset using PROC SORT with the NODUPKEY option.

Theory:- Duplicate records are rows with identical values for certain key variables. PROC SORT with the NODUPKEY option can remove duplicate rows based on BY variable(s).

Option	Description
NODUPKEY	Removes observations with duplicate BY values, keeping the first occurrence
BY	Key variable(s) used to check for duplicates
OUT	Name of output dataset (optional, defaults to overwriting input)

Code:-



Output:-

Obs	RollNo	Name	Marks
1	101	Amit	85
2	102	Neha	92
3	103	Yash	88

Practical No.13

Aim:- To extract components of a date such as day, month, year, and weekday using SAS date functions.

Theory:- SAS stores dates as the **number of days since January 1, 1960**. You can extract parts of a date using the following functions:

Function	Description
DAY(date)	Returns the day of the month (1–31)
MONTH(date)	Returns the month number (1–12)
YEAR(date)	Returns the year
WEEKDAY(date)	Returns the day of the week (1=Sunday, 7=Saturday)

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data attendance;
2      input Name $ JoinDate :date9.;
3      format JoinDate date9.;   /* Display date properly */
4      datalines;
5  Amit 01JAN2024
6  Neha 15FEB2024
7  Yash 28MAR2024
8  ;
9  run;
10
11 data attendance_components;
12     set attendance;
13
14     DayJoined    = day(JoinDate);       /* Day of month */
15     MonthJoined  = month(JoinDate);     /* Month number */
16     YearJoined   = year(JoinDate);      /* Year */
17     WeekdayJoined = weekday(JoinDate); /* Day of week (1=Sun) */
18 run;
19
20 proc print data=attendance_components;
21 run;
```

Output:-

Obs	Name	JoinDate	DayJoined	MonthJoined	YearJoined	WeekdayJoined
1	Amit	01JAN2024	1	1	2024	2
2	Neha	15FEB2024	15	2	2024	5
3	Yash	28MAR2024	28	3	2024	5

Practical No.14

Aim:- To display basic information about a SAS dataset, including variable names, types, lengths, and dataset attributes using PROC CONTENTS.

Theory:- PROC CONTENTS provides a summary of a dataset.

- **It shows details like:-** Dataset name and label, Number of observations and variables, Variable names, types (numeric/character), lengths, formats, Creation date and modification date
- **This procedure is useful for:-** Understanding dataset structure before analysis, Checking variable types, Identifying data issues

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data students;
2      input RollNo Name $ Marks;
3      datalines;
4  101 Amit 85
5  102 Neha 92
6  103 Yash 88
7  ;
8  run;
9
10 proc contents data=students;
11 run;
```

Output:-

The CONTENTS Procedure

Data Set Name	WORK.STUDENTS	Observations	3
Member Type	DATA	Variables	3
Engine	V9	Indexes	0
Created	04/12/2025 19:00:10	Observation Length	24
Last Modified	04/12/2025 19:00:10	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	utf-8 Unicode (UTF-8)		

Engine/Host Dependent Information	
Data Set Page Size	131072
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	5431
Obs in First Data Page	3
Number of Data Set Repairs	0
Filename	/saswork/SAS_work1FAB0001BB69_odaws01-apse1.oda.sas.com/SAS_work6C800001BB69_odaws01-apse1.oda.sas.com/students.sas7bdat
Release Created	9.0401M8
Host Created	Linux
Inode Number	536877284
Access Permission	rw-r--r--
Owner Name	u64406284
File Size	256KB
File Size (bytes)	262144

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
3	Marks	Num	8
2	Name	Char	8
1	RollNo	Num	8

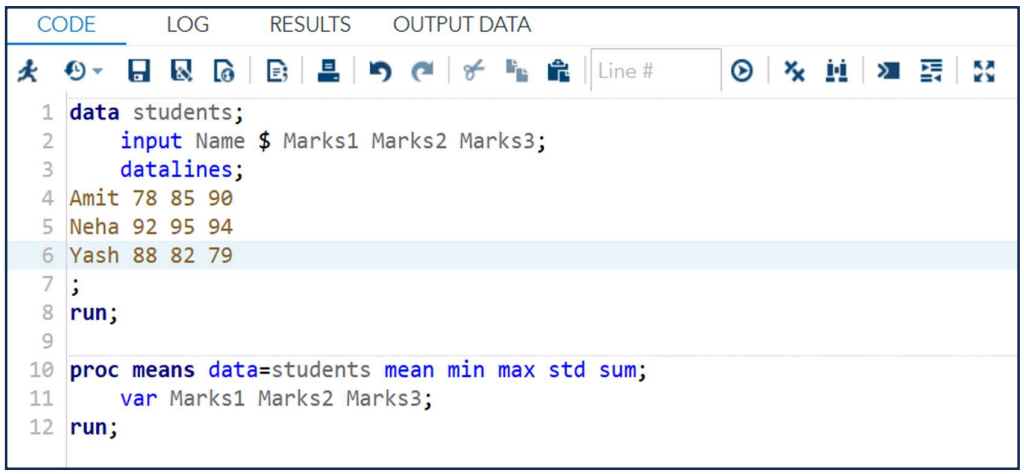
Practical No.15

Aim:- To generate basic descriptive statistics such as mean, minimum, maximum, standard deviation, and sum for numeric variables using PROC MEANS.

Theory:- PROC MEANS computes summary statistics for numeric variables.

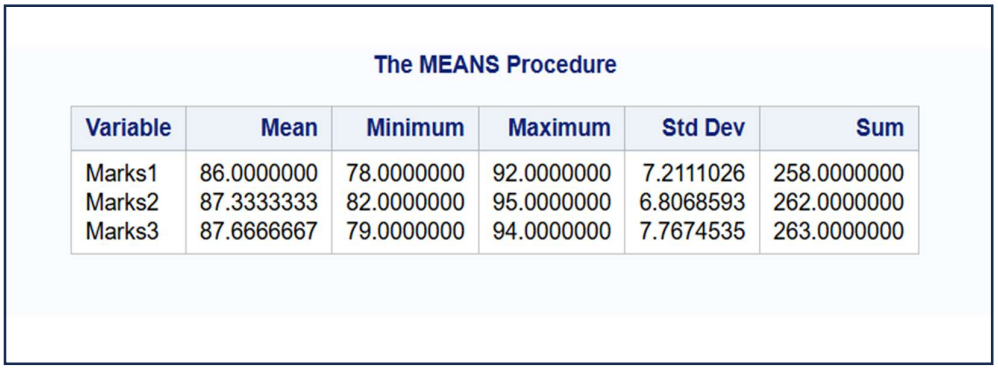
- Common statistics include:
  - N → Number of observations
  - Mean → Average
  - Min → Minimum value
  - Max → Maximum value
  - Std → Standard deviation
  - Sum → Total sum
  - You can specify variables with the VAR statement.

Code:-




```
1  data students;
2      input Name $ Marks1 Marks2 Marks3;
3      datalines;
4  Amit 78 85 90
5  Neha 92 95 94
6  Yash 88 82 79
7  ;
8  run;
9
10 proc means data=students mean min max std sum;
11     var Marks1 Marks2 Marks3;
12 run;
```

Output:-



The MEANS Procedure

Variable	Mean	Minimum	Maximum	Std Dev	Sum
Marks1	86.0000000	78.0000000	92.0000000	7.2111026	258.0000000
Marks2	87.3333333	82.0000000	95.0000000	6.8068593	262.0000000
Marks3	87.6666667	79.0000000	94.0000000	7.7674535	263.0000000

Practical No.16

Aim:- To generate frequency and percentage tables for categorical variables using PROC FREQ in SAS.

Theory:- PROC FREQ is used to summarize categorical data. It provides: Frequency → Number of occurrences of each value, Percent → Percentage of total observations, Cumulative Frequency / Percent → Optional. Useful for data exploration, reporting, and checking distributions.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data students;
2      input Name $ Gender $ Marks;
3      datalines;
4  Amit M 85
5  Neha F 92
6  Yash M 88
7  Tushar M 90
8  Neha F 95
9  ;
10 run;
11
12 proc freq data=students;
13     tables Gender;
14 run;
```

Output:-

The FREQ Procedure

Gender	Frequency	Percent	Cumulative Frequency	Cumulative Percent
F	2	40.00	2	40.00
M	3	60.00	5	100.00

Practical No.17

Aim:- To create cross-tabulations (two-way frequency tables) between two categorical variables using PROC FREQ in SAS.

Theory:- Cross-tabulations (contingency tables) show how two categorical variables relate. PROC FREQ can generate:- Frequency counts, Row / Column / Overall Percentages. Options:- **norow** → suppress row percentages, **nocol** → suppress column percentages, **nopercent** → suppress overall percentages

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data students;
2      input Name $ Gender $ Grade $;
3      datalines;
4  Amit M A
5  Neha F B
6  Yash M B
7  Tushar M A
8  Neha F A
9  ;
10 run;
11
12 proc freq data=students;
13     tables Gender*Grade;
14 run;
```

Output:-

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Gender by Grade		
	Grade		
Gender	A	B	Total
F	1 20.00 50.00 33.33	1 20.00 50.00 50.00	2 40.00
M	2 40.00 66.67 66.67	1 20.00 33.33 50.00	3 60.00
Total	3 60.00	2 40.00	5 100.00

Practical No.18

Aim:- To perform a one-sample t-test to check whether the mean of a numeric variable differs from a specified value using PROC TTEST.

Theory:- A one-sample t-test tests whether the mean of a numeric variable is significantly different from a hypothesized value.
Null Hypothesis (H_0): The population mean = specified valueAlternative
Hypothesis (H_1): The population mean ≠ specified value
PROC TTEST provides:- Sample mean, standard deviation, standard error, t-value, degrees of freedom, p-value, Confidence interval for the mean

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data students;
2      input Name $ Marks;
3      datalines;
4  Amit 85
5  Neha 92
6  Yash 88
7  Tushar 90
8  ;
9  run;
10
11 proc ttest data=students h0=85 sides=2;
12     var Marks;
13 run;
```

Output:-

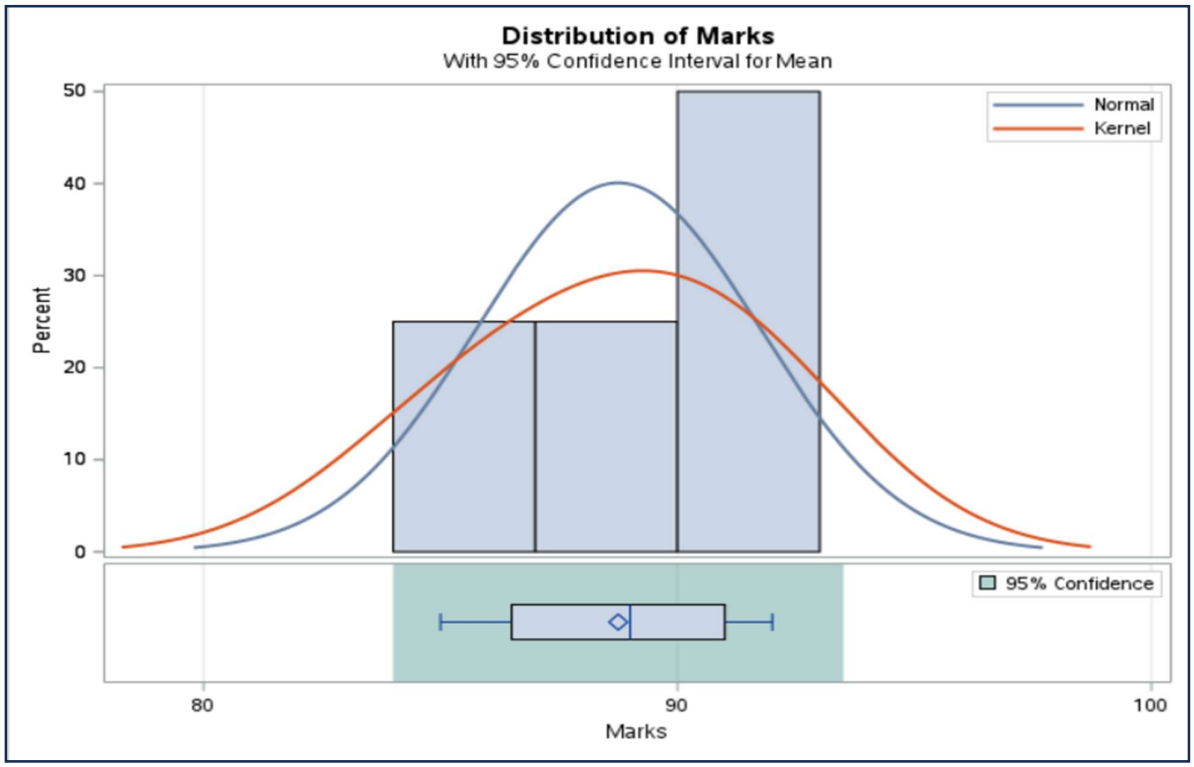
The TTEST Procedure

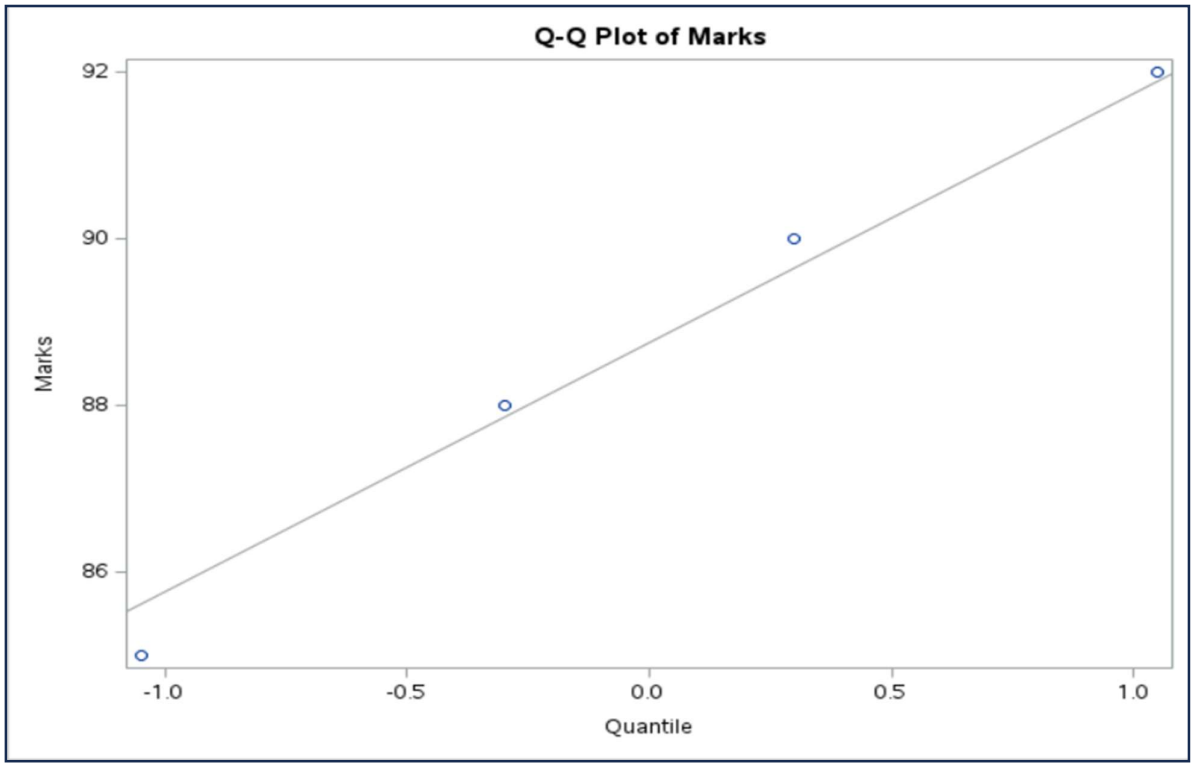
Variable: Marks

N	Mean	Std Dev	Std Err	Minimum	Maximum
4	88.7500	2.9861	1.4930	85.0000	92.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
88.7500	83.9985	93.5015	2.9861	1.6916	11.1337

DF	t Value	Pr > |t|
3	2.51	0.0868


Distribution of Marks
With 95% Confidence Interval for Mean
Normal
Kernel
95% Confidence
Percent
50
40
30
20
10
0
80
90
100
Marks


Q-Q Plot of Marks
Marks
92
90
88
86
-1.0
-0.5
0.0
0.5
1.0
Quantile

Practical No.19

Aim:- To perform an independent two-sample t-test to compare the means of a numeric variable between two groups using PROC TTEST.

Theory:- An independent two-sample t-test compares the means of a numeric variable between two independent groups. Null Hypothesis (H_0): $\mu_1 = \mu_2$ (means of the two groups are equal). Alternative Hypothesis (H_1): $\mu_1 \neq \mu_2$ (means are different).

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data students;
2      input Name $ Gender $ Marks;
3      datalines;
4  Amit M 85
5  Rohit M 88
6  Neha F 92
7  Tushar F 90
8  ;
9  run;
10
11 proc ttest data=students;
12     class Gender;    /* Grouping variable */
13     var Marks;       /* Numeric variable to compare */
14 run;
```

Output:-

The TTEST Procedure

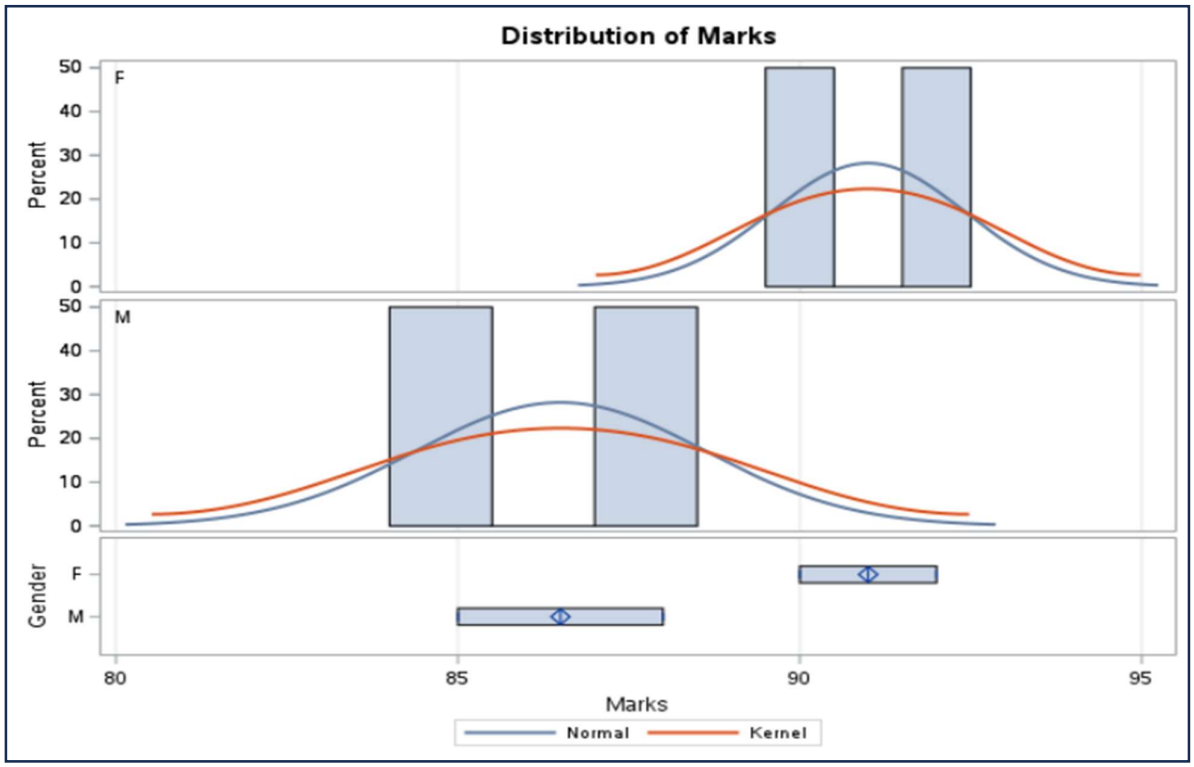
Variable: Marks

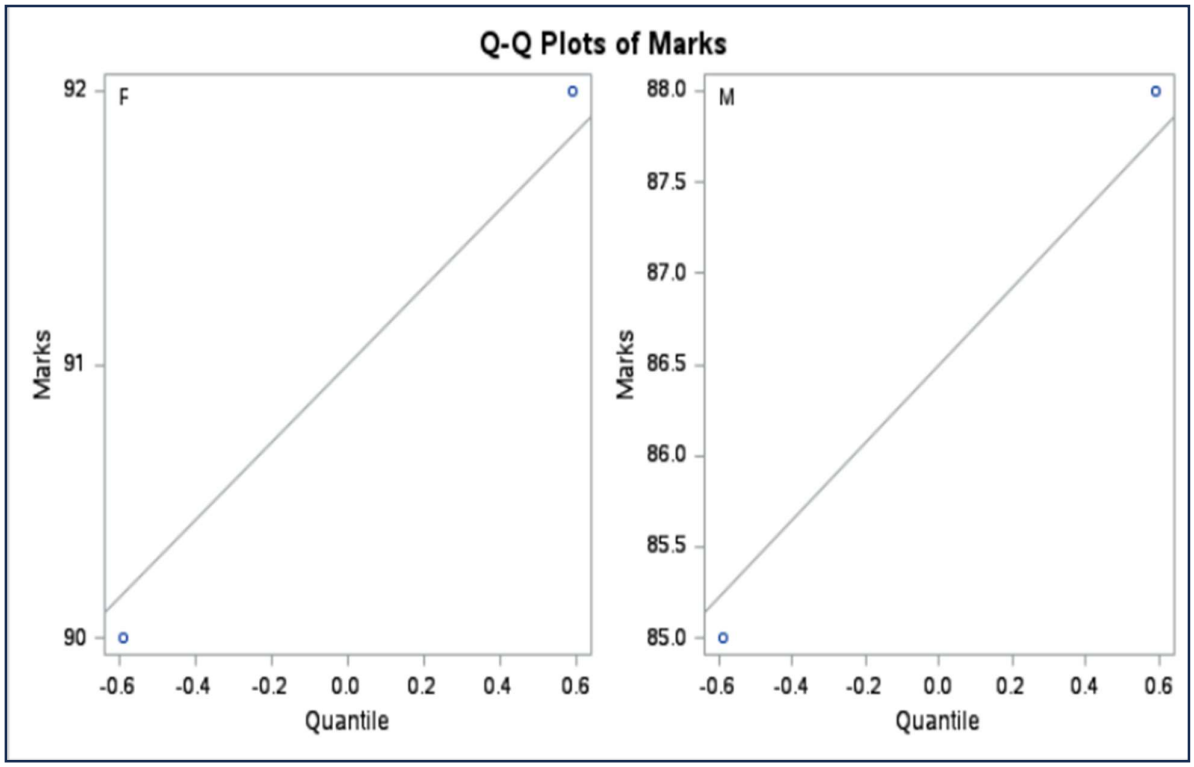
Gender	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
F		2	91.0000	1.4142	1.0000	90.0000	92.0000
M		2	86.5000	2.1213	1.5000	85.0000	88.0000
Diff (1-2)	Pooled		4.5000	1.8028	1.8028		
Diff (1-2)	Satterthwaite		4.5000		1.8028		

Gender	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
F		91.0000	78.2938	103.7	1.4142	0.6310	45.1278
M		86.5000	67.4407	105.6	2.1213	0.9464	67.6917
Diff (1-2)	Pooled	4.5000	-3.2567	12.2567	1.8028	0.9386	11.3300
Diff (1-2)	Satterthwaite	4.5000	-4.4671	13.4671			

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	2	2.50	0.1299
Satterthwaite	Unequal	1.7423	2.50	0.1481

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	1	1	2.25	0.7487


Distribution of Marks
F
M
Percent
50
40
30
20
10
0
Gender
F
M
80
85
90
95
Marks
Normal
Kernel

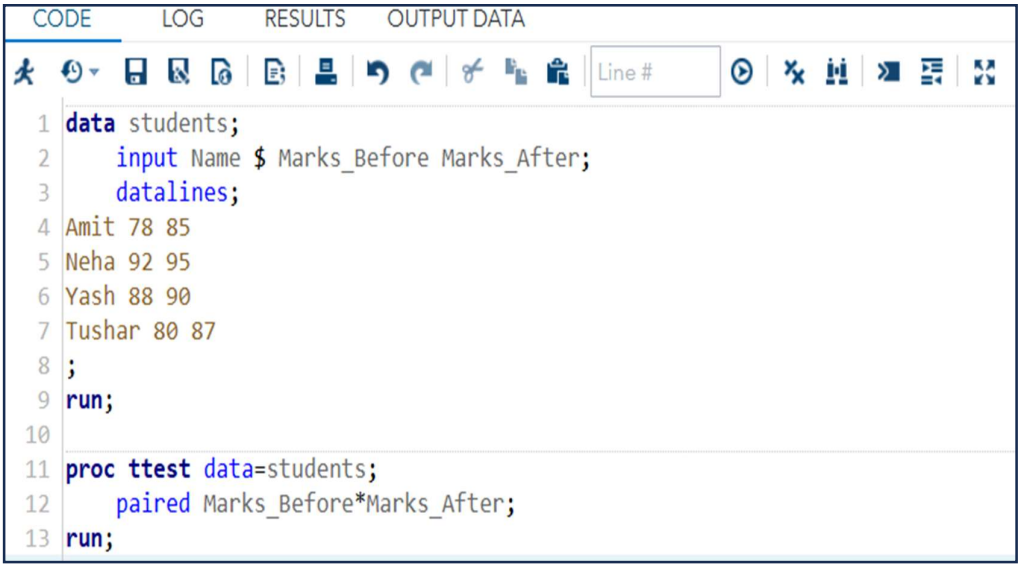

Q-Q Plots of Marks
F
M
Marks
92
91
90
88.0
87.5
87.0
86.5
86.0
85.5
85.0
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
Quantile

Practical No.20

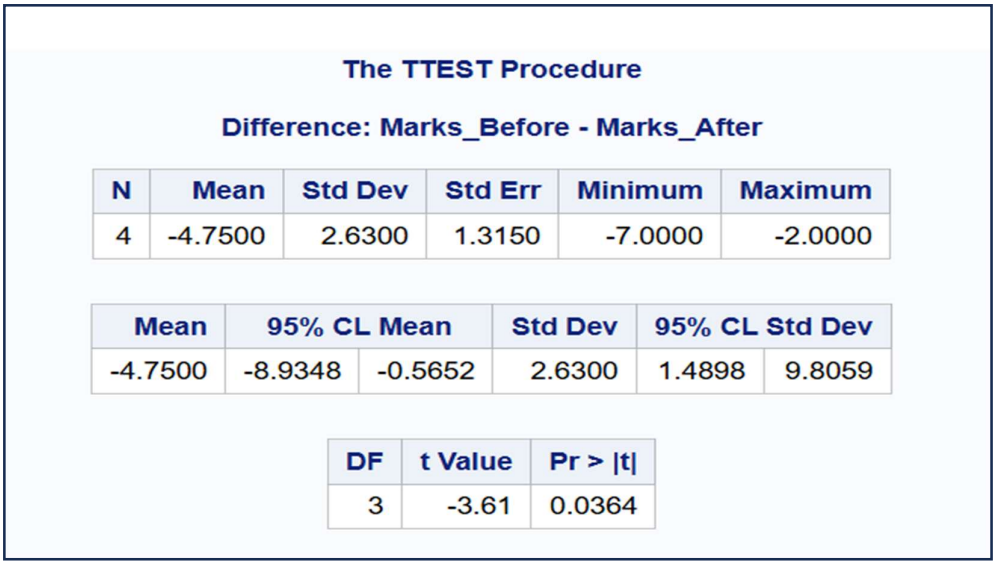
Aim:- To perform a paired t-test to compare the means of two related measurements using PROC TTEST.

Theory:- A paired t-test is used to check whether the mean difference between two related measurements, such as before-and-after observations, is significantly different from zero. SAS's PROC TTEST with the PAIRED statement calculates the mean difference, standard deviation, t-value, p-value, and confidence interval.

Code:-



Output:-



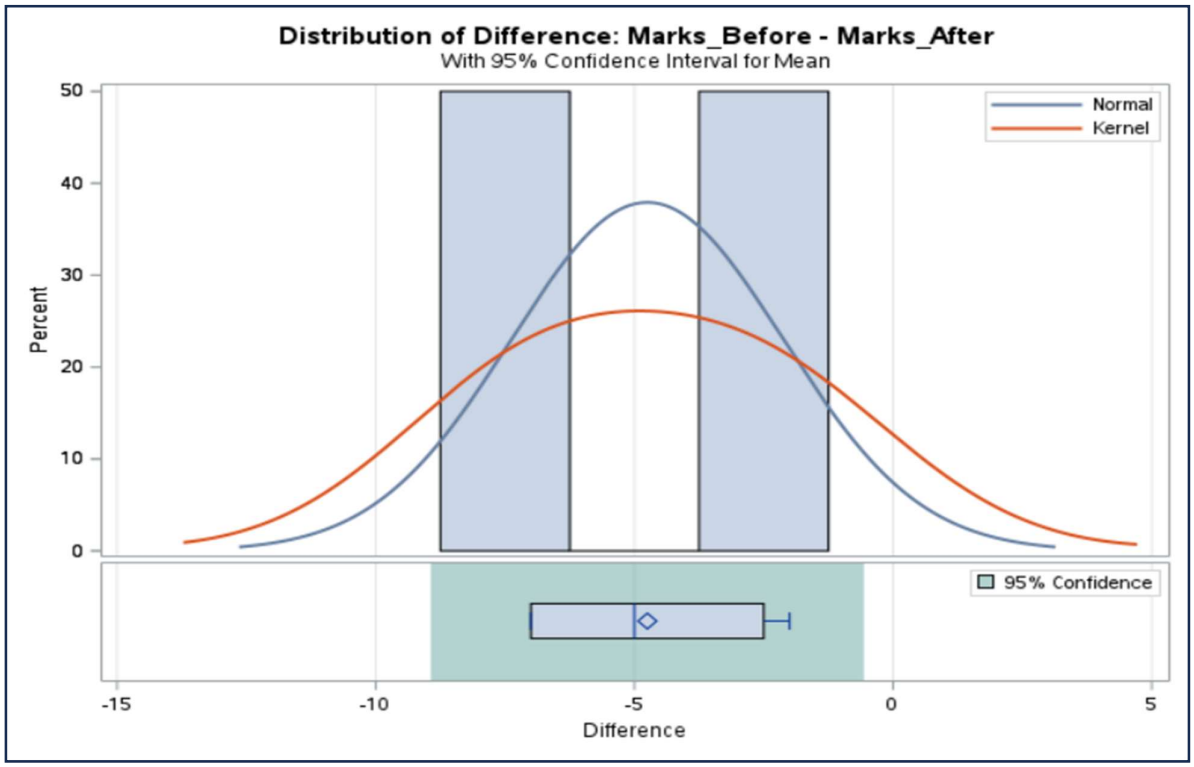
The TTEST Procedure

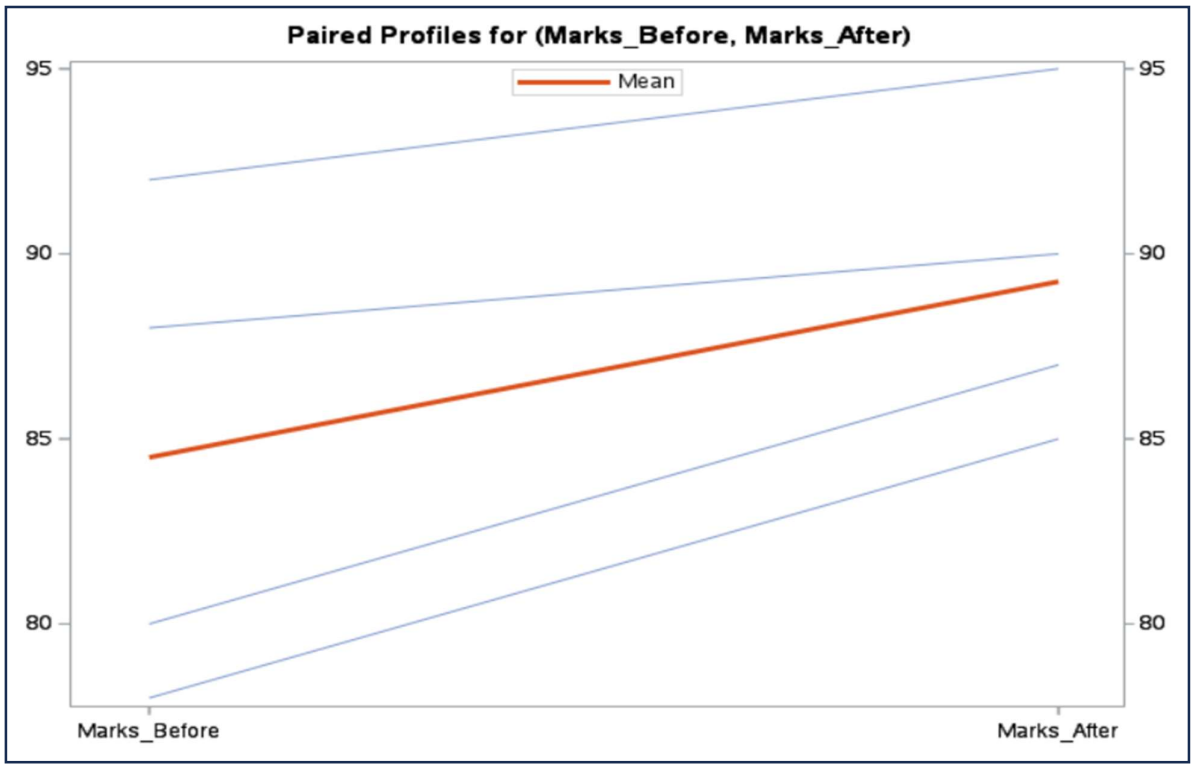
Difference: Marks_Before - Marks_After

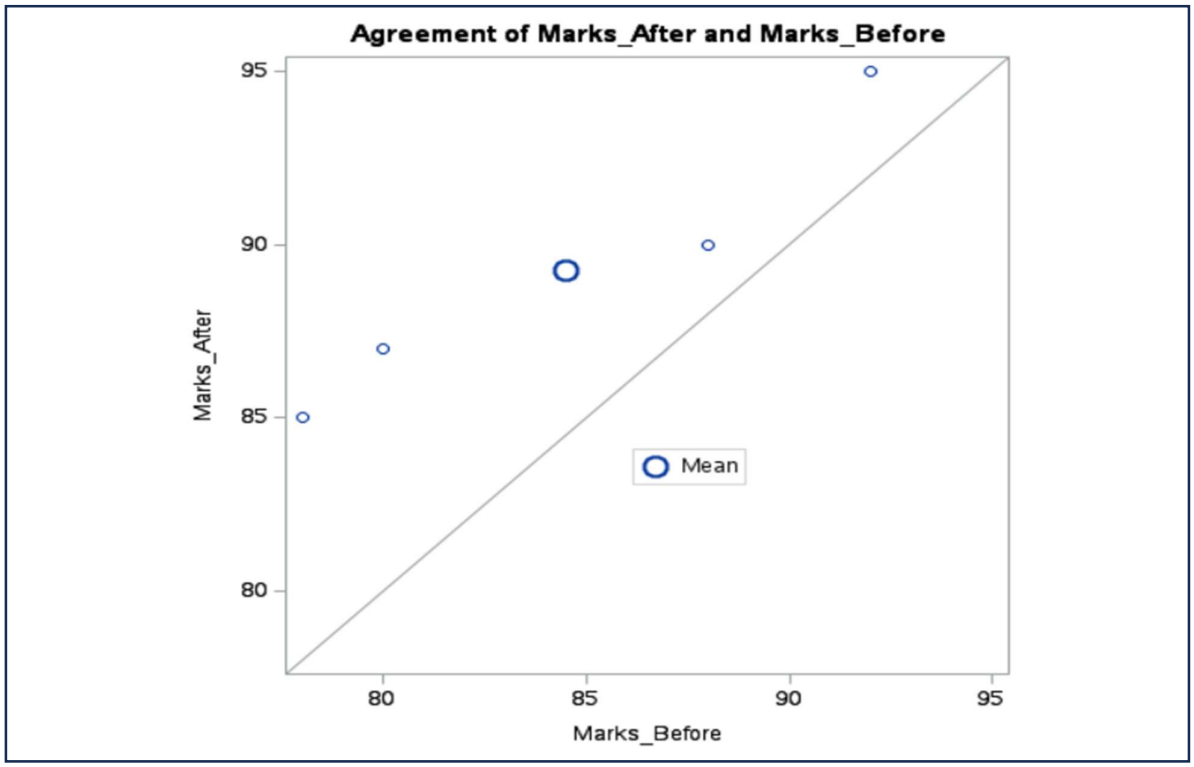
N	Mean	Std Dev	Std Err	Minimum	Maximum
4	-4.7500	2.6300	1.3150	-7.0000	-2.0000

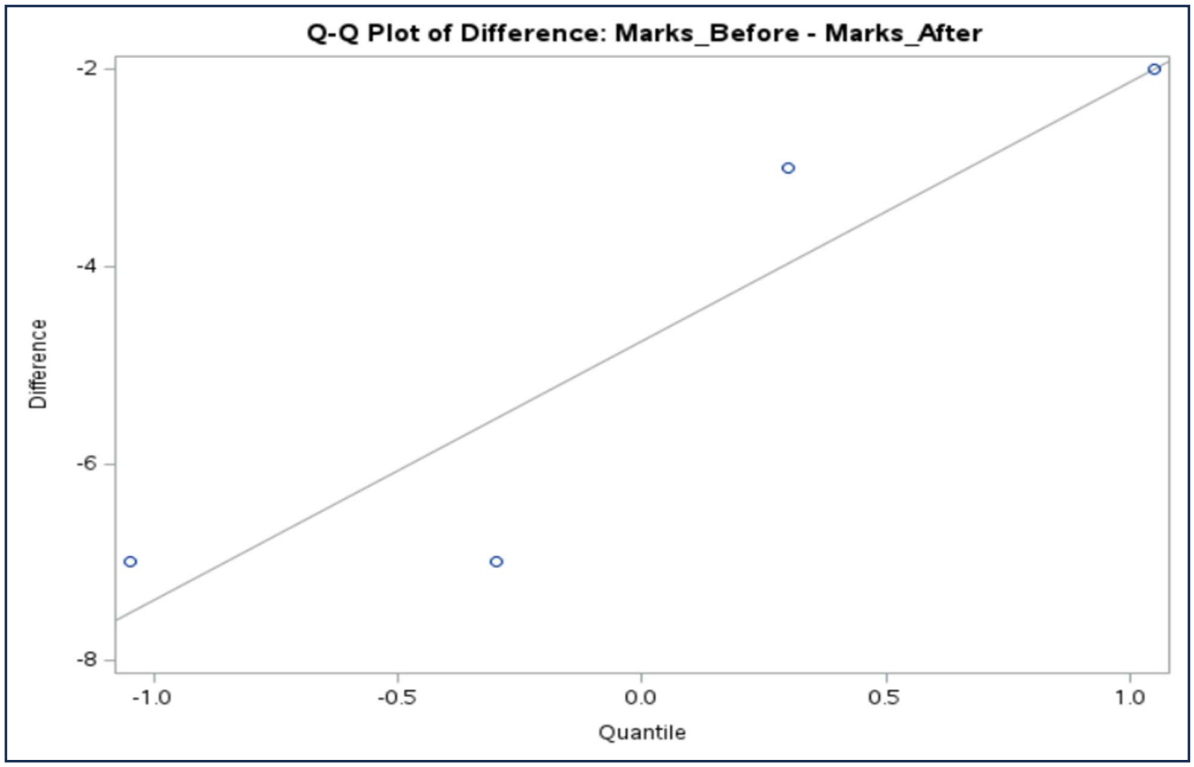
Mean	95% CL Mean		Std Dev	95% CL Std Dev	
-4.7500	-8.9348	-0.5652	2.6300	1.4898	9.8059

DF	t Value	Pr > |t|
3	-3.61	0.0364


Distribution of Difference: Marks_Before - Marks_After
With 95% Confidence Interval for Mean
Normal
Kernel
95% Confidence
Percent
50
40
30
20
10
0
-15
-10
-5
0
5
Difference


Paired Profiles for (Marks_Before, Marks_After)
Mean
95
90
85
80
Marks_Before
Marks_After


Agreement of Marks_After and Marks_Before
95
90
85
80
Marks_After
Mean
80
85
90
95
Marks_Before


Q-Q Plot of Difference: Marks_Before - Marks_After
-2
-4
-6
-8
Difference
-1.0
-0.5
0.0
0.5
1.0
Quantile

Practical No.21

Aim:- To compare the means of a numeric variable across more than two groups using one-way ANOVA in SAS.

Theory:- One-way ANOVA tests whether the means of a numeric variable are significantly different across multiple groups. It partitions the total variation into variation between groups and within groups, and calculates the F-statistic to test the null hypothesis that all group means are equal. PROC ANOVA in SAS provides F-value, p-value, and group means for comparison.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
data students;
    input Name $ Group $ Marks;
    datalines;
Amit A 85
Neha A 88
Yash B 90
Tushar B 92
Riya C 78
Sara C 80
;
run;

proc anova data=students;
    class Group;      /* Grouping variable */
    model Marks = Group;   /* Numeric variable ~ Group */
run;
```

Output:-

The ANOVA Procedure

Class Level Information		
Class	Levels	Values
Group	3	A B C

Number of Observations Read	6
Number of Observations Used	6

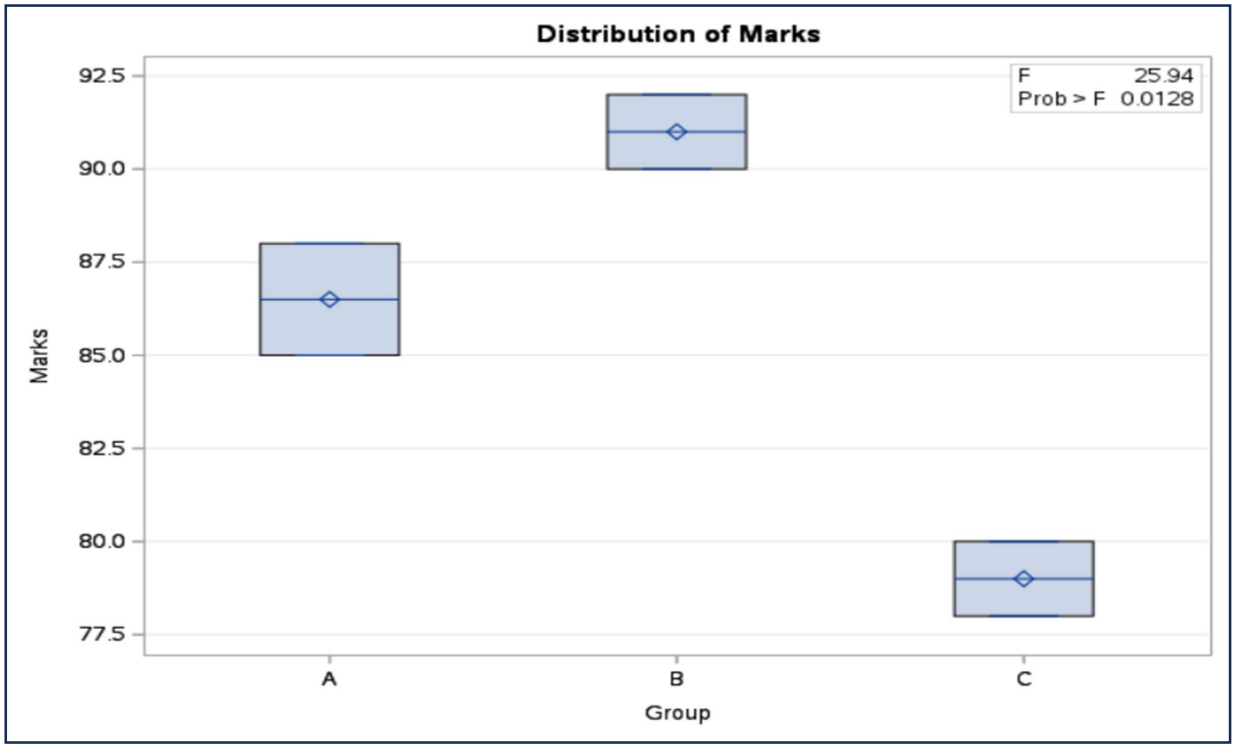
The ANOVA Procedure

Dependent Variable: Marks

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	147.0000000	73.5000000	25.94	0.0128
Error	3	8.5000000	2.8333333		
Corrected Total	5	155.5000000			

R-Square	Coeff Var	Root MSE	Marks Mean
0.945338	1.968714	1.683251	85.50000

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Group	2	147.0000000	73.5000000	25.94	0.0128

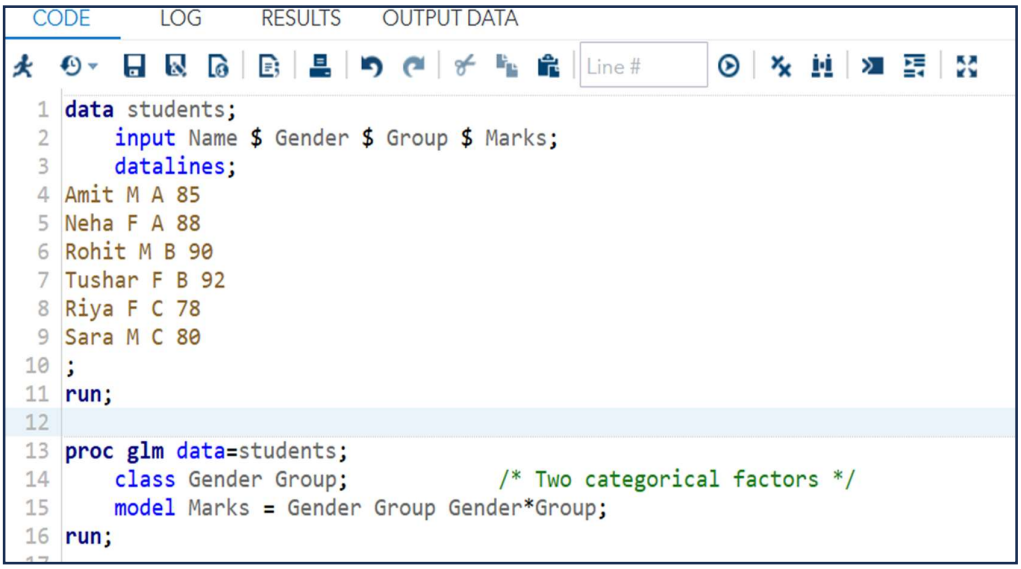

Distribution of Marks
F 25.94
Prob > F 0.0128
92.5
90.0
87.5
85.0
82.5
80.0
77.5
Marks
A
B
C
Group

Practical No.22

Aim:- To study the effect of two categorical factors on a numeric variable using two-way ANOVA in SAS.

Theory:- Two-way ANOVA examines the influence of two independent categorical variables on a numeric dependent variable and also tests whether there is an interaction effect between the factors. PROC GLM in SAS is used for two-way ANOVA and provides F-values, p-values, and means for each factor and their interaction.

Code:-



Output:-

The GLM Procedure

Class Level Information		
Class	Levels	Values
Gender	2	F M
Group	3	A B C

Number of Observations Read	6
Number of Observations Used	6

The GLM Procedure

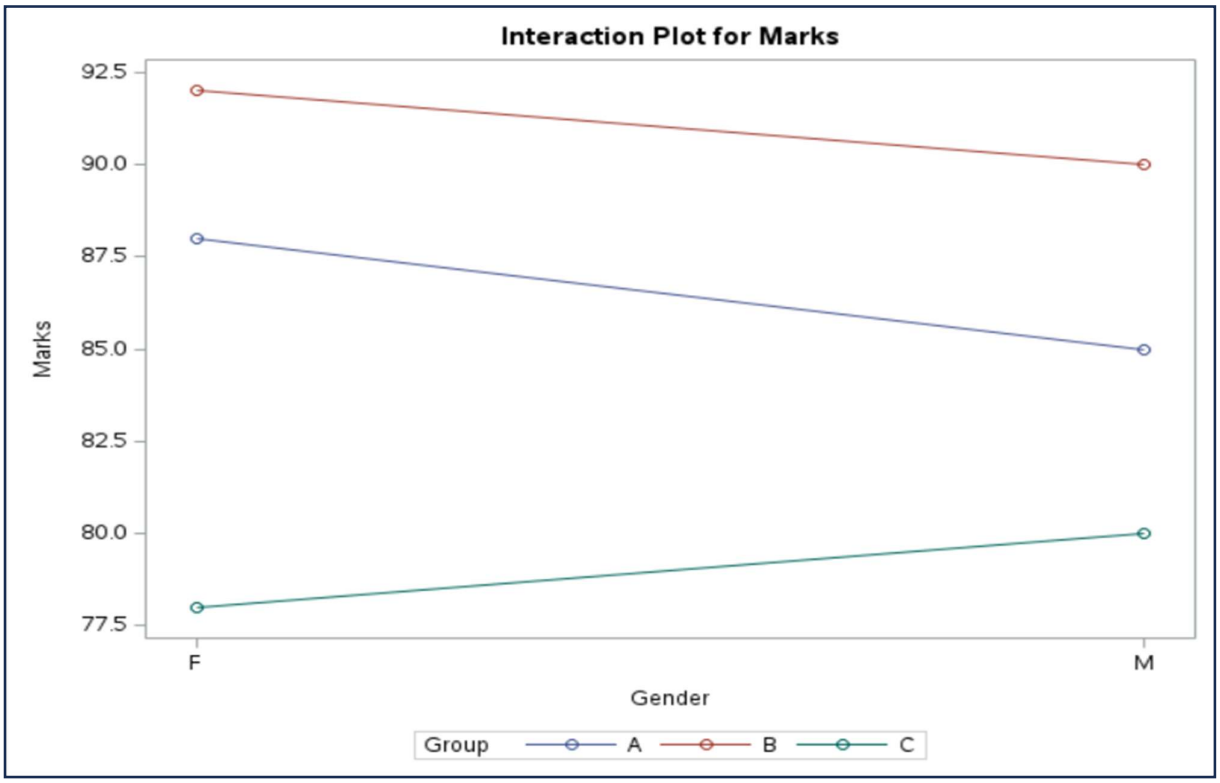
Dependent Variable: Marks

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	155.5000000	31.1000000	.	.
Error	0	0.0000000	.		
Corrected Total	5	155.5000000			

R-Square	Coeff Var	Root MSE	Marks Mean
1.000000	.	.	85.50000

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Gender	1	1.5000000	1.5000000	.	.
Group	2	147.0000000	73.5000000	.	.
Gender*Group	2	7.0000000	3.5000000	.	.

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Gender	1	1.5000000	1.5000000	.	.
Group	2	147.0000000	73.5000000	.	.
Gender*Group	2	7.0000000	3.5000000	.	.



Practical No.23

Aim:- To test whether there is an association between two categorical variables using the Chi-square test in SAS.

Theory:- The Chi-square test evaluates whether two categorical variables are independent or related. PROC FREQ with the TABLES statement and the CHISQ option computes the observed and expected frequencies, the Chi-square statistic, and the p-value. A p-value less than 0.05 indicates a significant association between the variables.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data survey;
2      input Gender $ Preference $ Count;
3      datalines;
4  M A 10
5  M B 15
6  F A 12
7  F B 18
8  ;
9  run;
10
11 proc freq data=survey;
12     tables Gender*Preference / chisq;
13     weight Count;  /* Use COUNT if frequency data is provided */
14 run;
```

Output:-

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Gender by Preference		
	Preference		
Gender	A	B	Total
F	12 21.82 40.00 54.55	18 32.73 60.00 54.55	30 54.55
M	10 18.18 40.00 45.45	15 27.27 60.00 45.45	25 45.45
Total	22 40.00	33 60.00	55 100.00

Statistics for Table of Gender by Preference

Statistic	DF	Value	Prob
Chi-Square	1	0.0000	1.0000
Likelihood Ratio Chi-Square	1	0.0000	1.0000
Continuity Adj. Chi-Square	1	0.0000	1.0000
Mantel-Haenszel Chi-Square	1	0.0000	1.0000
Phi Coefficient		0.0000	
Contingency Coefficient		0.0000	
Cramer's V		0.0000	

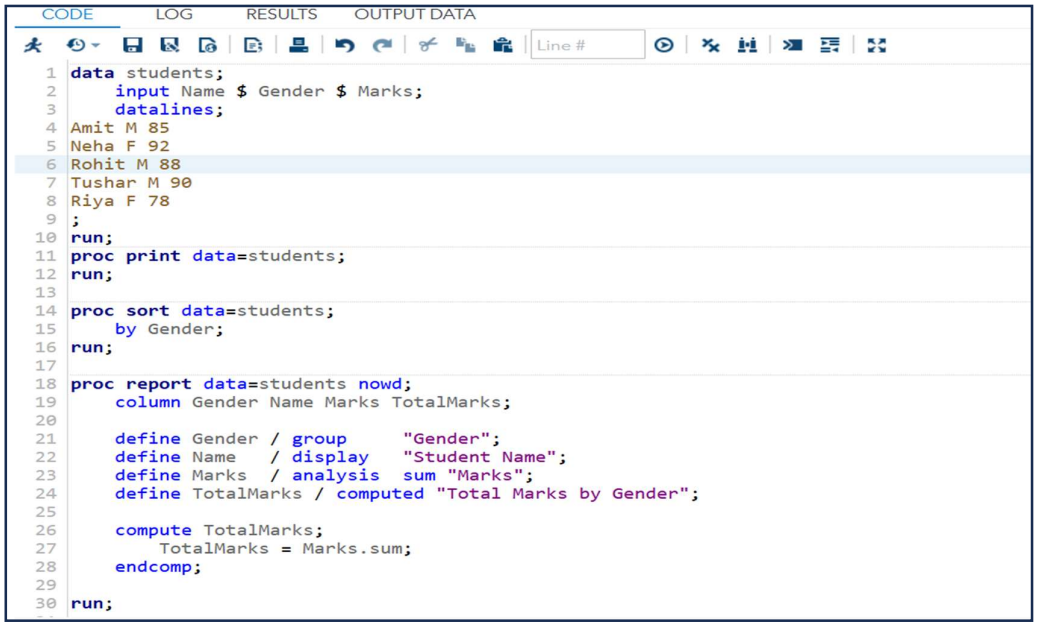
Fisher's Exact Test	
Cell (1,1) Frequency (F)	12
Left-sided Pr <= F	0.6081
Right-sided Pr >= F	0.6093
Table Probability (P)	0.2173
Two-sided Pr <= P	1.0000

Practical No.24

Aim:- To create a formatted tabular report of a dataset using PROC REPORT in SAS.

Theory:- PROC REPORT is used to generate customized tabular reports in SAS. It allows displaying variables in columns, grouping, computing summary statistics, and formatting output. This procedure is more flexible than PROC PRINT and is useful for generating professional-looking reports for analysis or presentation.

Code:-




```
data students;
    input Name $ Gender $ Marks;
    datalines;
Amit M 85
Neha F 92
Rohit M 88
Tushar M 90
Riya F 78
;
run;
proc print data=students;
run;

proc sort data=students;
    by Gender;
run;

proc report data=students nowd;
    column Gender Name Marks TotalMarks;

    define Gender / group     "Gender";
    define Name   / display   "Student Name";
    define Marks  / analysis  sum "Marks";
    define TotalMarks / computed "Total Marks by Gender";

    compute TotalMarks;
        TotalMarks = Marks.sum;
    endcomp;

run;
```

Output:-

Gender	Student Name	Marks	Total Marks by Gender
F	Neha	92	92
F	Riya	78	78
M	Amit	85	85
M	Rohit	88	88
M	Tushar	90	90

Practical No.25

Aim:- To generate a histogram and a box plot for visualizing the distribution and spread of numerical data using the PROC SGPLOT procedure in SAS.

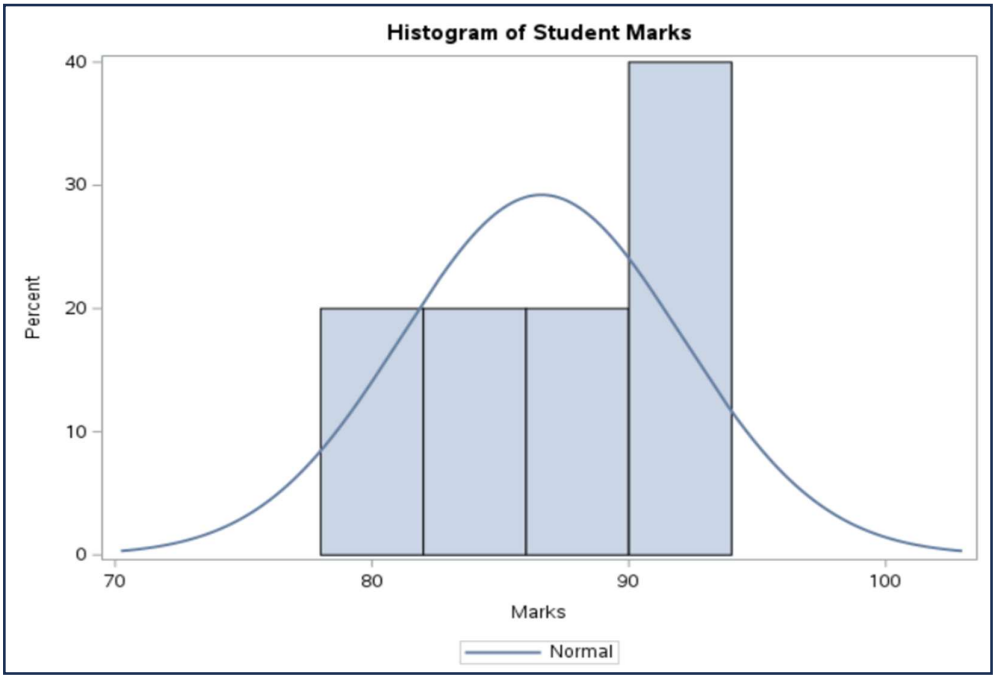
Theory:- PROC SGPLOT is a graphical procedure in SAS used to create high-quality statistical plots. A histogram helps visualize the distribution and frequency of a numeric variable, while a box plot shows the spread, median, and outliers of the data. These visual tools help in understanding patterns and variations within a dataset.

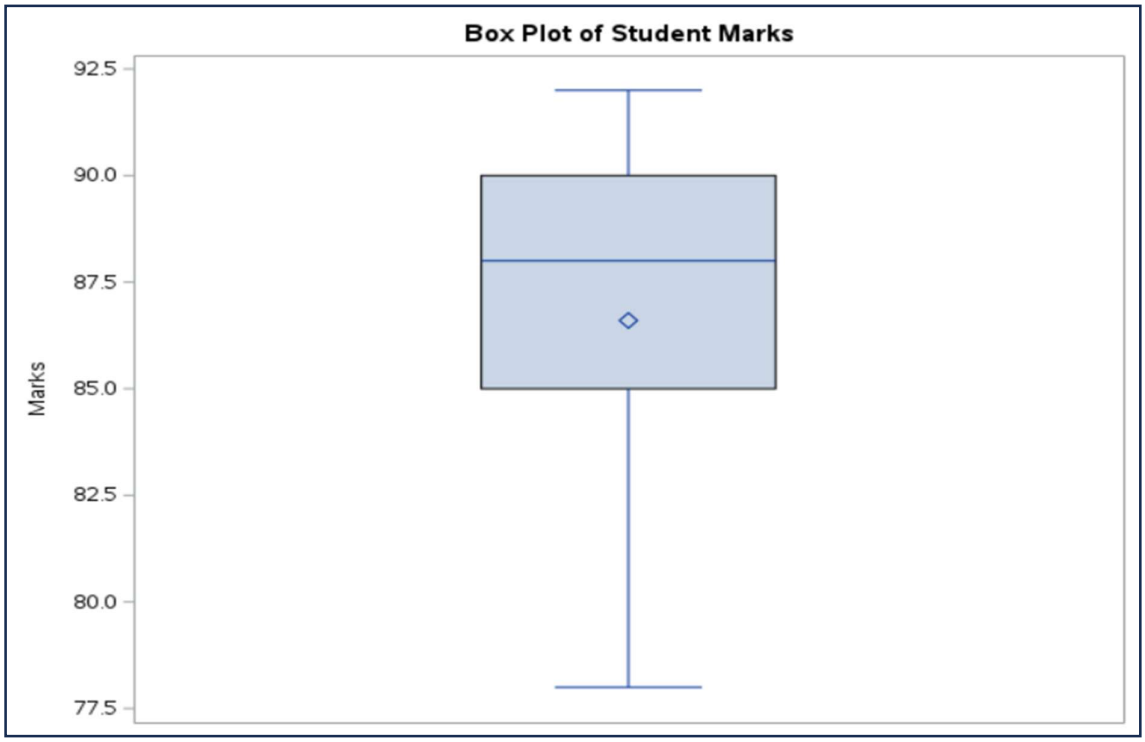
Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data marks;
2      input Name $ Marks;
3      datalines;
4  Amit 85
5  Neha 92
6  Rohit 88
7  Tushar 90
8  Riya 78
9  ;
10 run;
11
12 proc sgplot data=marks;
13     title "Histogram of Student Marks";
14     histogram Marks;
15     density Marks;
16 run;
17
18 proc sgplot data=marks;
19     title "Box Plot of Student Marks";
20     vbox Marks;
21 run;
```

Output:-




Box Plot of Student Marks
Marks
92.5
90.0
87.5
85.0
82.5
80.0
77.5

Practical No.26

Aim:- To compute correlation coefficients among numerical variables using PROC CORR in SAS and interpret the relationships between them.

Theory:- PROC CORR in SAS calculates correlation coefficients (such as Pearson's r) that measure the strength and direction of relationships between numerical variables. The procedure also produces a correlation matrix that displays pairwise correlations, sample size, and significance values. It is widely used in data analysis to determine linear association among variables.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data scores;
2      input Name $ Maths Science English;
3      datalines;
4  Amit 85 88 82
5  Neha 92 90 89
6  Rohit 78 80 75
7  Tushar 88 85 84
8  Riya 76 79 74
9  ;
10 run;
11
12 proc corr data=scores;
13     title "Correlation Matrix of Subject Scores";
14 run;
```

Output:-

Correlation Matrix of Subject Scores

The CORR Procedure

3 Variables:	Maths Science English

Simple Statistics

Variable	N	Mean	Std Dev	Sum	Minimum	Maximum
Maths	5	83.80000	6.72309	419.00000	76.00000	92.00000
Science	5	84.40000	4.82701	422.00000	79.00000	90.00000
English	5	80.80000	6.30079	404.00000	74.00000	89.00000

Pearson Correlation Coefficients, N = 5
Prob > |r| under H0: Rho=0

	Maths	Science	English
Maths	1.00000	0.92751 0.0232	0.99620 0.0003
Science	0.92751 0.0232	1.00000	0.94036 0.0173
English	0.99620 0.0003	0.94036 0.0173	1.00000

Practical No.27

Aim:- To perform simple linear regression analysis using PROC REG in SAS and evaluate how an independent variable predicts a dependent variable.

Theory:- Linear regression is a statistical method used to study the relationship between a dependent variable and one or more independent variables. PROC REG in SAS fits a linear model and provides estimates of regression coefficients, R-square, ANOVA table, and significance tests. It helps determine how strongly the predictor variable influences the outcome.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data study;
2      input Hours Marks;
3      datalines;
4  2 50
5  3 55
6  4 60
7  5 65
8  6 70
9  7 74
10 8 78
11 ;
12 run;
13
14 proc reg data=study;
15     model Marks = Hours;
16     title "Simple Linear Regression: Marks predicted by Hours of Study";
17 run;
18 quit;
```

Output:-

Simple Linear Regression: Marks predicted by Hours of Study

The REG Procedure
Model: MODEL1
Dependent Variable: Marks

Number of Observations Read	7
Number of Observations Used	7

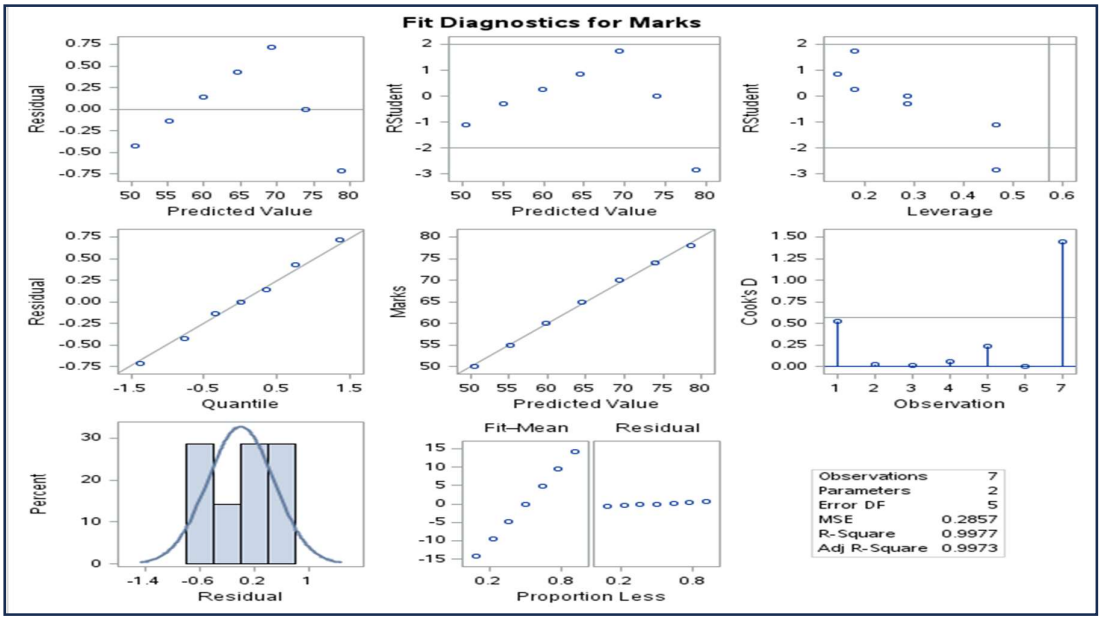
Analysis of Variance

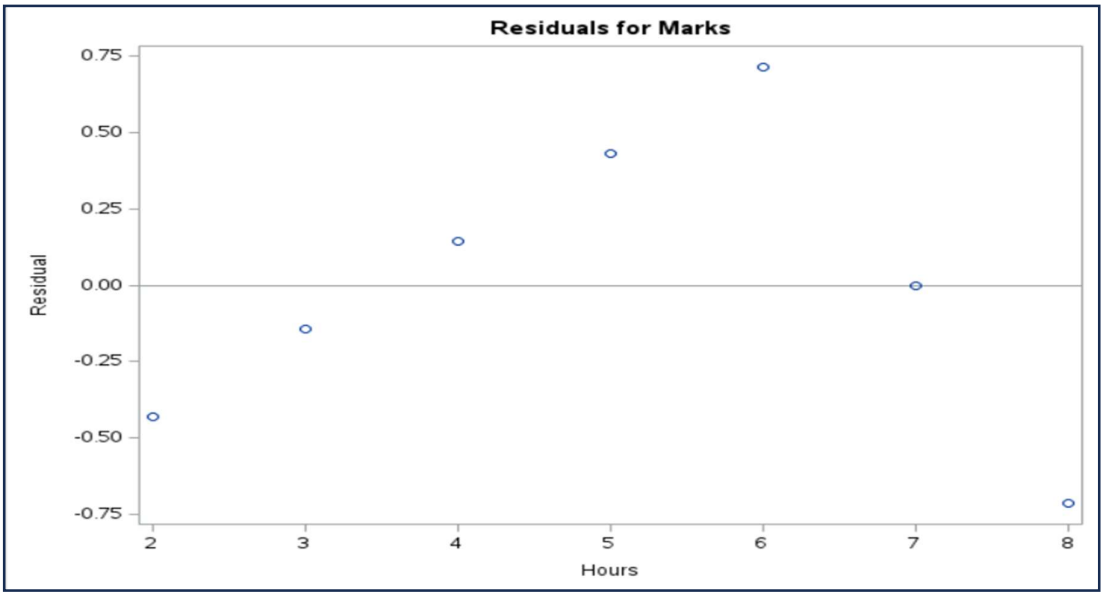
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	622.28571	622.28571	2178.00	<.0001
Error	5	1.42857	0.28571		
Corrected Total	6	623.71429			

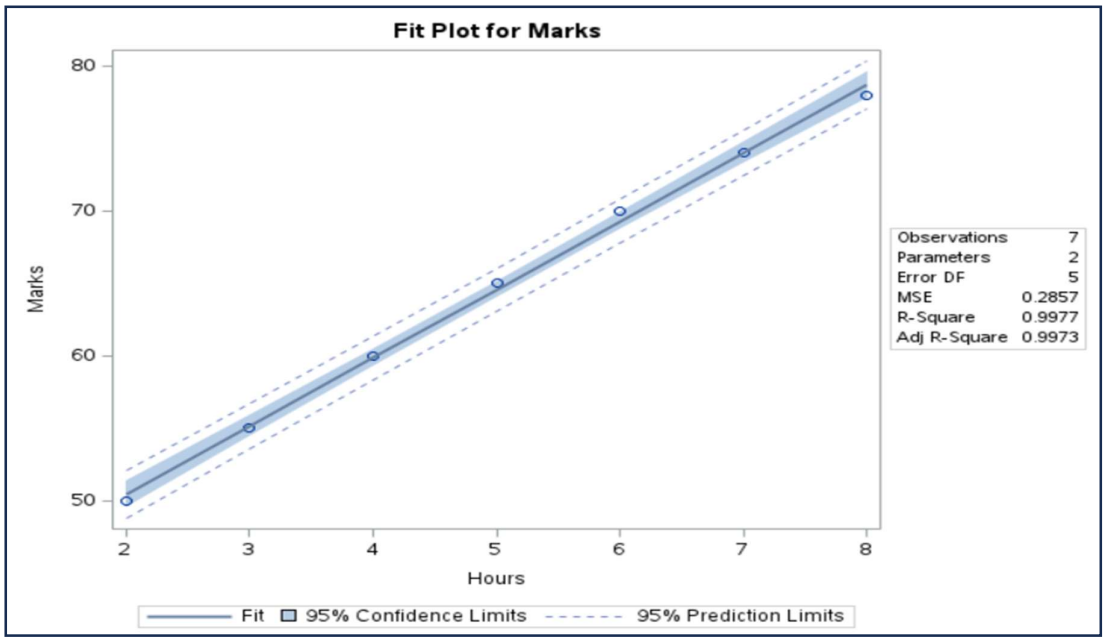
Root MSE	0.53452	R-Square	0.9977
Dependent Mean	64.57143	Adj R-Sq	0.9973
Coeff Var	0.82780		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	41.00000	0.54398	75.37	<.0001
Hours	1	4.71429	0.10102	46.67	<.0001


Fit Diagnostics for Marks
Residual
Predicted Value
RStudent
Predicted Value
RStudent
Leverage
Residual
Quantile
Marks
Predicted Value
Cook's D
Observation
Percent
Residual
Fit–Mean
Residual
Proportion Less
Observations 7
Parameters 2
Error DF 5
MSE 0.2857
R-Square 0.9977
Adj R-Square 0.9973


Residuals for Marks
Residual
Hours


Fit Plot for Marks
Marks
Hours
Observations 7
Parameters 2
Error DF 5
MSE 0.2857
R-Square 0.9977
Adj R-Square 0.9973
Fit
95% Confidence Limits
95% Prediction Limits

Practical No.28

Aim:- To perform logistic regression using PROC LOGISTIC in SAS to model the relationship between a binary outcome variable and one or more predictor variables.

Theory:- Logistic regression is used when the dependent variable is binary (0/1). PROC LOGISTIC in SAS estimates the probability of an event occurring using the logistic function. It provides odds ratios, parameter estimates, and model significance tests to evaluate how predictors affect the likelihood of the event.

Code:-

CODE LOG RESULTS OUTPUT DATA

```
1  data exam;
2      input Hours Pass;
3      datalines;
4  2 0
5  3 0
6  4 0
7  5 1
8  6 1
9  7 1
10 8 1
11 ;
12 run;
13
14 proc logistic data=exam;
15     model Pass(event='1') = Hours;
16     title "Logistic Regression: Probability of Passing Based on Hours Studied";
17 run;
18 quit;
```

Output:-

Logistic Regression: Probability of Passing Based on Hours Studied

The LOGISTIC Procedure

Model Information	
Data Set	WORK.EXAM
Response Variable	Pass
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	7
Number of Observations Used	7

Response Profile		
Ordered Value	Pass	Total Frequency
1	0	3
2	1	4

Probability modeled is Pass=1.

Model Convergence Status
Complete separation of data points detected.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	11.561	4.011
SC	11.507	3.903
-2 Log L	9.561	0.011

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	9.5499	1	0.0020
Score	5.2500	1	0.0219
Wald	0.1875	1	0.6650

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-53.2863	124.2	0.1840	0.6680
Hours	1	11.8329	27.3257	0.1875	0.6650

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
Hours	>999.999	<0.001	>999.999

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	100.0	Somers' D	1.000
Percent Discordant	0.0	Gamma	1.000
Percent Tied	0.0	Tau-a	0.571
Pairs	12	c	1.000

Practical No.29

Aim:- To export SAS procedure results into a Microsoft Excel file using the Output Delivery System (ODS EXCEL).

Theory:- ODS EXCEL allows SAS users to create structured and formatted Excel files directly from procedure outputs. It supports multiple sheets, styling, and automatic table formatting. This makes it easy to store, share, and analyze SAS results using Microsoft Excel.

Code:-

CODE LOG RESULTS

```
1  ods excel file="/home/u64406284/output.xlsx"
2            options(sheet_name="Summary");
3
4  proc means data=sashelp.class;
5  run;
6
7  ods excel close;
```

Output:-

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.3157895	1.4926722	11.0000000	16.0000000
Height	19	62.3368421	5.1270752	51.3000000	72.0000000
Weight	19	100.0263158	22.7739335	50.5000000	150.0000000