# Assignment # 02

**Submitted by: Hanzala Bin Rashid**

**Roll Number: 140515**

**Submitted by: Sir Bilal**

**Due Date: 24-06-2023**

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

Source Code:

```
void main() {
  List<String> names = [
    'Ahmed',
    'Saif',
    'Tamim',
    'Okasha',
    'Hafsa',
  ];

  names.forEach((name) {
    print(name);
  });
}
```

Output:

```
flutter: Ahmed
flutter: Saif
flutter: Tamim
flutter: Okasha
flutter: Hafsa
```

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.

Source Code:

```
void main() {
  List<String> days = [];

  days.add('Monday');
  days.add('Tuesday');
```

```
  days.add('Wednesday');
  days.add('Thursday');
  days.add('Friday');
  days.add('Saturday');
  days.add('Sunday');

  days.forEach((day) {
    print(day);
  });
}
```

Output:

```
 flutter: Monday
 flutter: Tuesday
 flutter: Wednesday
 flutter: Thursday
 flutter: Friday
 flutter: Saturday
 flutter: Sunday
```

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

Source Code:

```
void main() {
  List<String> daysOfWeek = [
    'Sunday',
    'Monday',
    'Tuesday',
    'Wednesday',
    'Thursday',
    'Friday',
    'Saturday'
  ];

  // Removing one day at a time from the end of the list
  while (daysOfWeek.isNotEmpty) {
    print('Current list: $daysOfWeek');
    String removedDay = daysOfWeek.removeLast();
    print('Removed day: $removedDay');
  }
}
```

Output:

```
flutter: Current list: [Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday]
flutter: Removed day: Saturday
flutter: Current list: [Sunday, Monday, Tuesday, Wednesday, Thursday, Friday]
flutter: Removed day: Friday
flutter: Current list: [Sunday, Monday, Tuesday, Wednesday, Thursday]
flutter: Removed day: Thursday
flutter: Current list: [Sunday, Monday, Tuesday, Wednesday]
flutter: Removed day: Wednesday
flutter: Current list: [Sunday, Monday, Tuesday]
flutter: Removed day: Tuesday
flutter: Current list: [Sunday, Monday]
flutter: Removed day: Monday
flutter: Current list: [Sunday]
flutter: Removed day: Sunday
```

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

Source Code:

```dart
void main() {
  List<int> numbers = [15, 7, 22, 4, 9, 13, 11, 5];

  int smallestNumber = getSmallestNumber(numbers);
  int greatestNumber = getGreatestNumber(numbers);

  print('List of numbers: $numbers');
  print('Smallest number: $smallestNumber');
  print('Greatest number: $greatestNumber');
}

int getSmallestNumber(List<int> numbers) {
  int smallest = numbers[0];

  for (int i = 1; i < numbers.length; i++) {
    if (numbers[i] < smallest) {
      smallest = numbers[i];
    }
  }

  return smallest;
}

int getGreatestNumber(List<int> numbers) {
  int greatest = numbers[0];

  for (int i = 1; i < numbers.length; i++) {
    if (numbers[i] > greatest) {
      greatest = numbers[i];
```

```
  }
 }

 return greatest;
}
```

Output:

```
flutter: List of numbers: [15, 7, 22, 4, 9, 13, 11, 5]
flutter: Smallest number: 4
flutter: Greatest number: 22
```

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.

Source Code:

```
void main() {
  // Create a map
  Map<String, String> contacts = {
    'John': '1234567890',
    'Jane': '9876543210',
    'Alex': '555555',
    'Sarah': '1234',
    'Mike': '5678',
    'Emily': '123456789',
  };

  // Use 'where' to filter keys with length 4
  List<String> keysWithLength4 = contacts.keys.where((key) => key.length ==
4).toList();

  // Print the keys with length 4
  print('Keys with length 4: $keysWithLength4');
}
```

Output:

```
flutter: Keys with length 4: [John, Jane, Alex, Mike]
```

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.

Source Code:

```dart
void main() {
  // Create the world map
  Map<String, Map<String, dynamic>> world = {
    'Pakistan': {
      'capitalCity': 'Islamabad',
      'currency': 'Pakistani Rupee',
      'language': 'Urdu',
    },
    'United States': {
      'capitalCity': 'Washington, D.C.',
      'currency': 'United States Dollar',
      'language': 'English',
    },
    'France': {
      'capitalCity': 'Paris',
      'currency': 'Euro',
      'language': 'French',
    },
    // Add more countries here...
  };

  // Retrieve and print information for a specific country
  String countryKey = 'Pakistan';
  if (world.containsKey(countryKey)) {
    Map<String, dynamic>? countryInfo = world[countryKey];
    String capitalCity = countryInfo!['capitalCity'];
    String currency = countryInfo!['currency'];
    String language = countryInfo!['language'];

    print('Country: $countryKey');
    print('Capital City: $capitalCity');
    print('Currency: $currency');
    print('Language: $language');
  } else {
    print('Country not found.');
  }
}
```

Output:

```
flutter: Country: Pakistan
flutter: Capital City: Islamabad
flutter: Currency: Pakistani Rupee
flutter: Language: Urdu
```

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.

Source Code:

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

  String key = 'fri';
  double newValue = 5000.0;

  if (expenses.containsKey(key)) {
    expenses[key] = newValue;
  } else {
    expenses[key] = newValue;
  }

  print('Expenses: $expenses');
}
```

Output:

```
flutter: Expenses: {sun: 3000.0, mon: 3000.0, tue: 3234.0, fri: 5000.0}
```

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},

];

Source Code:

```dart
void main() {
  List<Map> 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('Updated usersEligibility: $usersEligibility');
}
```

Output:

```
flutter: Updated usersEligibility: [{name: John, eligible: true}, {name: Mike, eligible: true}, {name: Sarah, eligible: true}]
```

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

Source Code:

```dart
void main() {
  List<int> numbers = [5, 10, 3, 8, 2, 7];
  int maximum = findMaximum(numbers);
  print('Maximum value: $maximum');
}
```

```
int findMaximum(List<int> numbers) {
  if (numbers.isEmpty) {
    throw Exception('The list is empty.');
  }

  int maximum = numbers[0];

  for (int i = 1; i < numbers.length; i++) {
    if (numbers[i] > maximum) {
      maximum = numbers[i];
    }
  }

  return maximum;
}
```

Output:

```
flutter: Maximum value: 10
```

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.

Source Code:

```
void main() {
  List<String> originalList = ['apple', 'banana', 'orange', 'banana', 'kiwi',
'apple'];
  List<String> newList = removeDuplicates(originalList);
  print('Original List: $originalList');
  print('List without Duplicates: $newList');
}

List<String> removeDuplicates(List<String> originalList) {
  List<String> newList = [];
  Set<String> uniqueSet = {};

  for (String element in originalList) {
    if (!uniqueSet.contains(element)) {
      uniqueSet.add(element);
      newList.add(element);
    }
  }
```

```
  return newList;
}
```

Output:

```
flutter: Original List: [apple, banana, orange, banana, kiwi, apple]
flutter: List without Duplicates: [apple, banana, orange, kiwi]
```

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

Source Code:

```
void main() {
  List<int> originalList = [1, 2, 3, 4, 5, 6];
  int n = 3;
  List<int> newList = getFirstNElements(originalList, n);
  print('Original List: $originalList');
  print('New List with First $n Elements: $newList');
}

List<T> getFirstNElements<T>(List<T> originalList, int n) {
  if (n < 0 || n > originalList.length) {
    throw Exception('Invalid value of n.');
  }

  return originalList.sublist(0, n);
}
```

Output:

```
flutter: Original List: [1, 2, 3, 4, 5, 6]
flutter: New List with First 3 Elements: [1, 2, 3]
```

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

Source Code:

```
void main() {
  List<String> originalList = ['apple', 'banana', 'orange', 'kiwi'];
  List<String> reversedList = reverseList(originalList);
  print('Original List: $originalList');
```

```
  print('Reversed List: $reversedList');
}

List<String> reverseList(List<String> originalList) {
  List<String> reversedList = List.from(originalList.reversed);
  return reversedList;
}
```

Output:

```
flutter: Original List: [apple, banana, orange, kiwi]
flutter: Reversed List: [kiwi, orange, banana, apple]
```

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.

Source Code:

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

List<int> getUniqueElements(List<int> originalList) {
  List<int> uniqueList = [];
  Set<int> uniqueSet = {};

  for (int element in originalList) {
    if (!uniqueSet.contains(element)) {
      uniqueSet.add(element);
      uniqueList.add(element);
    }
  }

  return uniqueList;
}
```

Output:

```
flutter: Original List: [1, 2, 3, 2, 4, 5, 1, 3]
flutter: Unique List: [1, 2, 3, 4, 5]
```

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

Source Code:

```
void main() {
  List<int> originalList = [5, 2, 8, 3, 1, 9];
  List<int> sortedList = sortList(originalList);
  print('Original List: $originalList');
  print('Sorted List: $sortedList');
}

List<int> sortList(List<int> originalList) {
  List<int> sortedList = List.from(originalList); // Create a copy of the
original list
  sortedList.sort(); // Sort the copied list in ascending order
  return sortedList;
}
```

Output:

```
flutter: Original List: [5, 2, 8, 3, 1, 9]
flutter: Sorted List: [1, 2, 3, 5, 8, 9]
```

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

Source Code:

```
void main() {
  List<int> originalList = [5, -2, 8, -3, 1, -9];
  List<int> positiveNumbers = getPositiveNumbers(originalList);
  print('Original List: $originalList');
  print('Positive Numbers: $positiveNumbers');
}

List<int> getPositiveNumbers(List<int> originalList) {
  List<int> positiveNumbers = originalList.where((number) => number >
0).toList();
  return positiveNumbers;
}
```

Output:

```
flutter: Original List: [5, -2, 8, -3, 1, -9]
flutter: Positive Numbers: [5, 8, 1]
```

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

Source Code:

```
void main() {
  List<int> originalList = [5, 2, 8, 3, 1, 9];
  List<int> evenNumbers = getEvenNumbers(originalList);
  print('Original List: $originalList');
  print('Even Numbers: $evenNumbers');
}

List<int> getEvenNumbers(List<int> originalList) {
  List<int> evenNumbers = originalList.where((number) => number % 2 ==
0).toList();
  return evenNumbers;
}
```

Output:

```
flutter: Original List: [5, 2, 8, 3, 1, 9]
flutter: Even Numbers: [2, 8]
```

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

Source Code:

```
void main() {
  List<int> originalList = [1, 2, 3, 4, 5];
  List<int> squaredList = squareValues(originalList);
  print('Original List: $originalList');
  print('Squared List: $squaredList');
}

List<int> squareValues(List<int> originalList) {
  List<int> squaredList = originalList.map((number) => number * number).toList();
  return squaredList;
```

```
}
```

Output:

```
flutter: Original List: [1, 2, 3, 4, 5]
flutter: Squared List: [1, 4, 9, 16, 25]
```

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".

Source Code:

```dart
void main() {
  Map<String, dynamic> person = {
    'name': 'John',
    'age': 25,
    'isStudent': true,
  };

  bool isEligible = person['isStudent'] && person['age'] > 18;

  if (isEligible) {
    print('Eligible');
  } else {
    print('Not eligible');
  }
}
```

Output:

```
flutter: 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".

Source Code:

```dart
void main() {
  Map<String, dynamic> product = {
    'name': 'iPhone',
    'price': 999.99,
```

```
    'quantity': 5,
  };

  int quantity = product['quantity'];

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

Output:

```
flutter: In 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".

Source Code:

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

  bool isMatch = car['isSedan'] && car['color'] == 'Red';

  if (isMatch) {
    print('Match');
  } else {
    print('No match');
  }
}
```

Output:

```
flutter: 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".

Source Code:

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

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

  if (isActiveAdmin) {
    print('Active admin');
  } else {
    print('Not an active admin');
  }
}
```

Output:

```
flutter: 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".

Source Code:

```dart
void main() {
  Map<String, int> shoppingCart = {
    'Banana': 2,
    'Orange': 3,
    'Apple': 5,
    'Mango': 1,
  };

  bool productFound = shoppingCart.containsKey('Apple');

  if (productFound) {
    print('Product found');
  } else {
```

```
    print('Product not found');
  }
}
```

Output:

```
flutter: Product found
```