(ASSIGNMENT: 3)

Name	MOHAMMAD SHEHROZ
Roll No	KNOWFAD9330
Course Name	Flutter App Dev
Batch	4

Q.1: Create a list of names and print all names using list.

```
void main(){

var name=['Ali','Mohsin','Shehbaz'];

for(var i=0;i<name.length;i++) {

print(name[i]);

}

}
```

Q.2: Create an empty list of type string called days. Use the add method to add names of 7 days and print all days.

```
void main(){

  var days=[];

  days.add("Saturday");

  days.add("Sunday");

  days.add("Monday");

  days.add("Tuesday");

  days.add("Wednesday");

  days.add("Thursday");

  days.add("Friday");

  for(var i=0;i<days.length;i++)

  {

    print(days[i]);

  }

}
```

Q.3: Create a list of Days and remove one by one from the end of list.

```
void main(){
  var days=[];
  days.add("Saturday");
  days.add("Sunday");
  days.add("Monday");
  days.add("Tuesday");
  days.add("Wednesday");
  days.add("Thursday");
  days.add("Friday");
  while(!days.isEmpty)
  {
    days.removeLast();
  }
  print(days);
}
```

Q.4: Create a list of numbers & write a program to get the smallest & greatest number from a list.

```
void main(){
  var numbers=[25,29,14,26];
  //maximum number get
  var max=numbers[0];
  for(var i=0;i<numbers.length;i++)
  {
    if(numbers[i]>=max)
    {
      max=numbers[i];
    }
  }
  print("Maximum Value is $max");
  //minimum number get
  var min=numbers[0];
  for(var i=0;i<numbers.length;i++)
  {
    if(numbers[i]<=min)
    {
      min=numbers[i];
    }
  }
  print("Minimum Value is $min");
}
```

Q.5: Create a map with name, phone keys and store some values to it. Use where to find all keys that have length 4.

```
void main() {
  Map<String, String> contacts = {
    'John': '1234567890',
    'Alice': '9876543210',
    'Bob': '5678901234',
    'Emily': '2345678901',
  };

  var keysWithLengthFour = contacts.keys.where((key) => key.length == 4);

  print('Keys with length 4: $keysWithLengthFour');
}
```

Q.6: Create Map variable name world then inside it create countries Map, Key will be the name country & country value will have another map having capitalCity, currency and language to it. by using any country key print all the value of Capital & Currency.

```
void main() {
  Map<String, Map<String, dynamic>> world = {
    'countries': {
      'USA': {
        'capitalCity': 'Washington, D.C.',
        'currency': 'US Dollar',
        'language': 'English',
      },
      'India': {
        'capitalCity': 'New Delhi',
        'currency': 'Indian Rupee',
        'language': 'Hindi',
      },
      'Germany': {
        'capitalCity': 'Berlin',
        'currency': 'Euro',
        'language': 'German',
      },
    },
  };

  String country = 'India';
  String capital = world['countries'][country]['capitalCity'];
  String currency = world['countries'][country]['currency'];

  print('Capital: $capital');
  print('Currency: $currency');
}
```

Q.7:
```
Map<String, double> expenses = {
  'sun': 3000.0,
  'mon': 3000.0,
  'tue': 3234.0,
};
```

Check if "fri" exist in expanses; if exist change it's value to 5000.0 otherwise add 'fri' to expenses and set its value to 5000.0 then print expenses.

```
void main() {
  Map<String, double> expenses = {
    'sun': 3000.0,
    'mon': 3000.0,
    'tue': 3234.0,
  };

  if (expenses.containsKey('fri')) {
    expenses['fri'] = 5000.0;
  } else {
    expenses['fri'] = 5000.0;
  }

  print(expenses);
}
```

Q.8: remove all false values from below list by using removeWhere or retainWhere property.

List<Map<String, bool>> usersEligibility = [
{'name': 'John', 'eligible': true},
{'name': 'Alice', 'eligible': false},
{'name': 'Mike', 'eligible': true},
{'name': 'Sarah', 'eligible': true},
{'name': 'Tom', 'eligible': false},
];

```
void main() {
  List<Map<String, bool>> usersEligibility = [
    {'name': 'John', 'eligible': true},
    {'name': 'Alice', 'eligible': false},
    {'name': 'Mike', 'eligible': true},
    {'name': 'Sarah', 'eligible': true},
    {'name': 'Tom', 'eligible': false},
  ];


  usersEligibility.removeWhere((user) => user['eligible'] == false);


  print(usersEligibility);
}
```

Q.9: Given a list of integers, write a dart code that returns the maximum value from the list.

```
void main() {
  List<int> numbers = [5, 2, 9, 1, 7, 3];
  int maxValue = numbers.reduce((value, element) => value > element ? value : element);
  print('Maximum value: $maxValue');
}
```

Q.10: Write a Dart code that takes in a list of strings and removes any duplicate elements, returning a new list without duplicates. The order of elements in the new list should be the same as in the original list.

```
void main() {
  List<String> strings = ['apple', 'banana', 'orange', 'banana', 'grape', 'apple'];
  List<String> uniqueStrings = strings.toSet().toList();
  print('Original list: $strings');
  print('List without duplicates: $uniqueStrings');
}
```

Q 11: Write a Dart code that takes in a list and an integer n as parameters. The program should print a new list containing the first n elements from the original list.

```
void main() {
  List<int> originalList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
  int n = 5;

  List<int> newList = originalList.take(n).toList();
  print(newList);
}
```

Q.12: Write a Dart code that takes in a list of strings and prints a new list with the elements in reverse order. The original list should remain unchanged.

```
void main() {
  List<String> originalList = ['apple', 'banana', 'orange', 'grape'];

  List<String> reversedList = originalList.reversed.toList();

  print('Original list: $originalList');
  print('Reversed list: $reversedList');
}
```

Q.13: Implement a code that takes in a list of integers and returns a new list containing only the unique elements from the original list. The order of elements in the new list should be the same as in the original list.

```
List<int> getUniqueElements(List<int> inputList) {
  List<int> uniqueList = [];
  inputList.forEach((element) {
    if (!uniqueList.contains(element)) {
      uniqueList.add(element);
    }
  });
  return uniqueList;
}


void main() {
  List<int> originalList = [1, 2, 3, 2, 4, 5, 1, 3, 6, 7, 7, 6];
  List<int> uniqueList = getUniqueElements(originalList);
  print('Original list: $originalList');
  print('Unique list: $uniqueList');
}
```

Q.14: Write a Dart code that takes in a list of integers and prints a new list with the elements sorted in ascending order. The original list should remain unchanged.

```
void main() {
  List<int> originalList = [5, 2, 9, 1, 7, 3];


  List<int> sortedList = List.from(originalList);
  sortedList.sort();
  print('Original list: $originalList');
  print('Sorted list: $sortedList');
}
```

Q.15: Implement a Dart code that uses the where() method to filter out negative numbers from a list of integers. The program should take in the original list as a parameter and print a new list containing only the positive numbers.

```
void main() {

  List<int> originalList = [1, -2, 3, -4, 5, -6, 7, -8];

  List<int> positiveNumbers = originalList.where((number) => number > 0).toList();

  print('Original list: $originalList');

  print('Positive numbers: $positiveNumbers');

}
```

Q.16: Implement a Dart code that uses the where() method to filter out odd numbers from a list of integers. The program should take in the original list as a parameter and print a new list containing only the even numbers.

```
void main() {

  List<int> originalList = [1, 2, 3, 4, 5, 6, 7, 8, 9];

  List<int> evenNumbers = originalList.where((number) => number % 2 == 0).toList();

  print('Original list: $originalList');

  print('Even numbers: $evenNumbers');

}
```

Q.17: Given a list of integers, write a Dart code that uses the map() method to create a new list with each value squared. The program should take in the original list as a parameter and print the new list.

```
void main() {

  List<int> originalList = [1, 2, 3, 4, 5];

  List<int> squaredList = originalList.map((number) => number * number).toList();

  print('Original list: $originalList');

  print('Squared list: $squaredList');

}
```

Q.18: Create a map named "person" with the following key-value pairs: "name" as "John", "age" as 25, "isStudent" as true. Write a Dart code to check if the person is both a student and over 18 years old. Print "Eligible" if both conditions are true, otherwise print "Not eligible".

```
void main() {
  Map<String, dynamic> person = {
    'name': 'John',
    'age': 25,
    'isStudent': true,
  };
  bool isStudent = person['isStudent'];
  int age = person['age'];
  if (isStudent && age > 18) {
    print('Eligible');
  } else {
    print('Not eligible');
  }
}
```

Q.19: Given a map representing a product with keys "name", "price", and "quantity", write Dart code to check if the product is in stock. If the quantity is greater than 0, print "In stock", otherwise print "Out of stock".

```
void main() {
  Map<String, dynamic> product = {
    'name': 'Example Product',
    'price': 9.99,
    'quantity': 5,
  };

  int quantity = product['quantity'];

  if (quantity > 0) {
    print('In stock');
  } else {
    print('Out of stock');
  }
}
```

Q.20: Create a map named "car" with the following key-value pairs: "brand" as "Toyota", "color" as "Red", "isSedan" as true. Write Dart code to check if the car is a sedan and red in color. Print "Match" if both conditions are true, otherwise print "No match".

```
void main() {
  Map<String, dynamic> car = {
    'brand': 'Toyota',
    'color': 'Red',
    'isSedan': true,
  };

  String color = car['color'];
  bool isSedan = car['isSedan'];

  if (color == 'Red' && isSedan) {
    print('Match');
  } else {
    print('No match');
  }
}
```

Q.21: Given a map representing a user with keys "name", "isAdmin", and "isActive", write Dart code to check if the user is an active admin. If the user is both an admin and active, print "Active admin", otherwise print "Not an active admin".

```
void main() {
  Map<String, dynamic> user = {
    'name': 'John Doe',
    'isAdmin': true,
    'isActive': true,
  };

  bool isAdmin = user['isAdmin'];
  bool isActive = user['isActive'];

  if (isAdmin && isActive) {
    print('Active admin');
  } else {
    print('Not an active admin');
  }
}
```

Q.22: Given a map representing a shopping cart with keys as product names and values as quantities, write Dart code to check if a product named "Apple" exists in the cart. Print "Product found" if it exists, otherwise print "Product not found".

```
void main() {
  Map<String, int> shoppingCart = {
    'Banana': 2,
    'Orange': 3,
    'Apple': 1,
    'Grapes': 4,
  };
  String productName = 'Apple';
  if (shoppingCart.containsKey(productName)) {
    print('Product found');
  } else {
    print('Product not found');
  }
}
```