1. Q-1.Solution.

void main() {

  List<String> names = ["Hamza", "Ali", "Usman", "Dowood", "Azhar"];

    for (String name in names) {

    print(name);

  }

}

**DEBUG CONSOLE**

Hamza

Ali

Usman

Dowood

Azhar

1. Q-2. Solution.

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");

  for (String day in days) {

    print(day);

  }

}

**DEBUG CONSOLE**

Monday

Tuesday

Wednesday

Thursday

Friday

Saturday

Sunday

1. Q-3. Solution.

void main() {

  List<String> days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"];

  while (days.isNotEmpty) {

    String removedDay = days.removeLast();

    print("Removed Day: $removedDay");

    print("Remaining Days: $days");

  }

}

**DEBUG CONSOLE**

Removed Day: Sunday

Remaining Days: [Monday, Tuesday, Wednesday, Thursday, Friday, Saturday]

Removed Day: Saturday

Remaining Days: [Monday, Tuesday, Wednesday, Thursday, Friday]

Removed Day: Friday

Remaining Days: [Monday, Tuesday, Wednesday, Thursday]

Removed Day: Thursday

Remaining Days: [Monday, Tuesday, Wednesday]

Removed Day: Wednesday

Remaining Days: [Monday, Tuesday]

Removed Day: Tuesday

Remaining Days: [Monday]

Removed Day: Monday

Remaining Days: []

1. Q-4. SOLUTION.

void main() {

  List<int> numbers = [5, 12, 8, 3, 15, 7, 10];

  int smallest = findSmallestNumber(numbers);

  int greatest = findGreatestNumber(numbers);

  print("List of Numbers: $numbers");

  print("Smallest Number: $smallest");

  print("Greatest Number: $greatest");

}

int findSmallestNumber(List<int> numbers) {

  int smallest = numbers[0];

  for (int number in numbers) {

    if (number < smallest) {

      smallest = number;

    }

  }

  return smallest;

}

int findGreatestNumber(List<int> numbers) {

  int greatest = numbers[0];

  for (int number in numbers) {

    if (number > greatest) {

      greatest = number;

    }

  }

  return greatest;

}

**DEBUG CONSOLE**

List of Numbers: [5, 12, 8, 3, 15, 7, 10]

Smallest Number: 3

Greatest Number: 15

1. Q-5. SOLUTION.

void main() {

  Map<String, String> contactMap = {

    "Asif": "1234567890",

    "Ali":"123",

    "Amjad":"0633654",

    "Azhar":"0214587",

  };

  List<String> keysWithLength4 = contactMap.keys.where((key) => key.length == 4).toList();

  print("Original Map: $contactMap");

  print("Keys with Length 4: $keysWithLength4");

}

**DEBUG CONSOLE**

Original Map: {Asif: 1234567890, Ali: 123, Amjad: 0633654, Azhar: 0214587}

Keys with Length 4: [Asif]

1. Q-6. SOLUTIONS.
2. Q-7. SOLUTION.

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("Updated Expenses: $expenses");

}

**DEBUG CONSOLE**

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

1. Q-8. SOLUTION.
2. Q-9. SOLUTION.

void main() {

  List<int> numbers = [10, 5, 8, 15, 3, 20, 12];

  int maxValue = findMaxValue(numbers);

  print("Maximum Value: $maxValue");

}

int findMaxValue(List<int> numbers) {

  int maxValue = numbers[0];

  for (int number in numbers) {

    if (number > maxValue) {

      maxValue = number;

    }

  }

  return maxValue;

}

**DEBUG CONSOLE**

Maximum Value: 20

1. Q-10. SOLUTION.

void main() {

  List<String> originalList = ["apple", "orange", "banana", "apple", "grape", "banana"];

  List<String> newListWithoutDuplicates = removeDuplicates(originalList);

  print("Original List: $originalList");

  print("List Without Duplicates: $newListWithoutDuplicates");

}

List<String> removeDuplicates(List<String> inputList) {

  Set<String> uniqueSet = Set<String>();

  List<String> result = [];

  for (String element in inputList) {

    if (uniqueSet.add(element)) {

      result.add(element);

    }

  }

  return result;

}

**DEBUG CONSOLE**

Original List: [apple, orange, banana, apple, grape, banana]

List Without Duplicates: [apple, orange, banana, grape]

1. Q-11. SOLUTIONS.

void main() {

  List<int> originalList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

  int n = 5;

  List<int> newList = getFirstNElements(originalList, n);

  print("Original List: $originalList");

  print("New List with the First $n Elements: $newList");

}

List<int> getFirstNElements(List<int> inputList, int n) {

  return inputList.sublist(0, n);

}

**DEBUG CONSOLE**

Original List: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] New List with the First 5 Elements: [1, 2, 3, 4, 5]

1. Q-12. SOLUTIONS.

void main() {

  List<String> originalList = ["apple", "orange", "banana", "grape", "kiwi"];

  List<String> reversedList = getReversedElements(originalList);

  print("Original List: $originalList");

  print("Reversed List: $reversedList");

}

List<String> getReversedElements(List<String> inputList) {

  List<String> reversedList = List.from(inputList.reversed);

  return reversedList;

}

**DEBUG CONSOLE**

Original List: [apple, orange, banana, grape, kiwi]

Reversed List: [kiwi, grape, banana, orange, apple]

1. Q-13. SOLUTIONS.

void main() {

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

  List<int> uniqueList = getUniqueElements(originalList);

  print("Original List: $originalList");

  print("Unique List: $uniqueList");

}

List<int> getUniqueElements(List<int> inputList) {

  Set<int> uniqueSet = Set<int>();

  List<int> uniqueList = [];

  for (int element in inputList) {

    if (uniqueSet.add(element)) {

      uniqueList.add(element);

    }

  }

  return uniqueList;

}

**DEBUG CONSOLE**

Original List: [3, 1, 2, 4, 3, 2, 5, 6, 4, 7, 8, 9, 5]

Unique List: [3, 1, 2, 4, 5, 6, 7, 8, 9]

1. Q-14. SOLUTIONS.

void main() {

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

  List<int> sortedList = getSortedElements(originalList);

  print("Original List: $originalList");

  print("Sorted List (Ascending): $sortedList");

}

List<int> getSortedElements(List<int> inputList) {

  List<int> sortedList = List.from(inputList)..sort();

  return sortedList;

}

**DEBUG CONSOLE**

Original List: [5, 2, 8, 1, 7, 3, 6, 4]

Sorted List (Ascending): [1, 2, 3, 4, 5, 6, 7, 8]

1. Q-15. SOLUTIONS.

void main() {

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

  List<int> positiveNumbersList = filterPositiveNumbers(originalList);

  print("Original List: $originalList");

  print("Positive Numbers List: $positiveNumbersList");

}

List<int> filterPositiveNumbers(List<int> inputList) {

  List<int> positiveNumbersList = inputList.where((number) => number > 0).toList();

  return positiveNumbersList;

}

**DEBUG CONSOLE**

Original List: [3, -1, 2, -4, 5, -6, 4, 7, -8, 9, -5]

Positive Numbers List: [3, 2, 5, 4, 7, 9]

1. Q-16. SOLUTIONS.

void main() {

  List<int> originalList = [3, 6, 9, 12, 5, 8, 11, 14, 7, 10];

  List<int> evenNumbersList = filterEvenNumbers(originalList);

  print("Original List: $originalList");

  print("Even Numbers List: $evenNumbersList");

}

List<int> filterEvenNumbers(List<int> inputList) {

  List<int> evenNumbersList = inputList.where((number) => number % 2 == 0).toList();

  return evenNumbersList;

}

**DEBUG CONSOLE**

Original List: [3, 6, 9, 12, 5, 8, 11, 14, 7, 10]

Even Numbers List: [6, 12, 8, 14, 10]

1. Q-17. SOLUTIONS.

void main() {

  List<int> originalList = [2, 4, 6, 8, 10];

  List<int> squaredList = squareValues(originalList);

  print("Original List: $originalList");

  print("Squared List: $squaredList");

}

List<int> squareValues(List<int> inputList) {

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

  return squaredList;

}

**DEBUG CONSOLE**

Original List: [2, 4, 6, 8, 10]

Squared List: [4, 16, 36, 64, 100]

1. Q-18. SOLUTIONS.

void main() {

  Map<String, dynamic> person = {

    "name": "John",

    "age": 25,

    "isStudent": true,

  };

    if (isEligible(person)) {

    print("Eligible");

  } else {

    print("Not eligible");

  }

}

bool isEligible(Map<String, dynamic> person) {

  return person["isStudent"] == true && person["age"] > 18;

}

**DEBUG CONSOLE**

Eligible

1. Q-19. SOLUTIONS.

void main() {

  Map<String, dynamic> product = {

    "name": "Widget",

    "price": 19.99,

    "quantity": 5,

  };

  checkStock(product);

}

void checkStock(Map<String, dynamic> product) {

  if (product["quantity"] > 0) {

    print("In stock");

  } else {

    print("Out of stock");

  }

}

**DEBUG CONSOLE**

In stock

1. Q-20. SOLUTIONS.

void main() {

  Map<String, dynamic> car = {

    "brand": "Toyota",

    "color": "Red",

    "isSedan": true,

  };

  checkCarMatch(car);

}

void checkCarMatch(Map<String, dynamic> car) {

  if (car["isSedan"] == true && car["color"] == "Red") {

    print("Match");

  } else {

    print("No match");

  }

}

**DEBUG CONSOLE**

Match

1. Q-21. SOLUTIONS.

void main() {

  Map<String, dynamic> user = {

    "name": "John",

    "isAdmin": true,

    "isActive": true,

  };

 checkActiveAdmin(user);

}

void checkActiveAdmin(Map<String, dynamic> user) {

 if (user["isAdmin"] == true && user["isActive"] == true) {

    print("Active admin");

  } else {

    print("Not an active admin");

  }

}

**DEBUG CONSOLE**

Active admin

1. Q-22. SOLUTIONS.

void main() {

  Map<String, int> shoppingCart = {

    "Apple": 5,

    "Banana": 3,

    "Orange": 2,

  };

  checkProductExistence(shoppingCart);

}

void checkProductExistence(Map<String, int> cart) {

  if (cart.containsKey("Apple")) {

    print("Product found");

  } else {

    print("Product not found");

  }

}

**DEBUG CONSOLE**

Product found