ASSIGNMENTS

Question 1.1: Write the Answer to these questions. Note: Give at least one example for each of the questions.

1) What is the difference between static and dynamic variables in Python?

Static Variables:

- Defined at the class level.
- Shared across all instances of the class.
- Example:

```
class MyClass:
    static_var = 10  # static variable

print(MyClass.static_var)  # Output: 10
```

Dynamic Variables:

- Defined at the instance level.
- Unique to each instance.
- Example:

```
class MyClass:
    def __init__(self, value):
        self.dynamic_var = value  # dynamic variable

obj1 = MyClass(5)
obj2 = MyClass(10)
print(obj1.dynamic_var)  # Output: 5
print(obj2.dynamic_var)  # Output: 10
```

2) Explain the purpose of "pop", "popitem", "clear()" in a dictionary with suitable examples.

- pop: Removes and returns the value for a specified key.

```
my_dict = {'a': 1, 'b': 2}
value = my_dict.pop('a')
print(value)  # Output: 1
print(my_dict)  # Output: {'b': 2}
```

- popitem: Removes and returns the last key-value pair.

```
my_dict = {'a': 1, 'b': 2}
item = my_dict.popitem()
print(item)  # Output: ('b', 2)
print(my_dict)  # Output: {'a': 1}
```

- clear: Removes all items from the dictionary.

```
my_dict = {'a': 1, 'b': 2}
my_dict.clear()
print(my_dict)  # Output: {}
```

3) What do you mean by FrozenSet? Explain it with suitable examples.

- A frozenset is an immutable version of a set.
- Example:

```
my_set = frozenset([1, 2, 3])
print(my_set)  # Output: frozenset({1, 2, 3})
# my_set.add(4)  # This will raise an error
```

4) Differentiate between mutable and immutable data types in Python and give examples of mutable and immutable data types.

Mutable Data Types:

- Can be changed after creation.
- Examples: list, dictionary, set.

```
my_list = [1, 2, 3]
my_list.append(4)  # my_list is now [1, 2, 3, 4]
```

Immutable Data Types:

- Cannot be changed after creation.
- Examples: string, tuple, frozenset.

```
my_tuple = (1, 2, 3)
# my_tuple[0] = 4  # This will raise an error
```

5) What is init? Explain with an example.

init():

- A special method in Python classes, called a constructor, which initializes a new instance of the class.

Example:

```
class MyClass:
    def __init__(self, name):
        self.name = name

obj = MyClass("John")
print(obj.name)  # Output: John
```

6) What is docstring in Python? Explain with an example.

Docstring:

- A string literal that appears as the first statement in a module, function, class, or method definition, used to document the object.

Example:

```
def my_function():
    """This is a docstring."""
    return

print(my_function.__doc__)  # Output: This is a docstring.
```

7) What are unit tests in Python?

Unit Tests:

- Tests that check the functionality of a specific section of code, usually a function or a method, ensuring it works as intended.

Example:

```
import unittest

def add(a, b):
    return a + b

class TestAddFunction(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(2, 3), 5)

if __name__ == '__main__':
    unittest.main()
```

8) What is break, continue and pass in Python?

break:

- Terminates the loop prematurely

```
for i in range(5):
    if i == 3:
        break
    print(i)
# Output: 0 1 2
```

continue:

- Skips the current iteration and continues with the next

```
for i in range(5):
    if i == 3:
        continue
    print(i)
# Output: 0 1 2 4
```

pass:

- A placeholder that does nothing; used to fill in a block of code where something is syntactically required.

```
for i in range(5):
    if i == 3:
        pass
    print(i)
# Output: 0 1 2 3 4
```

9) What is the use of self in Python?

self:

- Represents the instance of the class.
- Allows access to the attributes and methods of the class in Python.

```python
class MyClass:
    def __init__(self, name):
        self.name = name

    def greet(self):
        return f"Hello, {self.name}"

obj = MyClass("Alice")
print(obj.greet())  # Output: Hello, Alice
```

10) What are global, protected and private attributes in Python?

Global Attributes:

- Accessible from anywhere in the program.
- Example:

```python
global_var = 10

def func():
    global global_var
    global_var = 20

func()
print(global_var)  # Output: 20
```

Protected Attributes:

- Indicated by a single underscore (_).
- Should not be accessed outside the class or its subclasses.
- Example:

```python
class MyClass:
    def __init__(self):
        self._protected_var = 10

obj = MyClass()
print(obj._protected_var)  # Output: 10
```

Private Attributes:

- Indicated by a double underscore (__).
- Cannot be accessed directly outside the class.
- Example:

```python
class MyClass:
    def __init__(self):
        self.__private_var = 10

obj = MyClass()
print(obj.__private_var)  # AttributeError
```

11) What are modules and packages in Python?

Modules:

- Single files (or files) that contain Python code, such as functions or classes.
- Example:

```
# mymodule.py
def my_function():
    return "Hello from a module"
```

Packages:

- A way of structuring Python's module namespace by using "dotted module names".
- Consists of a **init**.py file to be considered a package.
- Example:

```
mypackage/
    __init__.py
    module1.py
    module2.py
```

12) What are lists and tuples? What is the key difference between the two?

Lists:

- Mutable, ordered collections of items.

```
my_list = [1, 2, 3]
my_list[0] = 4
print(my_list)  # Output: [4, 2, 3]
```

Tuples:

- Immutable, ordered collections of items.

```
my_tuple = (1, 2, 3)
my_tuple[0] = 4  # This will raise an error
```

13) What is an Interpreted language & dynamically typed language? Write 5 differences between them.

Interpreted Language:

- Code is executed line by line by an interpreter.
- Example: Python, JavaScript

Dynamically Typed Language:

- Variables are bound to types only at runtime.
- Example: Python, Ruby

Differences:

1. **Execution**: Interpreted languages execute code line-by-line; dynamically typed languages determine variable types at runtime.
2. **Speed**: Interpreted languages are generally slower than compiled languages; dynamically typed languages can have runtime overhead.
3. **Error Checking**: Interpreted languages check for errors at runtime; dynamically typed languages may have fewer compile-time errors.
4. **Type Binding**: In interpreted languages, type binding happens at runtime; in dynamically typed languages, variable types are not known until runtime.
5. **Flexibility**: Interpreted languages offer flexibility in code execution; dynamically typed languages offer flexibility in variable usage.

14) What are Dict and list comprehensions?

Dict Comprehensions:

- A concise way to create dictionaries

```
my_dict = {x: x*x for x in range(5)}
print(my_dict)  # Output: {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```

List Comprehensions:

- A concise way to create lists.

```
my_list = [x*x for x in range(5)]
print(my_list)  # Output: [0, 1, 4, 9, 16]
```

15) What are decorators in Python? Explain it with an example. Write down its use cases.

Decorators:

- Functions that modify the functionality of other functions.
- Example:

```
def decorator_function(func):
    def wrapper():
        print("Something is happening before the function is called.")
        func()
        print("Something is happening after the function is called.")
    return wrapper

@decorator_function
def say_hello():
    print("Hello!")

say_hello()
```

Use Cases:

- Logging
- Access control and authentication
- Instrumentation and timing functions
- Cache results of expensive computations

16) How is memory managed in Python?

- Memory management in Python is handled by the Python Memory Manager.
- Involves private heap space.
- Includes built-in garbage collection, which recycles unused memory automatically.

17) What is lambda in Python? Why is it used?

lambda:

- A small anonymous function defined using the lambda keyword.
- Used for creating small, one-time, and inline function objects.

Example:

```python
add = lambda x, y: x + y
print(add(5, 3))  # Output: 8
```

18) Explain split() and join() functions in Python?

split():

- Splits a string into a list where each word is a list item.

```python
my_string = "Hello world"
print(my_string.split())  # Output: ['Hello', 'world']
```

join():

- Joins elements of an iterable into a single string

```python
my_list = ['Hello', 'world']
print(" ".join(my_list))  # Output: Hello world
```

19) What are iterators, iterable & generators in Python?

Iterable:

- An object capable of returning its members one at a time.
- Example: list, tuple, dict

Iterator:

- An object representing a stream of data.
- Implemented using __iter__() and __next__() methods.

Generator:

- A function that returns an iterator using yield.
- Example

```python
def my_generator():
    yield 1
    yield 2
    yield 3

for value in my_generator():
    print(value)
```

20) What is the difference between xrange and range in Python?

range:

- Generates a list of numbers in Python 2 and an iterator in Python 3.

xrange:

- Generates numbers on demand (lazy evaluation).
- Only available in Python 2.

21) Pillars of OOPS ?

- **Encapsulation:** Wrapping data and methods into a single unit (class).
- **Abstraction:** Hiding implementation details and showing only the functionality.
- **Inheritance:** Acquiring properties of one class into another.
- **Polymorphism:** Ability to take many forms; same function name but different signatures.

22) How will you check if a class is a child of another class?

Example:

```python
class Parent:
    pass

class Child(Parent):
    pass

print(issubclass(Child, Parent))  # Output: True
```

23) How does inheritance work in Python? Explain all types of inheritance with an example.

Single Inheritance:

- Inherits from one base class

```python
class Parent:
    pass

class Child(Parent):
    pass
```

Multiple Inheritance:

- Inherits from multiple base classes

```python
class Parent1:
    pass

class Parent2:
    pass

class Child(Parent1, Parent2):
    pass
```

Multilevel Inheritance:

- A class is derived from a class which is also derived from another class.

```python
class GrandParent:
    pass

class Parent(GrandParent):
    pass

class Child(Parent):
    pass
```

Hierarchical Inheritance:

- Multiple classes inherit from one base class

```python
class Parent:
    pass

class Child1(Parent):
    pass

class Child2(Parent):
    pass
```

Hybrid Inheritance:

- A combination of multiple types of inheritance.

```
class Parent1:
    pass

class Parent2:
    pass

class Child1(Parent1):
    pass

class Child2(Parent1, Parent2):
    pass
```

24) What is encapsulation? Explain it with an example.

Encapsulation:

- Wrapping data and methods into a single unit.
- Example

```
class MyClass:
    def __init__(self, value):
        self.__private_value = value

    def get_value(self):
        return self.__private_value

obj = MyClass(10)
print(obj.get_value())  # Output: 10
```

25) What is polymorphism? Explain it with an example.

Polymorphism:

- The ability of different classes to respond to the same function call.

Example:

```
class Dog:
    def sound(self):
        return "Bark"

class Cat:
    def sound(self):
        return "Meow"

def make_sound(animal):
    print(animal.sound())

dog = Dog()
cat = Cat()

make_sound(dog)  # Output: Bark
make_sound(cat)  # Output: Meow
```

Question 1. 2. Which of the following identifier names are invalid and why?

a) Serial_no.
b) lst_Room
c) Hundred$
d) Total_Marks
e) total-Marks
f) Total Marks
g) True
h) _Percentag

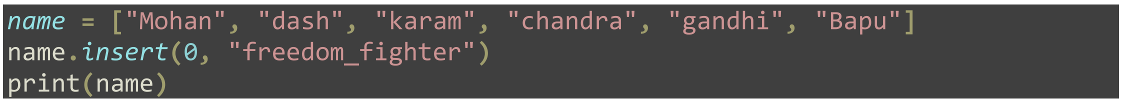
Invalid identifiers:

- **Serial_no.** (period may make it invalid in some contexts)
- **Hundred$** (dollar sign is not allowed)
- **total-Marks** (hyphen is not allowed)
- **Total Marks** (space is not allowed)
- **True** (reserved keyword)

Question 1.3.

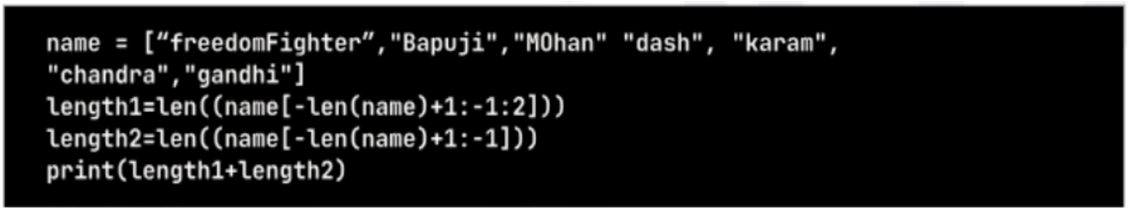
name= ["Mohan",”dash","karam”, "chandra","gandhi","Bapu"] do the following operations in this list;

a) add an element "freedom_fighter" in this list at the 0th index.



```
name = ["Mohan", "dash", "karam", "chandra", "gandhi", "Bapu"]
name.insert(0, "freedom_fighter")
print(name)
```

b) find the output of the following ,and explain how?



```
name = [“freedomFighter”,"Bapuji","MOhan" "dash", "karam",
"chandra","gandhi"]
length1=len((name[-len(name)+1:-1:2]))
length2=len((name[-len(name)+1:-1]))
print(length1+length2)
```

length1 = len(name[-6:-1:2]) = len([“Babuji”,”dash”,”chandra”]) = 3
length2 = len(name[-6:-1]) = len(["Bapuji", "MOhan", "dash", "karam", "chandra"]) = 5

print(length1 +length2) = 5+3 =   8 **Ans.**

c) add two more elements in the name ["NetaJi","Bose"] at the end of the list

```
name.extend(["Netaji","Bose"])
print(name)
```

d) what will be the value of temp:

```
name = ["Bapuji", "dash", "karam","chandra","gandi","Mohan"]
temp=name[-1]
name[-1]=name[0]
name[0]=temp
print(name)
```

temp = “Mohan”

Question 1.4. Find the output of the following.

```
animal = ['Human','cat','mat','cat','rat','Human', 'lion']
print(animal.count('Human'))
print(animal.index('rot'))
print(len(animal))
```

```
2
4
7
```

Question 1.5.

tuple1 = (10, 20, "Apple", 3.4, 'a', ["master", "ji"], ("sita", "geeta", 22), [{"roll_no": 3}], {"name": "Navneet"})

Perform the following operations:

a) print(len(tuple1))

b) print(tuple1[-1]["name"])

c) Fetch the value of roll_no from this tuple.

d) print(tuple1[-3][1])

e) Fetch the element 22 from this tuple.

Ans:

a) 9

b) Navneet

c) roll_no_value = tuple1[-2][0]["roll_no"]
print(roll_no_value)

output: 3

d) geeta

e) element_22 = tuple1[6][2]
print(element_22)

Question 1.6 - Question 19 -----------Coding Questions

Question 20. What do you mean by Measure of Central Tendency and Measures of Dispersion. How it can be calculated.

Definition: Measures of central tendency are statistical metrics used to identify the center point or typical value of a dataset. These measures summarize a dataset with a single value that represents the middle or center of its distribution.

Common Measures:

1. **Mean (Arithmetic Average):**
   1. The sum of all the values in the dataset divided by the number of values.
   2. Formula Mean = $\frac{\sum X}{N}$, where X is each value and N is the number of values.

2. **Median:**
   1. The middle value of a dataset when it is ordered from least to greatest. If the dataset has an even number of values, the median is the average of the two middle numbers.
   2. Calculation: Arrange the data in ascending order and find the middle value.
3. **Mode:**
   1. The value that appears most frequently in a dataset.
   2. Calculation: Identify the value that occurs most often in the data.

Example Calculation: Consider the dataset: [4, 8, 6, 5, 3, 8, 9]

- **Mean:** (4+8+6+5+3+8+9)/7=43/7≈6.14
- **Median:** Ordered dataset: [3, 4, 5, 6, 8, 8, 9] → Median = 6
- **Mode:** 8 (appears most frequently)

Measures of Dispersion

Definition: Measures of dispersion describe the spread or variability within a dataset. These measures help to understand the degree to which the data points differ from the central tendency.

Common Measures:

1. **Range:**
    1. The difference between the highest and lowest values in the dataset.
    2. Formula: Range=Maximum Value−Minimum Value
2. **Variance:**
    1. The average of the squared differences from the mean.
    2. Variance (σ^2) = $\frac{\sum(X-Mean)^2}{N}$
3. **Standard Deviation:**
    1. The square root of the variance, representing the average amount by which each data point differs from the mean.
    2. Standard Deviation = $\sqrt{variance}$
4. **Interquartile Range (IQR):**
    1. The range between the first quartile (Q1) and the third quartile (Q3), which represents the middle 50% of the data.
    2. Formula: IQR=Q3−Q1

Example Calculation: Consider the dataset: [4, 8, 6, 5, 3, 8, 9]

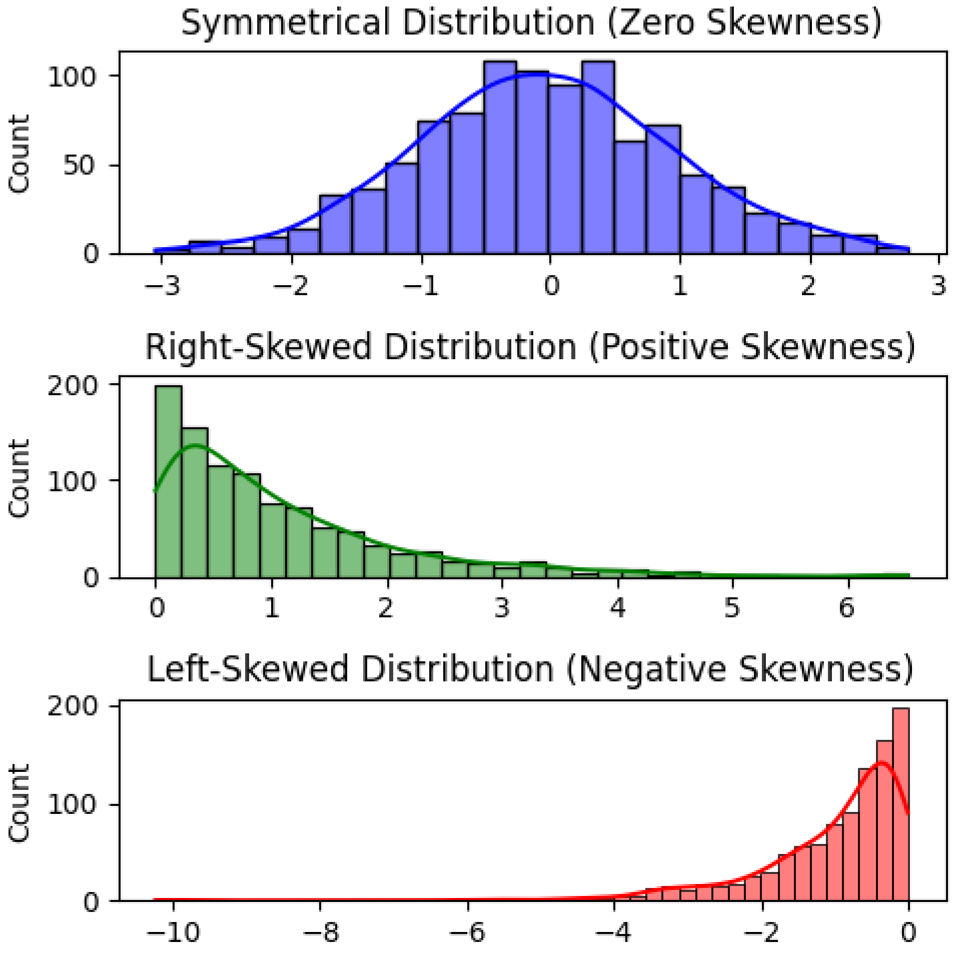
- **Range:** 9−3=69 - 3 = 69−3=6
- **Variance:**
    - Mean: 6.146.146.14
    - Differences from the mean: [−2.14,1.86, −0.14, −1.14, −3.14,1.86,2.86]
    - Squared differences: [4.5796,3.4596,0.0196,1.2996,9.8596,3.4596,8.1796]
    - Variance: $\frac{4.5796+3.4596+0.0196+1.2996+9.8596+3.4596+8.17967}{7}$ ≈4.98
- **Standard Deviation:** $\sqrt{4.98}$ ≈2.23
- **Interquartile Range (IQR):**
    - Ordered dataset: [3, 4, 5, 6, 8, 8, 9]
    - Q1 (25th percentile): 4
    - Q3 (75th percentile): 8
    - IQR: 8−4=48 - 4 = 48−4=4

Question 21. What do you mean by skewness. Explain its types. Use graph to show.

Definition: Skewness is a statistical measure that describes the asymmetry of the distribution of values in a dataset. It indicates whether the data points are concentrated more on one side of the mean than the other. Skewness can help identify the direction and extent of deviation from a normal distribution (which has a skewness of 0).

Types of Skewness:

1. **Positive Skewness (Right-Skewed):**
    - The right tail (larger values) is longer or fatter than the left tail.
    - Most data points are concentrated on the left side, and the mean is typically greater than the median.
    - Example: Income distribution, where a few high incomes create a long right tail.
2. **Negative Skewness (Left-Skewed):**
    - The left tail (smaller values) is longer or fatter than the right tail.
    - Most data points are concentrated on the right side, and the mean is typically less than the median.
    - Example: Age at retirement, where most people retire around a certain age, but a few retire much earlier.
3. **Zero Skewness (Symmetrical):**
    - The left and right tails are approximately mirror images of each other.
    - The mean and median are close or equal.
    - Example: Height distribution in a large population.


Symmetrical Distribution (Zero Skewness)

Right-Skewed Distribution (Positive Skewness)

Left-Skewed Distribution (Negative Skewness)

Question 22. Explain PROBABILITY MASSFUNCTION (PMF) and PROBABILITY DENSITY FUNCTION (PDF) and what is the difference between them?

Probability Mass Function (PMF)

- **Definition**: The PMF is used for discrete random variables. It gives the probability that a discrete random variable is exactly equal to some value.
- **Notation**: If X is a discrete random variable, then the PMF is often denoted as P(X=x)
- **Properties**:
    - P(X=x) ≥0 for all x
    - The sum of P(X=x) over all possible values of x is 1, i.e., ∑ P(X=x) = 1
- **Example**: The roll of a fair six-sided die. The PMF would assign a probability of 1/6 to each outcome (1 through 6).

Probability Density Function (PDF)

- **Definition**: The PDF is used for continuous random variables. It describes the relative likelihood of the random variable taking on a specific value, but it does not give probabilities directly.
- **Notation**: If X is a continuous random variable, then the PDF is often denoted as f(x)
- **Properties**:
    - f(x)≥ 0 for all x
    - The total area under the PDF curve is 1, i.e., $\int_{-\infty}^{\infty} f(x)dx = 1$
- The probability that X falls within a certain interval [a, b] is given by the integral of the PDF over that interval: $\int_{a}^{b} f(x)dx$
- **Example**: The height of adult women in a country. The PDF would describe how height values are distributed, and the probability of finding a woman in a specific height range is given by the area under the curve within that range.

Question 23. Whatis correlation. Explainits type in details. What are the methodsof determiningcorrelation

Correlation measures the relationship between two variables, indicating how one variable change when the other variable changes. It quantifies the degree to which two variables are linearly related.

Types of Correlation

1. **Positive Correlation**:
    - **Definition**: When one variable increases, the other variable also increases, and when one variable decreases, the other variable also decreases.
    - **Example**: Height and weight generally have a positive correlation because taller people tend to weigh more.
2. **Negative Correlation**:
    - **Definition**: When one variable increases, the other variable decreases, and vice versa.
    - **Example**: The number of hours spent watching TV and academic performance might have a negative correlation because more TV watching might lead to lower grades.
3. **No Correlation**:
    - **Definition**: There is no predictable relationship between the variables. Changes in one variable do not correspond to changes in the other variable.

- **Example**: The number of books read in a year and the number of pizzas eaten might have no correlation.

Methods of Determining Correlation

1. **Pearson Correlation Coefficient (r)**:
    1. **Definition**: Measures the strength and direction of the linear relationship between two continuous variables.
    2. **Range**: -1 to +1
        1. r=+1: Perfect positive linear relationship.
        2. r=−1: Perfect negative linear relationship.
        3. r=0: No linear relationship.
    3. **Formula**:

    $$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

    Usage: Widely used for continuous data where a linear relationship is expected.

2. **Spearman's Rank Correlation Coefficient (ρ)**:
    1. **Definition**: Measures the strength and direction of the monotonic relationship between two ranked variables.
    2. **Range**: -1 to +1
        1. ρ=+1: Perfect positive rank correlation.
        2. ρ=−1: Perfect negative rank correlation.
        3. ρ=0: No rank correlation.
    3. **Formula**:

    $$\rho = 1 - \frac{6\sum {d_i}^2}{n(n^2-1)}$$

Determining Correlation

1. **Scatter Plots**:
    - **Visual Tool**: Plotting the data points of two variables on a graph to visually inspect the relationship.
    - **Interpretation**: The pattern of the points can suggest positive, negative, or no correlation.
2. **Correlation Matrix**:
    - **Definition**: A table showing correlation coefficients between sets of variables.
    - **Usage**: Common in multivariate analysis to assess relationships between multiple pairs of variables simultaneously.

Question 24. Calculate coefficient of correlation between the marks obtained by 10students in Accountancy and statistics:

Student		2	3	4	5	6	7	8	9	10
Accountancy	45	70	65	30	90	40	50	75	85	60
Statistics	35	90	70	40	95	40	60	80	80	50

Use KarlPearson's Coefficient of Correlation Method to find it.

Accountancy marks: X= {45,70,65,30,90,40,50,75,85,60}
Statistics marks: Y= {35,90,70,40,95,40,60,80,80,50}

$\bar{X} = \frac{\sum X}{n}$ = 61, $\bar{Y} = \frac{\sum Y}{n}$ = 64, $\sum(x_i - \bar{x})(y_i - \bar{y})$ = 3535, $\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}$ = 3910

r = $\frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$ = 3535/3910 = 0.904

Conclusion: Strong positive corelation

Question 25. Discuss the 4 differences between correlation and regression.

Purpose and Interpretation:

- **Correlation**: Measures the strength and direction of the linear relationship between two variables. Symmetric.
- **Regression**: Models the relationship between a dependent variable and one or more independent variables, and makes predictions. Asymmetric.

Analysis Type:

- **Correlation**: Quantifies association without implying causation.
- **Regression**: Examines causal relationships and is used for prediction.

Output:

- **Correlation**: Provides a single statistic, the correlation coefficient (r).
- **Regression**: Produces an equation (e.g., Y=a+bX), along with R2, and regression coefficients.

Applicability:

- **Correlation**: Suitable for assessing the strength and direction of linear relationships.
- **Regression**: Useful for modeling relationships, making predictions, and inferring causality.

Question 26. Find the most likely price of Delhi corresponding to the price of Rs 70 at Agra from the following data:

Coefficient of correlation between the prices of the two places +0.8.

To estimate the price of Delhi corresponding to a price of Rs 70 in Agra with a correlation coefficient of +0.8, we need the means and standard deviations of the prices in both places.

Assuming typical values for demonstration:

- Mean price in Agra $(\overline{X})$: Rs 60
- Mean price in Delhi $(\overline{Y})$:): Rs 80
- Standard deviation of Agra prices (σX): Rs 10
- Standard deviation of Delhi prices (σY): Rs 15

Using these values:

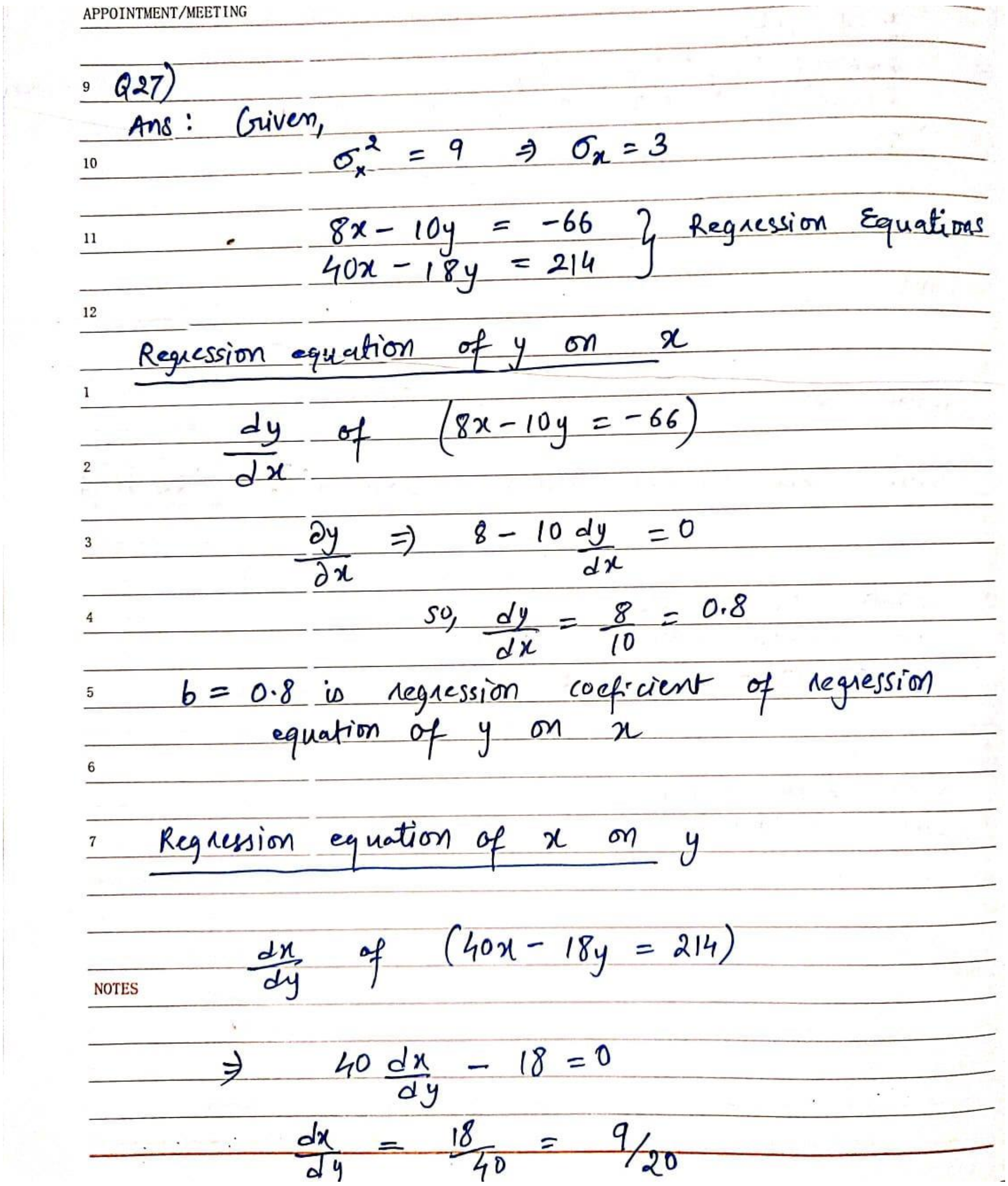
1. **Slope (b)**: b=0.8×15/10 =1.2 (slope = r * $\frac{\sigma y}{\sigma x}$)
2. **Intercept (a)**: a=80−1.2×60=80−72=8 ($\bar{Y}$ - b$\bar{X}$)
3. **Estimated price in Delhi**: Y^=8+1.2×70=8+84=92 (a + bx)
4. **So, the estimated price of Delhi corresponding to Rs 70 in Agra is Rs 92.**

Question 27. In a partially destroyed laboratory record of an analysis of correlation data, the following result only are legible: Variance of x =9, Regression equations are: 1) 8x-10y =-66, 2) 40x-18y=214.

What are

a) The mean value of x and y
b) The coefficient of correlation between x and y.
c) The σ of y

Ans:



APPOINTMENT/MEETING

Q27)

Ans : Given,

$$\sigma_x^2 = 9 \Rightarrow \sigma_x = 3$$

$$\left.\begin{aligned} 8x - 10y &= -66 \\ 40x - 18y &= 214 \end{aligned}\right\}$$ Regression Equations

Regression equation of y on x

$\frac{dy}{dx}$ of $(8x - 10y = -66)$

$$\frac{\partial y}{\partial x} \Rightarrow 8 - 10\frac{dy}{dx} = 0$$

so, $\frac{dy}{dx} = \frac{8}{10} = 0.8$

b = 0.8 is regression coefficient of regression equation of y on x

Regression equation of x on y

$\frac{dx}{dy}$ of $(40x - 18y = 214)$

$$\Rightarrow 40\frac{dx}{dy} - 18 = 0$$

$$\frac{dx}{dy} = \frac{18}{40} = 9/20$$

NOTES

APPOINTMENT/MEETING

$d = 9/20$ is regression coeficient of equation (regression)

9

of x on y

10

11 Regression lines intersect at the means of x & y

12

So, $8\bar{x} - 10\bar{y} = -66$

1

$40\bar{x} - 18\bar{y} = 214$

2

Solving for the equation we get

3

$\bar{x} = 13, \quad \bar{y} = 17$ a) Ans

4

5 coeficient of correlation (r)

6 $r^2 = b \cdot d$

7 $= 0.8 \times 9/20$

$= 0.36$

$r = \pm 0.6$ b) Ans

NOTES

now,

we know $b = r\frac{\sigma_{\cancel{x}y}}{\sigma_x} = 0.6 \times \frac{\sigma_y}{3}$

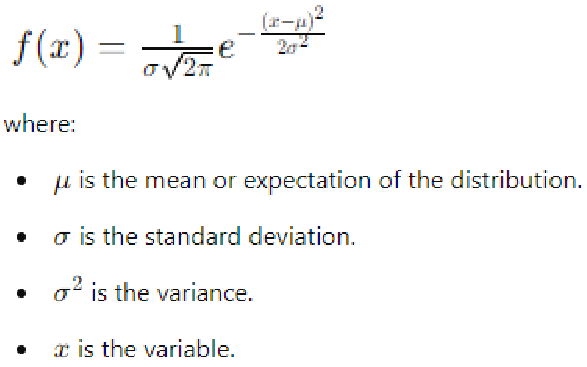
we know $b = 0.8$

so, $\sigma_y = \frac{3 \times 0.8}{0.6}$ 4 c) Ans

Question 28. What is Normal Distribution? What are the four assumptions of normal distribution? Explain in detail.

Normal Distribution

The normal distribution, also known as the Gaussian distribution, is a continuous probability distribution characterized by its symmetric, bell-shaped curve. It is defined by the probability density function:



$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where:

- μ is the mean or expectation of the distribution.
- σ is the standard deviation.
- σ^2 is the variance.
- x is the variable.

The normal distribution is significant in statistics and various fields because of the Central Limit Theorem, which states that the sum (or average) of a large number of independent, identically distributed variables will approximate a normal distribution, regardless of the original distribution of the variables.

Four Assumptions of Normal Distribution:

1. Independence

- **Definition**: Independence means that the value of one observation does not influence or predict the value of another.
- **Importance**: If data points are not independent, the standard errors of estimates can be biased, leading to incorrect conclusions.
- **Example**: In a study measuring the effect of a drug, each patient's response should be independent. If patients are sharing information and influencing each other's responses, this assumption is violated.

2. Linearity

- **Definition**: Linearity means that there is a straight-line relationship between the independent variable(s) and the dependent variable.
- **Importance**: Many statistical methods, including linear regression, assume a linear relationship. Non-linearity can lead to biased estimates and poor predictions.
- **Example**: In a study examining the effect of study time on test scores, a linear relationship means that increasing study time by one hour results in a consistent increase in test scores.

3. Homoscedasticity

- **Definition**: Homoscedasticity means that the variance of errors is the same across all levels of the independent variable(s).

- **Importance**: When the assumption of homoscedasticity is violated (i.e., heteroscedasticity), the efficiency of estimators is compromised, leading to less reliable hypothesis tests.
- **Example**: In a regression model predicting income based on education level, homoscedasticity implies that the variance of income is the same regardless of the level of education.

4. Normality of Errors

- **Definition**: Normality of errors means that the residuals (differences between observed and predicted values) are normally distributed.
- **Importance**: This assumption is critical for inferential statistics, allowing for accurate computation of confidence intervals and significance tests.
- **Example**: In a linear regression model, if the residuals are normally distributed, it implies that most of the errors are small, and large errors are rare.

Question 29. Write all the characteristics or Properties of the Normal Distribution Curve.

Characteristics of the Normal Distribution Curve

1. **Symmetry**:
    - The curve is perfectly symmetrical around its mean, μ.
2. **Mean, Median, and Mode**:
    - The mean, median, and mode of the distribution are all equal and located at the center of the distribution.
3. **Bell-shaped Curve**:
    - The normal distribution forms a bell-shaped curve where the highest point is at the mean.
4. **Asymptotic**:
    - The tails of the curve approach but never touch the horizontal axis, extending infinitely in both directions.
5. **Defined by Mean and Standard Deviation**:
    - The shape and position of the curve are determined by the mean (μ) and the standard deviation (σ).
6. **Empirical Rule (68-95-99.7 Rule)**:
    - About 68% of the data falls within one standard deviation of the mean.
    - About 95% falls within two standard deviations.
    - About 99.7% falls within three standard deviations.
7. **Total Area Under the Curve**:
    - The total area under the normal distribution curve is equal to 1, representing the total probability.

8. **Unimodal**:
    - The curve has a single peak, indicating one mode.
9. **No Skewness**:
    - The skewness of a normal distribution is zero, indicating no skewness.
10. **Kurtosis**:
    - The kurtosis of a normal distribution is zero, indicating a mesokurtic distribution (neither too peaked nor too flat).

Question 30. Which of the following options are correct about Normal Distribution Curve.

a) Within a range of 0.6745 of σ on both sides the middle 50% of the observations occur ie, mean +-0.6745 σ covers 50% area 25% on each side.

b) Mean $\pm$1S.D. (ie $mean \pm 1\sigma$) covers 68.268% area, 34.134% area lies on either side of the mean.

c) Mean $\pm$2S.D. (ie $mean \pm 2\sigma$) covers 95.45% area, 47.725% area lies on either side of the mean.

d) Mean $\pm$3S.D. (ie $mean \pm 3\sigma$) covers 99.73% area, 49.856% area lies on either side of the mean.

e) Only 0.27% area is outside the range $mean \pm 3\sigma$

- **Option (a):** This is incorrect. The range of mean ± 0.6745 standard deviations actually cover approximately 50% of the data, but it's not evenly split with 25% on each side. It's closer to 44% and 6% on each side.

- **Option (b):** This is correct. Mean ± 1 standard deviation covers approximately 68.268% of the data, with 34.134% on each side.

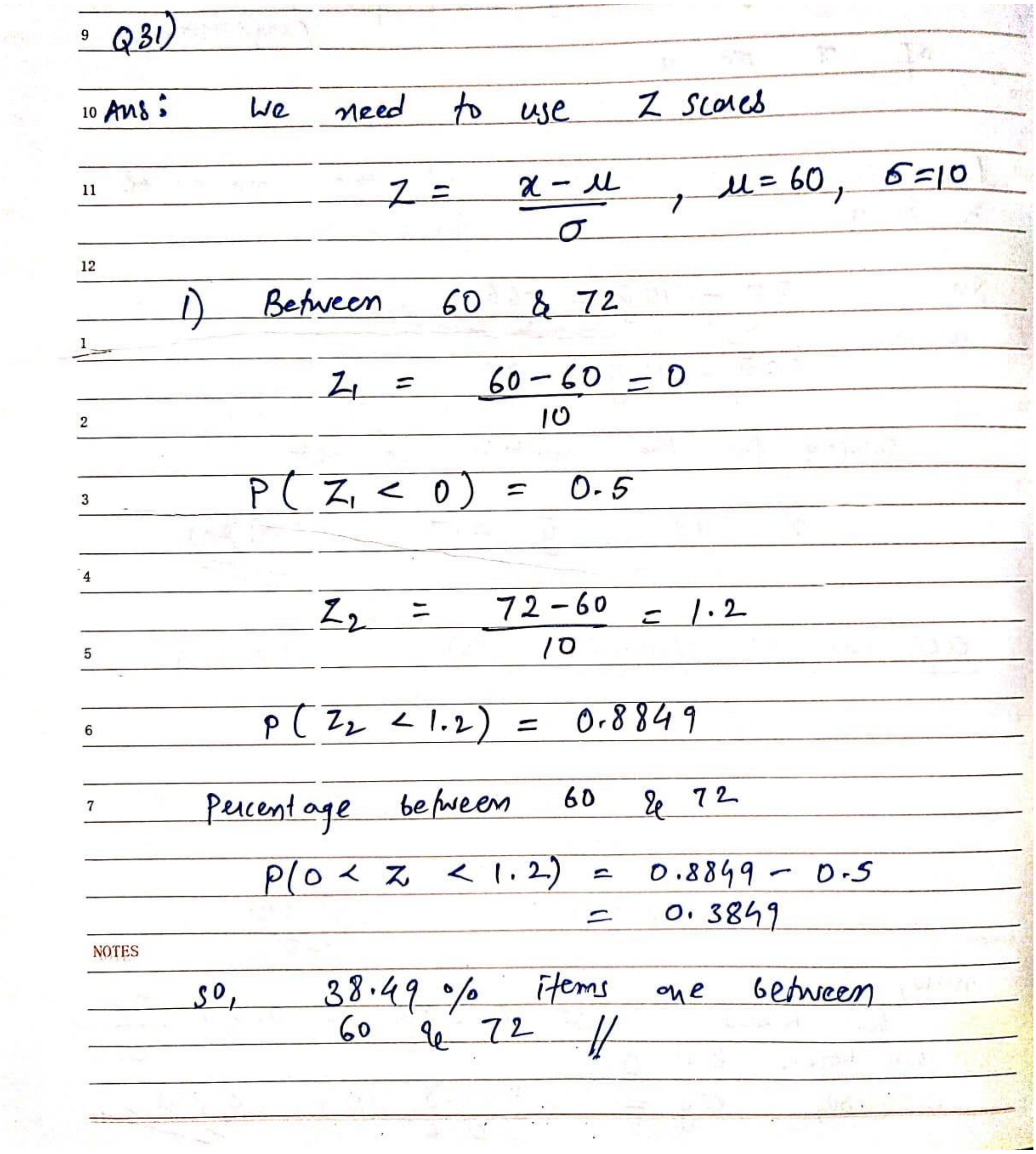
- **Option (c):** This is correct. Mean ± 2 standard deviations cover approximately 95.45% of the data, with 47.725% on each side.

- **Option (d):** This is correct. Mean ± 3 standard deviations cover approximately 99.73% of the data, with 49.856% on each side.

- **Option (e):** This is correct. If 99.73% of the data is within 3 standard deviations, then 100% - 99.73% = 0.27% of the data is outside that range.

Question 31. The mean of a distribution is 60 with a standard deviation of 10. Assuming that the distribution is normal , what percentage of items be

1) Between 60 and 72
2) Between 50 and 60
3) Beyond 72 and
4) Between 70 and 80?



Q31)

Ans : We need to use Z scores

$$Z = \frac{x - \mu}{\sigma}, \quad \mu = 60, \quad \sigma = 10$$

1) Between 60 & 72

$$Z_1 = \frac{60-60}{10} = 0$$

$$P(Z_1 < 0) = 0.5$$

$$Z_2 = \frac{72-60}{10} = 1.2$$

$$P(Z_2 < 1.2) = 0.8849$$

Percentage between 60 & 72

$$P(0 < z < 1.2) = 0.8849 - 0.5$$
$$= 0.3849$$

so, 38.49 % items are between 60 & 72 //

9 2) Between 50 & 60.

10 $Z_1 = \frac{50-60}{10} = -1$

11

$P(Z_1 < -1) = 0.1587$

12

1 $Z_2 = \frac{60-60}{10} = 0$

2 $P(Z_2 < 0) = 0.5$

3 Percentage between 50 & 60 is

4 $P(\cancel{0.1587} -1 < Z < 0) = 0.5 - 0.1587$

$= 0.3413$

5

So, 34.13 % of items are between 50 & 60.

6

7 3) Beyond 72

$Z = \frac{72-60}{10} = 1.2$ SUNDAY 23

NOTES

$P(Z > 1.2) = 1 - P(Z < 1.2)$

$= 1 - 0.8849$

$= 0.1151$

So, 11.51 % of items beyond 72

APPOINTMENT/MEETING

4) Between 70 & 80

9

$$Z_1 = \frac{70-60}{10} = 1$$

10

$$P(Z_1 < 1) = 0.8413$$

11

$$Z_2 = \frac{80-60}{10} = 2$$

12

$$P(Z_2 < 2) = 0.9772$$

1

$$P(1 < Z < 2) = 0.9772 - 0.8413$$
$$= 0.1359$$

2

3

13.59% of items are between 70 & 80

4

5

6

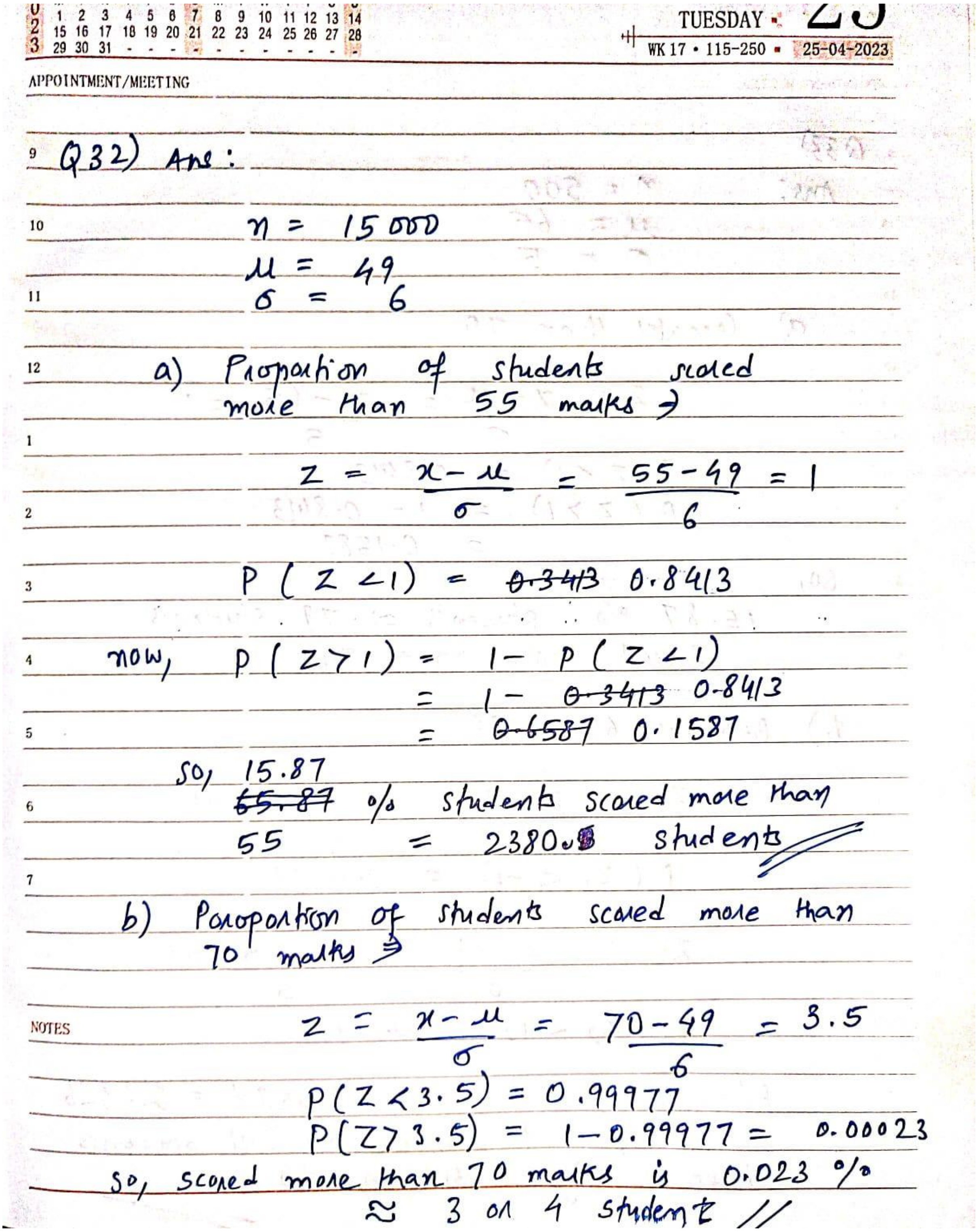
7

NOTES

Question 32. 15000 students sat for an examination. The mean marks was 49 and the distribution of marks had a standard deviation of 6. Assuming that the marks were normally distributed what proportion of students scored

a) More than 55 marks

b) More than 70 marks



1 2 3 4 5 6 7 8 9 10 11 12 13 14
15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31

TUESDAY 25
WK 17 • 115-250 • 25-04-2023

APPOINTMENT/MEETING

9 Q32) Ans :

10 $n = 15000$

$\mu = 49$

11 $\sigma = 6$

12 a) Proportion of students scored more than 55 marks ⇒

1 $$Z = \frac{x-\mu}{\sigma} = \frac{55-49}{6} = 1$$

2

3 $P(Z<1) =$ ~~0.3413~~ 0.8413

4 now, $P(Z>1) = 1 - P(Z<1)$

$= 1 -$ ~~0.3413~~ 0.8413

5 $=$ ~~0.6587~~ 0.1587

6 so, 15.87 ~~55.87~~ % students scored more than 55 = 2380.5 students //

7

b) Poroportion of students scored more than 70 marks ⇒

NOTES

$$Z = \frac{x-\mu}{\sigma} = \frac{70-49}{6} = 3.5$$

$P(Z<3.5) = 0.99977$

$P(Z>3.5) = 1-0.99977 = 0.00023$

so, scored more than 70 marks is 0.023 %

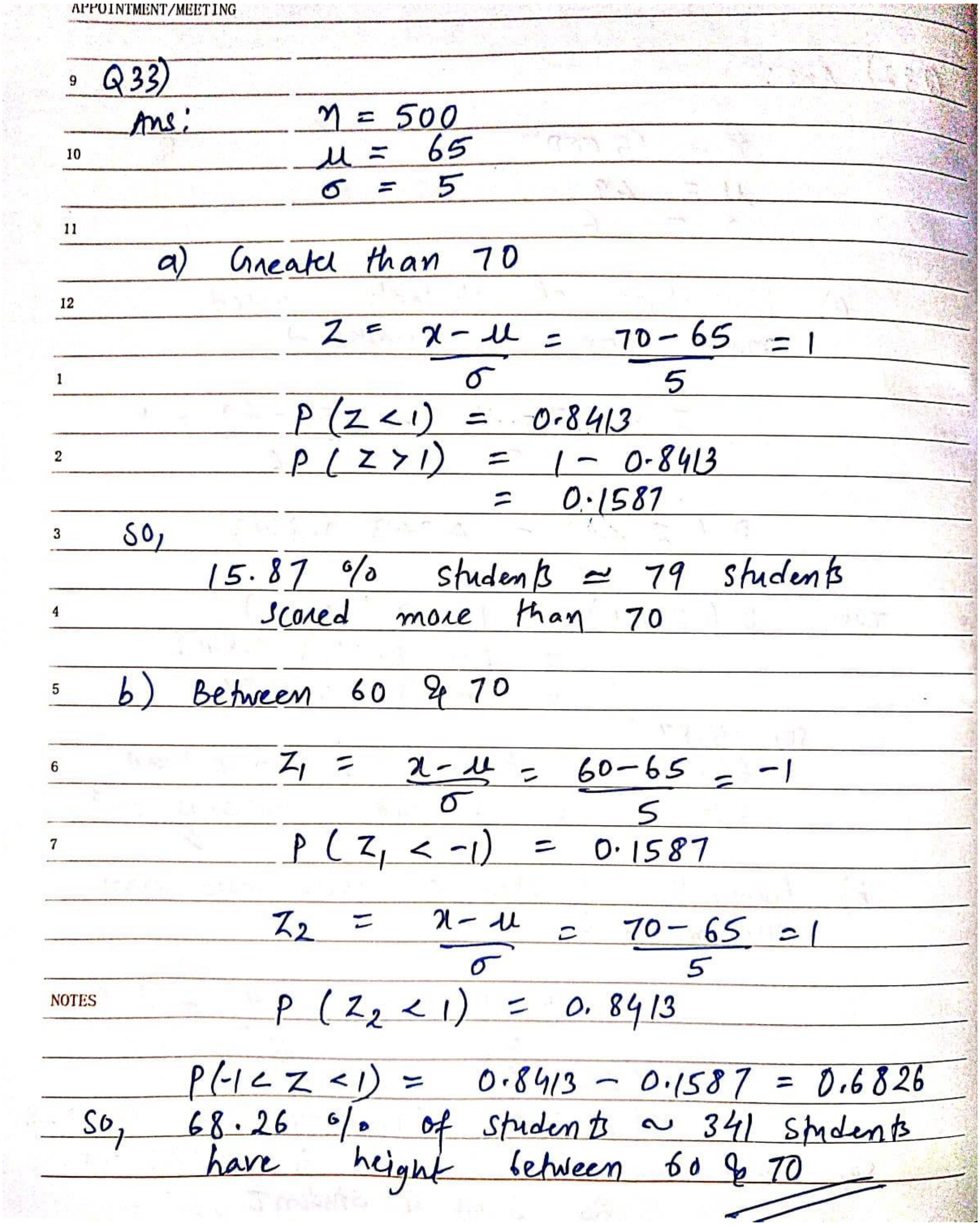
≈ 3 or 4 student //

v

Question 33. If the height of 500 students are normally distributed with mean 65 inch and standard deviation 5 inch. How many students have height :

a) Greater than 70 inch

b) Between 60 and 70 inch



APPOINTMENT/MEETING

9 Q33)

Ans:

$n = 500$

10 $\mu = 65$

$\sigma = 5$

11

a) Greater than 70

12

$$Z = \frac{x-\mu}{\sigma} = \frac{70-65}{5} = 1$$

1

2 $P(Z<1) = 0.8413$

$P(Z>1) = 1 - 0.8413$

$= 0.1587$

3 So,

15.87 % students ≃ 79 students

4 scored more than 70

5 b) Between 60 & 70

6 $$Z_1 = \frac{x-\mu}{\sigma} = \frac{60-65}{5} = -1$$

7 $P(Z_1 < -1) = 0.1587$

$$Z_2 = \frac{x-\mu}{\sigma} = \frac{70-65}{5} = 1$$

NOTES $P(Z_2 < 1) = 0.8413$

$P(-1 < Z < 1) = 0.8413 - 0.1587 = 0.6826$

So, 68.26 % of students ~ 341 students have height between 60 & 70

Question 34. What is the statistical hypothesis? Explain the errors in hypothesis testing. Explain the sample. What are large samples and small samples?

A statistical hypothesis is a specific claim or assertion about a population parameter, such as the mean or proportion. It is tested using sample data to determine whether there is enough evidence to reject it. There are two types of hypotheses:

- **Null Hypothesis (H0)**: Assumes no effect or no difference. It is the hypothesis that is initially assumed to be true.
- **Alternative Hypothesis (H1 or Ha)**: Contradicts the null hypothesis. It represents the effect or difference that the test aims to detect.

Errors in Hypothesis Testing

1. **Type I Error (α)**:
   - Occurs when the null hypothesis is true, but we incorrectly reject it.
   - The probability of making a Type I error is denoted by α (alpha), often set at 0.05 or 5%.
2. **Type II Error (β)**:
   - Occurs when the null hypothesis is false, but we fail to reject it.
   - The probability of making a Type II error is denoted by β (beta).

Sample

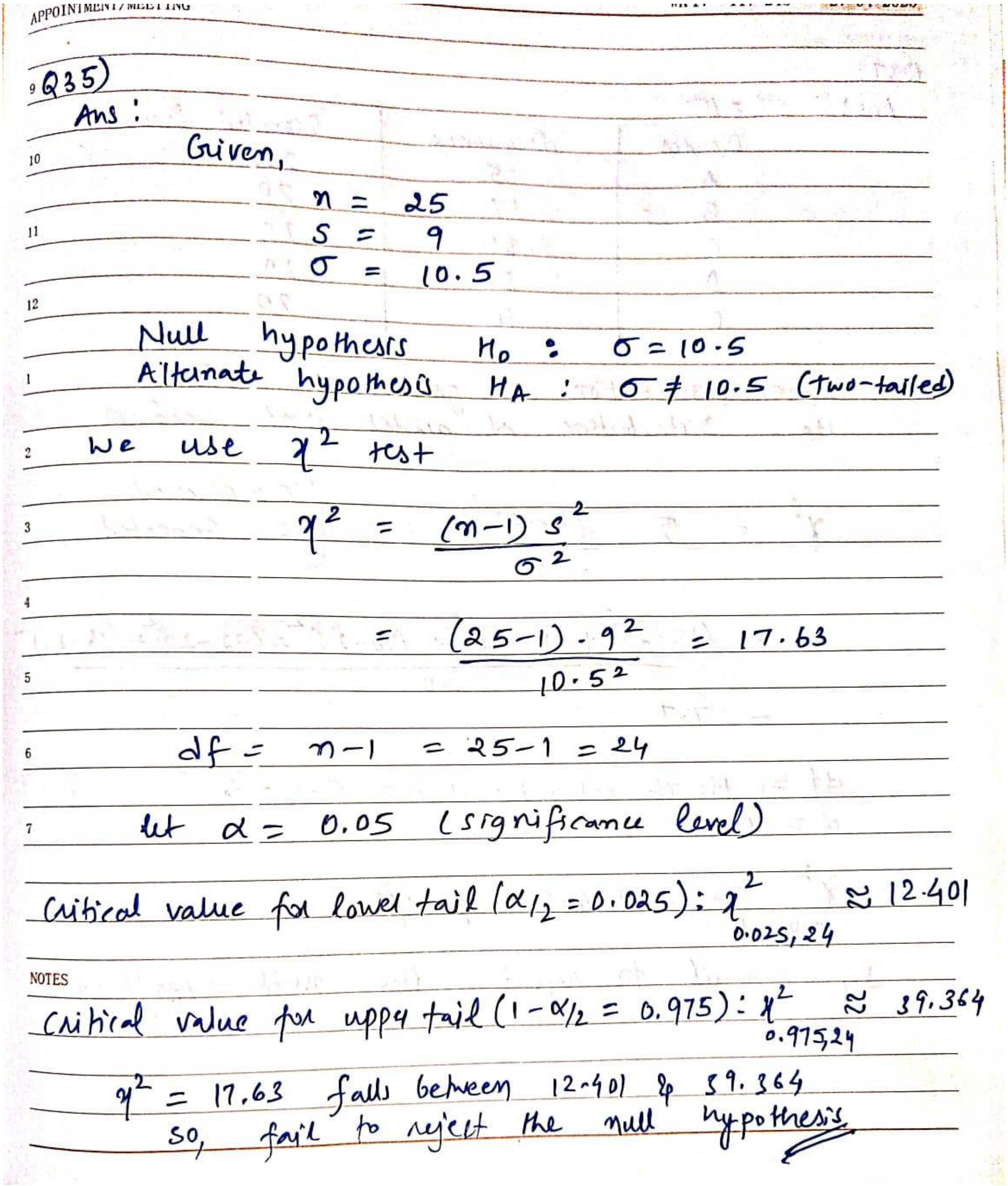
A sample is a subset of a population used to make inferences about the entire population. It should be representative to ensure accurate and reliable conclusions.

Large Samples vs. Small Samples

- **Large Samples**:
  - Typically considered to be samples with $n>30$.
  - Tend to provide more reliable estimates and normal distribution assumptions can be applied due to the Central Limit Theorem.
- **Small Samples**:
  - Typically considered to be samples with $n\leq30$.
  - Often require specific distributions (like t-distribution) and more careful statistical techniques to account for higher variability and potential bias.

Question 35. A random sample of size 25 from a population gives the sample standard deviation to be 9.0. Test the hypothesis that the population standard deviation is 10.5

Hint: Use chi-square distribution



APPOINTMENT / MEETING

Q35)

Ans :

Given,

$n = 25$

$s = 9$

$\sigma = 10.5$

Null hypothesis H_0 : $\sigma = 10.5$

Alternate hypothesis H_A : $\sigma \neq 10.5$ (two-tailed)

We use χ^2 test

$$\chi^2 = \frac{(n-1)\,s^2}{\sigma^2}$$

$$= \frac{(25-1)\cdot 9^2}{10.5^2} = 17.63$$

$df = n-1 = 25-1 = 24$

let $\alpha = 0.05$ (significance level)

Critical value for lower tail ($\alpha/2 = 0.025$): $\chi^2_{0.025,24} \approx 12.401$

NOTES

Critical value for upper tail ($1-\alpha/2 = 0.975$): $\chi^2_{0.975,24} \approx 39.364$

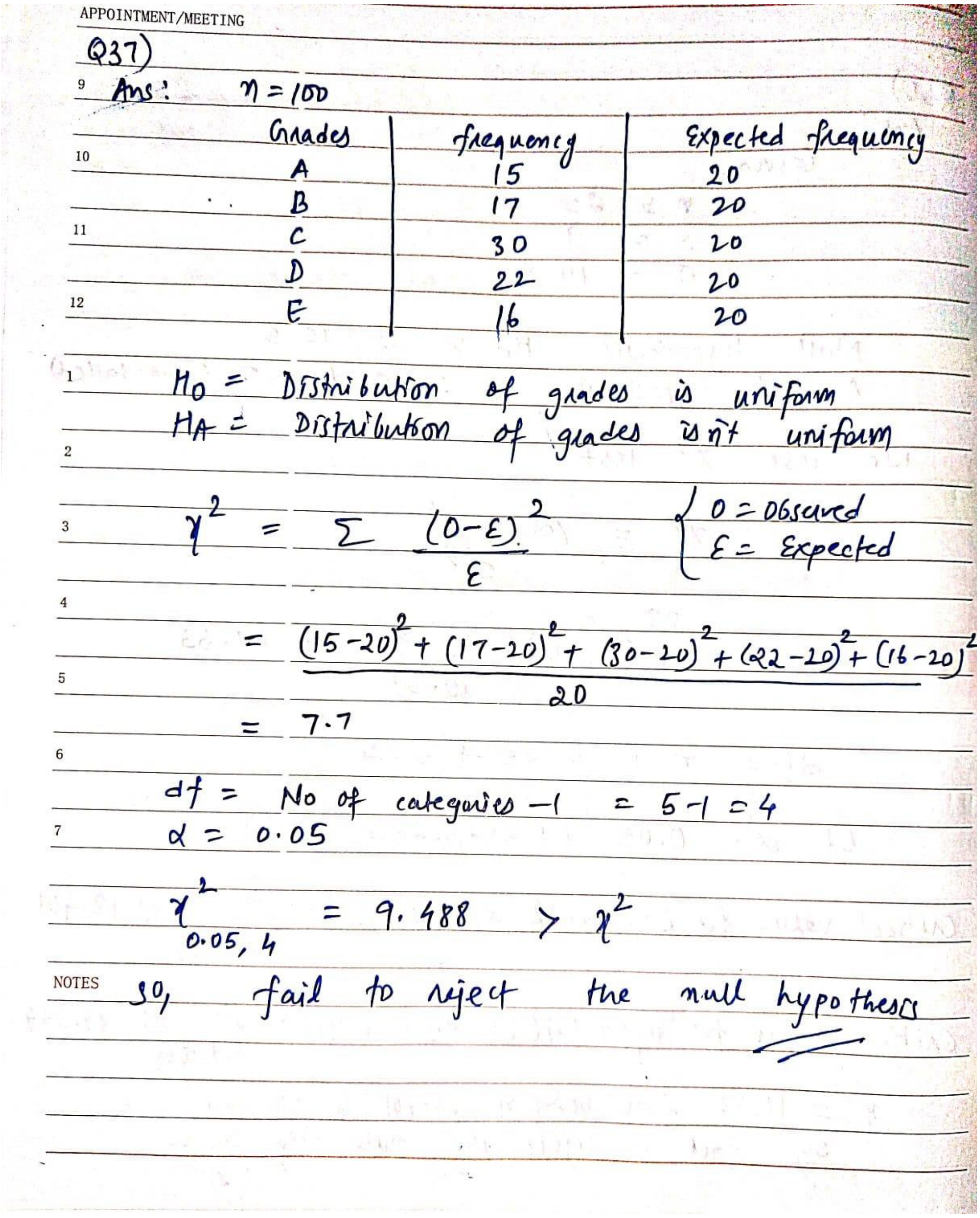
$\chi^2 = 17.63$ falls between 12.401 & 39.364

so, fail to reject the null hypothesis.

Question 37. 100 students of PW IOI obtained the following grades in Data Science paper:

Grade: [A,B,C,D,E]
Total Frequence: [15,17,30,22,16,100]

Using chi square test examine the hypothesis that the distribution of grades is uniform.



APPOINTMENT/MEETING

Q37)

Ans: $n = 100$

Grades	frequency	Expected frequency
A	15	20
B	17	20
C	30	20
D	22	20
E	16	20

H_0 = Distribution of grades is uniform
H_A = Distribution of grades is n't uniform

$$\chi^2 = \Sigma \frac{(O-E)^2}{E} \quad \left\{ \begin{array}{l} O = \text{Observed} \\ E = \text{Expected} \end{array} \right.$$

$$= \frac{(15-20)^2 + (17-20)^2 + (30-20)^2 + (22-20)^2 + (16-20)^2}{20}$$

$$= 7.7$$

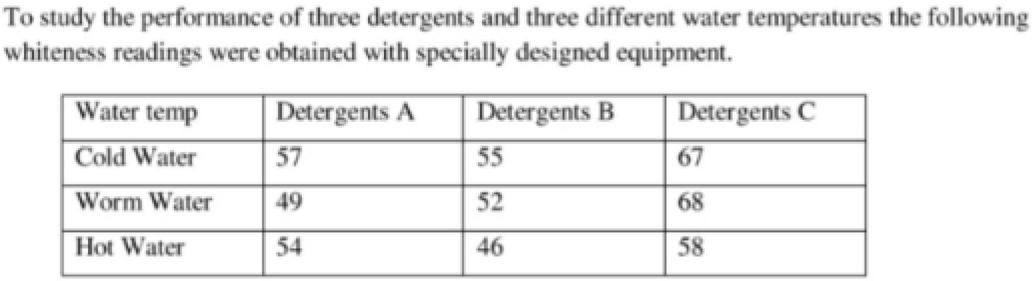
df = No of categories -1 = $5-1 = 4$
$\alpha = 0.05$

$$\chi^2_{0.05,\,4} = 9.488 > \chi^2$$

NOTES

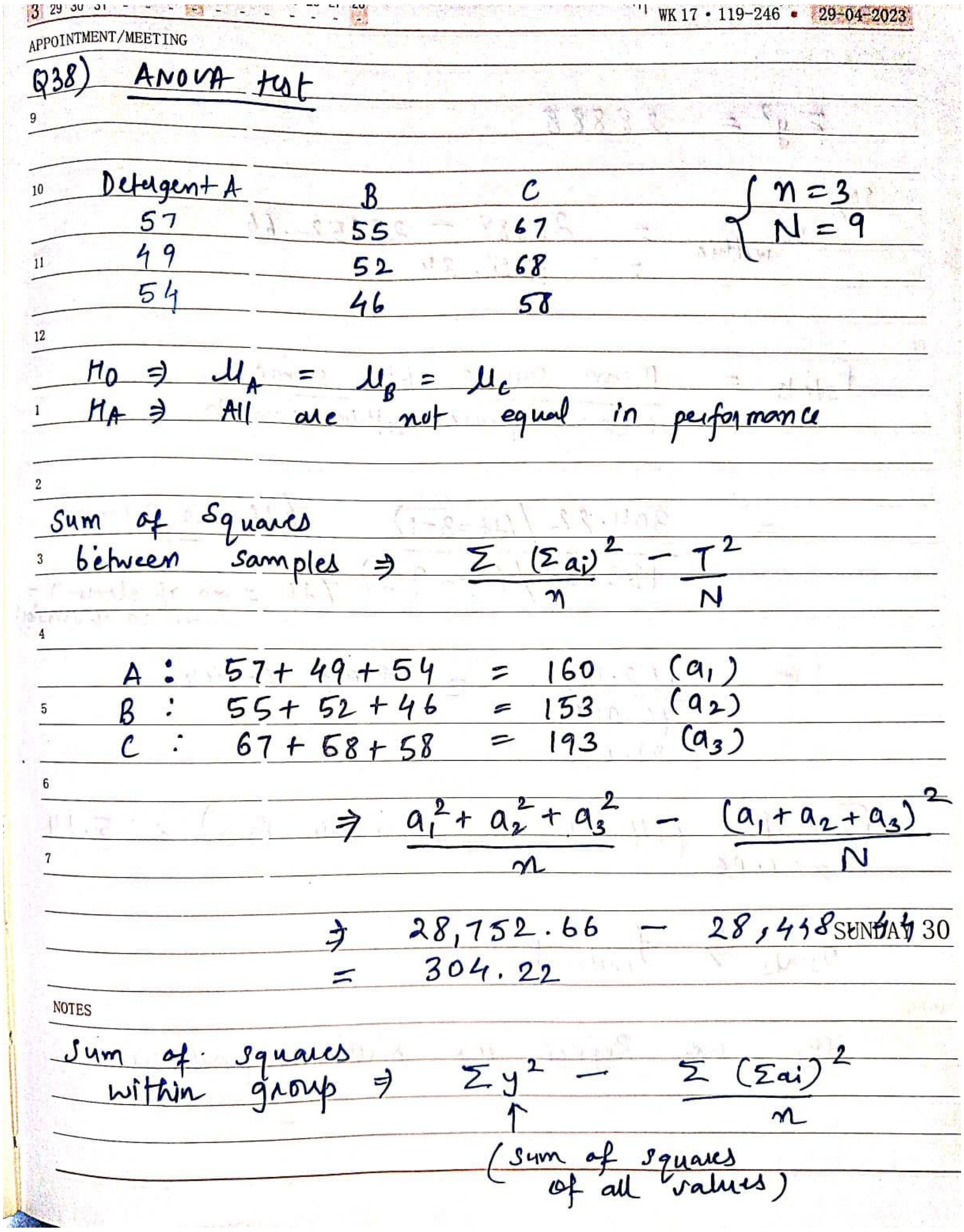
so, fail to reject the null hypothesis

Question 38. Anova Test:



To study the performance of three detergents and three different water temperatures the following whiteness readings were obtained with specially designed equipment.

Water temp	Detergents A	Detergents B	Detergents C
Cold Water	57	55	67
Worm Water	49	52	68
Hot Water	54	46	58



WK 17 • 119-246 • 29-04-2023

APPOINTMENT/MEETING

Q38) ANOVA test

Detergent A	B	C
57	55	67
49	52	68
54	46	58

$$\left\{ \begin{array}{l} n = 3 \\ N = 9 \end{array} \right.$$

$H_0 \Rightarrow \mu_A = \mu_B = \mu_C$

$H_A \Rightarrow$ All are not equal in performance

Sum of Squares between samples $\Rightarrow \dfrac{\Sigma (\Sigma a_i)^2}{n} - \dfrac{T^2}{N}$

A : 57 + 49 + 54 = 160 (a_1)

B : 55 + 52 + 46 = 153 (a_2)

C : 67 + 68 + 58 = 193 (a_3)

$$\Rightarrow \frac{a_1^2 + a_2^2 + a_3^2}{n} - \frac{(a_1 + a_2 + a_3)^2}{N}$$

$$\Rightarrow 28{,}752.66 - 28{,}448.44$$

$$= 304.22$$

SUNDAY 30

NOTES

Sum of squares within group $\Rightarrow \underset{\substack{\uparrow \\ \text{(Sum of squares of all values)}}}{\Sigma y^2} - \dfrac{\Sigma (\Sigma ai)^2}{n}$

$\Sigma y^2 = 28888$

So,

$$SS_{within} = 28888 - 28752.66$$
$$= 135.34$$

$$f_{stats} = \frac{\text{Mean square btw sample}}{\text{Mean square within sample}}$$

$$= \frac{304.22 / (df = 3-1)}{135.34 / (df = 9-3)}$$ (df = no of samples − 1) (df = no of elements − total no of samples)

$$= \frac{152.11}{\cancel{16.9175}\ 22.556} = \cancel{8.991}\ 6.744$$

$f_{critical}$ $\alpha = 0.05$ (df btw = 2, df within = 6) = 5.14

$f_{stats} > f_{critical}$

NOTES

So, we Reject the Null hypothesis

Question 39. How would you create a basic Flask route that displays "Hello,World !" on the homepage.

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello, World!"

if __name__=="main":
    app.run(host="0.0.0.0")
```

Question 40. Explain how to set up a Flask application to handle form submissions using POST requests.

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/submit', methods=['POST'])
def submit():
    data = request.form['data']
    return f"Received: {data}"

if __name__=="main":
    app.run(host="0.0.0.0")
```

Question 41. Write a Flask route that accepts a parameter in the URL and displays it on the page.

```
@app.route('/user/<name>')
def user(name):
    return f"Hello, {name}!"
```

Question 42. How can you implement user authentication in a Flask application?

Implementing user authentication in a Flask application involves several steps, including user registration, login, and session management. Here's a general outline of the process:

Database Setup

Use Flask-SQLAlchemy to set up the database and define a user model.

```python
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from flask_login import LoginManager, UserMixin

app = Flask(__name__)
app.config['SECRET_KEY'] = 'your_secret_key'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'

db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)
login_manager.login_view = 'login'
login_manager.login_message_category = 'info'

class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20), unique=True, nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    password = db.Column(db.String(60), nullable=False)

    def __repr__(self):
        return f"User('{self.username}', '{self.email}')"

@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))


if __name__=="main":
    app.run(host="0.0.0.0")
```

User Registration

Create a route for user registration where you collect user details and hash the password using Flask-Bcrypt.

```python
from flask import render_template, url_for, flash, redirect
from yourforms import RegistrationForm

@app.route("/register", methods=['GET', 'POST'])
def register():
    form = RegistrationForm()
    if form.validate_on_submit():
        hashed_password = bcrypt.generate_password_hash(form.password.data).decode('utf-8')
        user = User(username=form.username.data, email=form.email.data, password=hashed_password)
        db.session.add(user)
        db.session.commit()
        flash('Your account has been created! You can now log in', 'success')
        return redirect(url_for('login'))
    return render_template('register.html', title='Register', form=form)
```

User Login

Create a route for user login, verifying the credentials and managing the session using Flask-Login.

```
from flask import request
from flask_login import login_user, current_user, logout_user
from yourforms import LoginForm

@app.route("/login", methods=['GET', 'POST'])
def login():
    if current_user.is_authenticated:
        return redirect(url_for('home'))
    form = LoginForm()
    if form.validate_on_submit():
        user = User.query.filter_by(email=form.email.data).first()
        if user and bcrypt.check_password_hash(user.password, form.password.data):
            login_user(user, remember=form.remember.data)
            next_page = request.args.get('next')
            return redirect(next_page) if next_page else redirect(url_for('home'))
        else:
            flash('Login Unsuccessful. Please check email and password', 'danger')
    return render_template('login.html', title='Login', form=form)
```

User Logout

Add a route for logging out users, which ends their session.

```
@app.route("/logout")
def logout():
    logout_user()
    return redirect(url_for('home'))
```

Protecting Routes

Use the @login_required decorator to protect routes that require authentication.

```
from flask_login import login_required

@app.route("/account")
@login_required
def account():
    return render_template('account.html', title='Account')
```

Question 43. Describe the process of connecting a Flask app to a SQlite database using SQLAlchemy.

```
#Install Required Packages:

pip install Flask Flask-SQLAlchemy


#Setup Flask App:

        from flask import Flask
        from flask_sqlalchemy import SQLAlchemy

        app = Flask(__name__)
        app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'
        app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
        db = SQLAlchemy(app)

#Define Database Models:

    class User(db.Model):
        id = db.Column(db.Integer, primary_key=True)
        username = db.Column(db.String(20), unique=True, nullable=False)
        email = db.Column(db.String(120), unique=True, nullable=False)
        password = db.Column(db.String(60), nullable=False)


#Create the Database:

    from yourapplication import db
    db.create_all()

#Interacting with the Database:

#Create:

    user = User(username='John', email='john@example.com', password='password')
    db.session.add(user)
    db.session.commit()

#Read:

user = User.query.filter_by(username='John').first()


#Update:

user.email = 'newemail@example.com'
db.session.commit()


#Delete:

db.session.delete(user)
db.session.commit()
```

Question 44. How would you create a RESTful API endpoint in Flask that returns JSON data?

```
from flask import jsonify

@app.route('/api/data')
def data():
    return jsonify({'key': 'value'})
```

Question 45. Explain how to use Flask-WTF to create and validate forms in a flask application

Import FlaskForm and Define a Form Class:

- Create a form class by subclassing FlaskForm.
- Define form fields as class variables using Flask-WTF's field classes.

```
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, SubmitField
from wtforms.validators import DataRequired, Length, Email

class RegistrationForm(FlaskForm):
    username = StringField('Username', validators=[DataRequired(), Length(min=2,
max=20)])
    email = StringField('Email', validators=[DataRequired(), Email()])
    password = PasswordField('Password', validators=[DataRequired()])
    submit = SubmitField('Sign Up')
```

Render the Form in a Template:

- Pass the form to your template in the view function.
- Use Jinja2 syntax to render the form fields and handle errors.

```
from flask import render_template, Flask

app = Flask(__name__)
app.config['SECRET_KEY'] = 'your_secret_key'

@app.route('/register', methods=['GET', 'POST'])
def register():
    form = RegistrationForm()
    if form.validate_on_submit():
        # Process form data
        return 'Form Submitted!'
    return render_template('register.html', form=form)
```

```
<!-- register.html -->
<form method="POST">
    {{ form.hidden_tag() }}
    <div>
        {{ form.username.label }} {{ form.username() }}
        {% for error in form.username.errors %}
            <span style="color: red;">[{{ error }}]</span>
        {% endfor %}
    </div>
    <div>
        {{ form.email.label }} {{ form.email() }}
        {% for error in form.email.errors %}
            <span style="color: red;">[{{ error }}]</span>
        {% endfor %}
    </div>
    <div>
        {{ form.password.label }} {{ form.password() }}
        {% for error in form.password.errors %}
            <span style="color: red;">[{{ error }}]</span>
        {% endfor %}
    </div>
    <div>{{ form.submit() }}</div>
</form>
```

Handle Form Validation:

- In the view function, use form.validate_on_submit() to check if the form is submitted and valid.
- Handle valid and invalid form submissions accordingly.

Question 46. How can you implement file uploads in Flask application?

```
#Setup Configuration:

#Configure upload folder and allowed file extensions.

        app.config['UPLOAD_FOLDER'] = 'uploads/'
        ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}

#Check Allowed File Types:

#Function to check if the uploaded file is allowed.

    def allowed_file(filename):
        return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


#Create Upload Route:

#Handle file upload in a route.

    @app.route('/upload', methods=['GET', 'POST'])
    def upload_file():
        if request.method == 'POST':
            file = request.files['file']
            if file and allowed_file(file.filename):
                filename = secure_filename(file.filename)
                file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
                return 'File uploaded successfully!'
        return '''
        <form method=post enctype=multipart/form-data>
          <input type=file name=file>
          <input type=submit value=Upload>
        </form>
        '''
```

```
#Ensure Upload Directory Exists:

#Run Flask Application
```

Question 47. Describe the steps to create a Flask Blueprint and why you might use one.

Import and Create a Blueprint:

- Import the Blueprint class from Flask and create an instance.

```
from flask import Blueprint

main = Blueprint('main', __name__)
```

Define Routes within the Blueprint:

- Define the routes and views associated with the blueprint.

```
@main.route('/')
def index():
    return "Welcome to the main page!"
```

Register the Blueprint in the Main Application:

- In your main application file, import and register the blueprint with the Flask app instance.

```
from flask import Flask
from main import main as main_blueprint

app = Flask(__name__)
app.register_blueprint(main_blueprint)
```

Why Use a Flask Blueprint?

1. **Modular Code Organization:**
   - Blueprints allow you to separate different parts of your application into distinct modules. This makes the codebase easier to manage and understand, especially in larger applications.
2. **Code Reusability:**
   - Blueprints enable the reuse of code across different projects. You can encapsulate a set of related routes and logic into a blueprint and then include it in multiple applications.
3. **Separation of Concerns:**
   - By using blueprints, you can keep related routes and views together, separating them from the main application logic. This separation enhances code maintainability and readability.
4. **Collaboration and Scaling:**
   - Blueprints are helpful when multiple developers are working on the same project. Each developer can work on different blueprints, minimizing conflicts and improving productivity.

5. **Enhanced URL Prefixing and Routing:**
    - Blueprints provide a convenient way to group related routes under a common URL prefix, making it easier to manage and update URLs.

Question 48. How would you deploy a Flask application to a production server using Gunicorn and Nginx?

To deploy a Flask application using Gunicorn and Nginx:

1. **Install Dependencies:**
    - Ensure gunicorn and nginx are installed on your server.
2. **Run Gunicorn:**
    - Start Gunicorn with your Flask app:
      gunicorn -w 4 -b 0.0.0.0:8000 yourapp:app
      (Replace yourapp:app with your actual Flask app module and instance.)

Configure Nginx:

- Create a configuration file for your site in /etc/nginx/sites-available/

```
server {
    listen 80;
    server_name your_domain_or_IP;

    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Link the configuration file to /etc/nginx/sites-enabled/:

```
sudo ln -s /etc/nginx/sites-available/your_site /etc/nginx/sites-enabled/
```

Restart Nginx:

```
sudo systemctl restart nginx
```

Question 49.

Practical problem ---refer GitHub code.

Question 50. Machine Learning

1. **What is the difference between Series & Dataframes.**

   A Series is a one-dimensional array-like object in pandas, which can hold data of any type (integers, strings, floating point numbers, etc.). It has an index, which labels each element in the Series.

   A DataFrame is a two-dimensional, tabular data structure in pandas, similar to a spreadsheet or SQL table. It consists of multiple Series, each of which can be of different types, and it has both row and column indices.

2. **Create a database name Travel_Planner in mysql, create a table name "bookings" in that which having attributes (user_id INT, flight_id INT, hotel_id INT, activity_id INT, booking_date DATE). Fill with some dummy value. Now you have to read the content to this table using pandas as dataframe. Show the output.**

   SQL:

```sql
-- Connect to MySQL and create the database
CREATE DATABASE Travel_Planner;

-- Use the created database
USE Travel_Planner;

-- Create the bookings table
CREATE TABLE bookings (
    user_id INT,
    flight_id INT,
    hotel_id INT,
    activity_id INT,
    booking_date DATE
);

-- Insert some dummy values into the bookings table
INSERT INTO bookings (user_id, flight_id, hotel_id, activity_id, booking_date)
VALUES
(1, 101, 201, 301, '2023-07-01'),
(2, 102, 202, 302, '2023-07-02'),
(3, 103, 203, 303, '2023-07-03');
```

   Python:

```python
import pandas as pd
import mysql.connector

# Connect to the MySQL database
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='**********',
    database='travel_planner'
)

# Query the bookings table
query = "SELECT * FROM bookings"
```

```
# Load the data into a pandas DataFrame
df = pd.read_sql(query, conn)

# Close the connection
conn.close()

# Display the DataFrame
print(df)
```

3. **Difference between loc and iloc**
   - **loc**:
     - Label-based indexing for DataFrame rows and columns.
     - Example: df.loc[1:3, 'column_name'] accesses rows 1 to 3 for the specified column.
   - **iloc**:
     - Integer-based indexing for DataFrame rows and columns.
     - Example: df.iloc[1:3, 0] accesses rows 1 to 3 for the first column (index 0).

4. **What is the difference between supervised and unsupervised learning?**
   - **Supervised Learning**:
     - Uses labeled data (input-output pairs).
     - The model learns to predict outputs from inputs.
     - Common tasks: Classification (e.g., spam detection), Regression (e.g., house price prediction).
   - **Unsupervised Learning**:
     - Uses unlabeled data (no explicit output labels).
     - The model finds patterns or structures in the data.
     - Common tasks: Clustering (e.g., customer segmentation), Dimensionality Reduction (e.g., PCA).

5. **Explain the bias-variance tradeoff.**
   - **Bias**:
     - Error due to overly simplistic models.
     - Causes underfitting: The model cannot capture the underlying trend of the data.
   - **Variance**:
     - Error due to overly complex models.
     - Causes overfitting: The model captures noise and random fluctuations in the training data.
   - **Tradeoff**:
     - Aim to find a balance where the model is neither too simple nor too complex.
     - Minimize total error by balancing bias and variance.

6. **What are precision and recall? How are they different from accuracy?**
   - **Precision**:
     - Definition: Precision= TP ÷ (TP + FP)
     - Measures the accuracy of positive predictions.
   - **Recall**:
     - Definition: Recall= TP ÷ (TP +FN)

    - Measures the ability to identify all actual positives.
- **Accuracy**:
    - Definition: Accuracy= (TP+TN)/(TP+TN+FP+FN)
    - Measures the overall correctness of the model.
- **Differences**:
    - Precision focuses on the correctness of positive predictions.
    - Recall focuses on capturing all positive instances.
    - Accuracy considers both positive and negative predictions.

7. **What is overfitting and how can it be prevented?**
    - **Overfitting**:
        - Occurs when a model learns the training data too well, including noise and outliers.
        - Results in poor generalization to new, unseen data.
    - **Prevention**:
        - **Cross-Validation**: Use techniques like k-fold cross-validation to evaluate model performance.
        - **Regularization**: Add penalties for large coefficients (L1 or L2 regularization).
        - **Pruning**: Reduce the size of decision trees.
        - **Simpler Models**: Use less complex models with fewer parameters.
        - **More Training Data**: Increase the amount of training data to reduce overfitting.

8. **Explain the concept of cross-validation.**
    - **Cross-Validation**:
        - Technique to evaluate model performance and ensure it generalizes well to unseen data.
        - **k-fold Cross-Validation**: Split data into k subsets (folds), train on k-1 folds, and test on the remaining fold. Repeat k times and average the results.

9. **What is the difference between a classification and a regression problem?**
    - **Classification**:
        - Predicts discrete labels (e.g., spam or not spam).
        - Example: Email classification.
    - **Regression**:
        - Predicts continuous values (e.g., house prices).
        - Example: Predicting stock prices.

10. **Explain the concept of ensemble learning.**
    - **Ensemble Learning**:
        - Combines multiple models to improve overall performance.
        - **Bagging**: Combines predictions from multiple models trained on different subsets of the data (e.g., Random Forest).
        - **Boosting**: Sequentially trains models, with each new model focusing on correcting errors of previous ones (e.g., AdaBoost, XGBoost).

11. **What is gradient descent and how does it work?**
    - **Gradient Descent**:
        - Optimization algorithm to minimize the cost function.
        - Iteratively adjusts model parameters in the direction of the negative gradient (steepest descent).
        - Steps:
            1. Initialize parameters.
            2. Compute the gradient of the cost function.
            3. Update parameters using the learning rate.
            4. Repeat until convergence.

12. **Describe the difference between batch gradient descent and stochastic gradient descent.**
    - **Batch Gradient Descent**:
        - Uses the entire dataset to compute the gradient.
        - More stable updates but can be slow for large datasets.
    - **Stochastic Gradient Descent (SGD)**:
        - Uses a single data point to compute the gradient.
        - Faster and more efficient for large datasets but more noisy updates.

13. **What is the curse of dimensionality in machine learning?**
    - **Curse of Dimensionality**:
        - As the number of features (dimensions) increases, the volume of the feature space increases exponentially.
        - Leads to sparse data and makes it harder to find patterns and relationships.
        - Models may require exponentially more data to maintain performance.

14. **Explain the difference between L1 and L2 regularization.**
    - **L1 Regularization (Lasso)**:
        - Adds the absolute value of coefficients as a penalty to the cost function.
        - Promotes sparsity, leading to feature selection (some coefficients become zero).
    - **L2 Regularization (Ridge)**:
        - Adds the squared value of coefficients as a penalty to the cost function.
        - Shrinks coefficients but does not lead to zero coefficients.
        - Helps prevent overfitting by constraining large weights.

15. **What is a confusion matrix and how is it used?**
    - **Confusion Matrix**:
        - A table used to evaluate the performance of a classification model.
        - Shows the counts of true positives, false positives, true negatives, and false negatives.
        - Helps calculate metrics like accuracy, precision, recall, and F1-score.

16. **Define AUC-ROC curve.**
    - **AUC-ROC Curve**:
        - **ROC Curve**: Graphical representation of a classifier's performance by plotting True Positive Rate (TPR) against False Positive Rate (FPR) at various threshold settings.

   - **AUC (Area Under the Curve)**: Measures the entire two-dimensional area underneath the ROC curve.
   - Higher AUC indicates a better performing model.

17. **Explain the k-nearest neighbors algorithm.**
    - **k-Nearest Neighbors (k-NN)**:
       - A simple, instance-based learning algorithm.
       - Classifies a data point based on the majority class of its k nearest neighbors in the feature space.
       - Distance metrics like Euclidean distance are used to find the nearest neighbors.

18. **Explain the basic concept of a Support Vector Machine (SVM).**
    - **Support Vector Machine (SVM)**:
       - A supervised learning algorithm used for classification and regression.
       - Finds the optimal hyperplane that maximizes the margin between different classes.
       - Support vectors are the data points closest to the hyperplane.

19. **How does the kernel trick work in SVM?**
    - **Kernel Trick**:
       - Transforms data into a higher-dimensional space to make it linearly separable.
       - Allows SVM to find a nonlinear decision boundary.
       - Common kernels: Linear, Polynomial, Radial Basis Function (RBF).

20. **What are the different types of kernels used in SVM and when would you use each?**
    - **Linear Kernel**: When data is linearly separable.
    - **Polynomial Kernel**: When data requires polynomial separation.
    - **RBF Kernel**: For non-linear separation, widely used and powerful for various data distributions.
    - **Sigmoid Kernel**: Used in neural networks, but less common in SVMs.

21. **What is the hyperplane in SVM and how is it determined?**
    - **Hyperplane**:
       - A decision boundary that separates different classes.
       - Determined by maximizing the margin between the closest data points of different classes (support vectors).

22. **What are the pros and cons of using a Support Vector Machine (SVM)?**
    - **Pros**:
       - Effective in high-dimensional spaces.
       - Works well with clear margin of separation.
       - Robust to overfitting, especially in high-dimensional space.
    - **Cons**:
       - Not suitable for very large datasets.
       - Not as effective when there is a lot of noise.
       - Requires careful tuning of hyperparameters.

23. **Explain the difference between a hard margin and a soft margin SVM.**
    - **Hard Margin SVM**:
        - No tolerance for misclassification.
        - Assumes data is perfectly linearly separable.
    - **Soft Margin SVM**:
        - Allows some misclassification to handle noisy data.
        - Balances margin maximization and classification error minimization.

24. **Describe the process of constructing a decision tree.**
    - **Construction Process**:
        1. Start with the entire dataset.
        2. Select the best attribute to split the data based on criteria like information gain or Gini impurity.
        3. Split the data into subsets.
        4. Recursively repeat the process for each subset until stopping criteria are met (e.g., all instances in a subset belong to the same class).

25. **Describe the working principle of a decision tree.**
    - **Working Principle**:
        - Splits data into subsets based on the value of input features.
        - Creates branches for each possible value and continues splitting until it reaches a leaf node.
        - Leaf nodes represent the final decision or classification.

26. **What is information gain and how is it used in decision trees?**
    - **Information Gain**:
        - Measure of the reduction in entropy or uncertainty after a dataset is split on an attribute.
        - Used to select the best attribute for splitting at each node in a decision tree.
        - Higher information gain indicates a better attribute for splitting.

27. **Explain Gini impurity and its role in decision trees.**
    - **Gini Impurity**:
        - Measure of the probability of misclassifying a randomly chosen element if it was labeled according to the distribution of labels in a subset.
        - Used as a criterion for splitting in decision trees.
        - Lower Gini impurity indicates a purer split.

28. **What are the advantages and disadvantages of decision trees?**
    - **Advantages**:
        - Easy to understand and interpret.
        - Can handle both numerical and categorical data.
        - Requires little data preprocessing.
    - **Disadvantages**:
        - Prone to overfitting.
        - Can create biased trees if some classes dominate.

    - Sensitive to noisy data.

29. **How do random forests improve upon decision trees?**
    - **Random Forests**:
        - An ensemble method that combines multiple decision trees.
        - Improves accuracy and robustness by averaging the predictions of many trees.
        - Reduces overfitting by using different subsets of data and features for each tree.

30. **How does a random forest algorithm work?**
    - **Working Principle**:
        - Create multiple decision trees using bootstrapped samples of the training data.
        - Randomly select a subset of features for each tree.
        - Aggregate the predictions of all trees (majority vote for classification, average for regression).

31. **What is bootstrapping in the context of random forests?**
    - **Bootstrapping**:
        - A sampling method that randomly selects data points with replacement to create multiple training subsets.
        - Each decision tree in the random forest is trained on a different bootstrapped sample of the data.

32. **Explain the concept of feature importance in random forests.**
    - **Feature Importance**:
        - Measures the contribution of each feature to the prediction accuracy.
        - Calculated by assessing the decrease in accuracy or Gini impurity when a feature is excluded.
        - Helps in understanding which features are most influential.

33. **What are the key hyperparameters of a random forest and how do they affect the model?**
    - **Key Hyperparameters**:
        - **Number of Trees**: More trees generally improve performance but increase computation.
        - **Max Features**: Number of features to consider for splitting at each node. Controls overfitting.
        - **Max Depth**: Maximum depth of the trees. Limits overfitting by controlling tree size.
        - **Min Samples Split**: Minimum number of samples required to split an internal node. Prevents overfitting by requiring a minimum number of samples.
        - **Min Samples Leaf**: Minimum number of samples required at a leaf node. Ensures leaves have enough samples to make reliable predictions.

34. **Describe the logistic regression model and its assumptions.**
    - **Logistic Regression**:
        - A linear model used for binary classification.
        - Assumes a linear relationship between the input features and the log-odds of the

outcome.
    - Uses the sigmoid function to map predictions to probabilities.
- **Assumptions**:
    - Linearity: Linear relationship between features and the log-odds.
    - Independence: Observations are independent.
    - No multicollinearity: Features are not highly correlated.

35. How does logistic regression handle binary classification problems?

- **Binary Classification**:
    - Uses a linear combination of input features to calculate the log-odds of the positive class.
    - Applies the sigmoid function to convert log-odds to probabilities.
    - Classifies based on a threshold (e.g., 0.5).

36. What is the sigmoid function and how is it used in logistic regression?

- **Sigmoid Function**:
    - σ(z)=1/(1 + e^(-z))
    - Maps any real-valued number to a value between 0 and 1.
    - Used to convert the linear combination of input features into a probability in logistic regression.

37. Explain the concept of the cost function in logistic regression.

- **Cost Function**:
    - Measures the difference between predicted probabilities and actual labels.
    - Common cost function: Log-Loss (Binary Cross-Entropy).
    - Objective: Minimize the cost function to improve model performance.

38. How can logistic regression be extended to handle multiclass classification?

- **Multiclass Classification**:
    - **One-vs-Rest (OvR)**: Train a separate binary classifier for each class, treating the other classes as a single class.
    - **Multinomial Logistic Regression**: Generalizes logistic regression to handle multiple classes by using a softmax function.

39. What is the difference between L1 and L2 regularization in logistic regression?

- **L1 Regularization (Lasso)**:
    - Adds the absolute value of coefficients as a penalty to the cost function.
    - Promotes sparsity, leading to feature selection (some coefficients become zero).
- **L2 Regularization (Ridge)**:
    - Adds the squared value of coefficients as a penalty to the cost function.
    - Shrinks coefficients but does not lead to zero coefficients.
    - Helps prevent overfitting by constraining large weights.

40. **What is XGBoost and how does it differ from other boosting algorithms?**
    - **XGBoost**:
        - An optimized implementation of the Gradient Boosting algorithm.
        - Uses regularization to prevent overfitting.
        - Provides efficient handling of sparse data and missing values.
        - Faster and more accurate due to various algorithmic optimizations.

41. **Explain the concept of boosting in the context of ensemble learning.**
    - **Boosting**:
        - Sequentially trains models, with each new model focusing on correcting errors of the previous ones.
        - Combines weak learners to form a strong learner.
        - Common algorithms: AdaBoost, Gradient Boosting, XGBoost.

42. **How does XGBoost handle missing values?**
    - **Handling Missing Values**:
        - Uses a sparsity-aware algorithm to handle missing data.
        - Automatically learns the best imputation strategy during training.

43. **What are the key hyperparameters in XGBoost and how do they affect model performance?**
    - **Key Hyperparameters**:
        - **Learning Rate**: Controls the contribution of each tree. Lower values prevent overfitting.
        - **n_estimators**: Number of trees to build. More trees can improve performance but increase computation.
        - **max_depth**: Maximum depth of each tree. Controls the complexity of the model.
        - **subsample**: Fraction of samples used for training each tree. Reduces overfitting.
        - **colsample_bytree**: Fraction of features used for each tree. Helps prevent overfitting.
        - **gamma**: Minimum loss reduction required to make a split. Controls tree complexity.

44. **Describe the process of gradient boosting in XGBoost.**
    - **Gradient Boosting Process**:
        - Initialize predictions with a base model (e.g., mean of the target).
        - Compute the residuals (errors) for the current model.
        - Train a new model to predict the residuals.
        - Add the new model's predictions to the existing predictions.
        - Repeat the process for a specified number of iterations.

45. **What are the advantages and disadvantages of using XGBoost?**
    - **Advantages**:
        - High performance and accuracy.
        - Handles missing data efficiently.
        - Built-in regularization reduces overfitting.
        - Parallel processing capabilities for faster training.
    - **Disadvantages**:
        - Computationally intensive and requires more memory.

- Sensitive to hyperparameter tuning.
- Can be complex to interpret and understand.