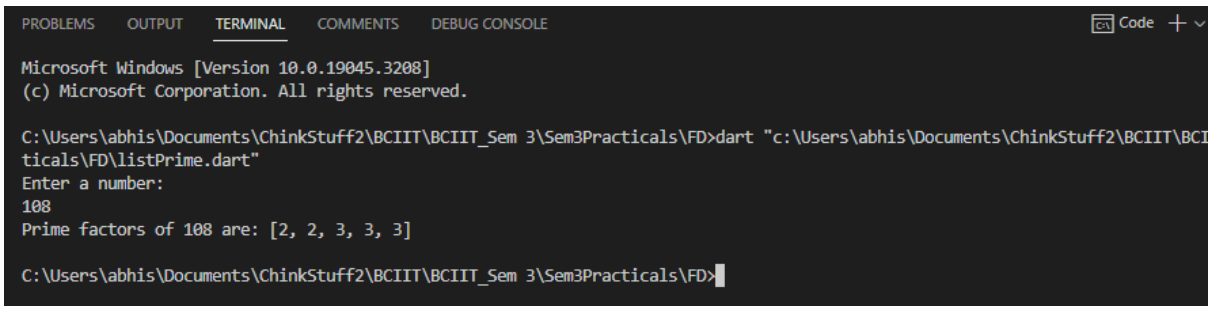
Q1)

```
import 'dart:io';

List<int> findFactors(int number) {
  List<int> factors = [];

  for (int i = 1; i <= number; i++) {
    if (number % i == 0) {
      factors.add(i);
    }
  }

  return factors;
}

void main() {
  print("Enter a number:");
  int number = int.parse(stdin.readLineSync() ?? '');

  List<int> factors = findFactors(number);

  print("Factors of $number are:");
  for (int factor in factors) {
    print(factor);
  }
}
```

```
C:\Users\abhis\Documents\ChinkStuff2\BCIIT\BCIIT_Sem 3\Sem3Practicals\FD>dart "c:\Users\abhis\Docum
ticals\FD\listFactors.dart"
Enter a number:
10
Factors of 10 are:
1
2
5
10

C:\Users\abhis\Documents\ChinkStuff2\BCIIT\BCIIT_Sem 3\Sem3Practicals\FD>
```

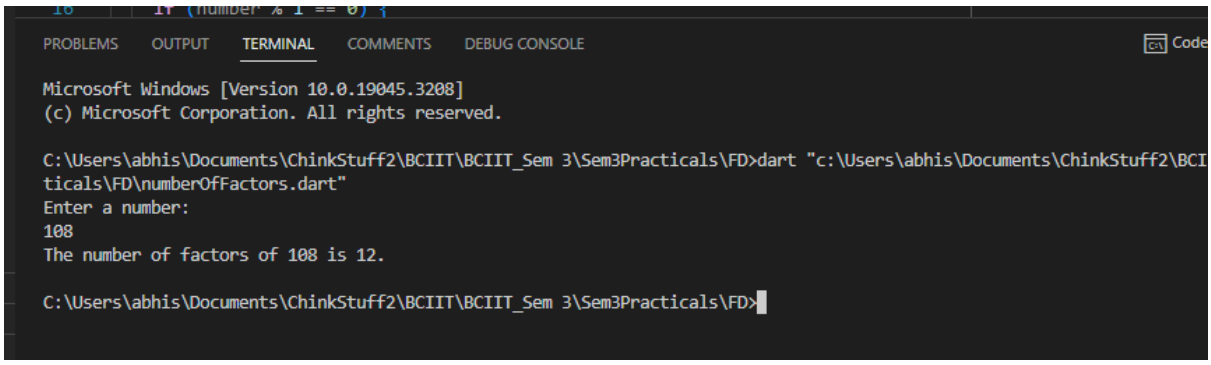
Q2)

```
bool checkNumberInPairs(List<int> numbers, int target) {
  for (int i = 0; i < numbers.length - 1; i++) {
    if (numbers[i] != target && numbers[i + 1] != target) {
      return false;
    }
  }
  return true;
}

void main() {
  List<int> numbers = [1, 2, 2, 3, 3, 2, 2, 1];
  int target = 2;

  bool result = checkNumberInPairs(numbers, target);

  if (result) {
    print("$target appears in every pair of adjacent integers.");
  } else {
    print("$target does not appear in every pair of adjacent integers.");
  }
}
```



Q3)

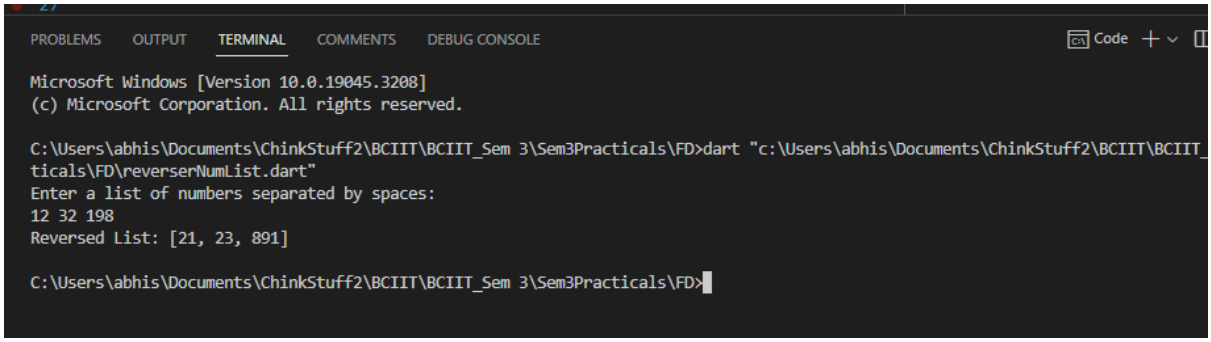
```
import 'dart:io';

void main() {
  print("Enter a number:");
  int number = int.parse(stdin.readLineSync() ?? '');

  int numberOfFactors = countFactors(number);

  print("The number of factors of $number is $numberOfFactors.");
}

int countFactors(int number) {
  int count = 0;

  for (int i = 1; i <= number; i++) {
    if (number % i == 0) {
      count++;
    }
  }

  return count;
}
```



Q4)

```
import 'dart:io';

void main() {
  print("Enter a list of numbers separated by spaces:");
  String input = stdin.readLineSync() ?? '';

  List<int> numbers = input
      .split(' ')
      .map((str) => int.tryParse(str) ?? 0) // Parse input to integers
      .toList();

  List<int> reversedNumbers = reverseDigitsInList(numbers);

  print("Reversed List: $reversedNumbers");
}

List<int> reverseDigitsInList(List<int> list) {
  List<int> reversedList = [];

  for (int number in list) {
    int reversedNumber = reverseDigits(number);
    reversedList.add(reversedNumber);
  }

  return reversedList;
}

int reverseDigits(int number) {
```

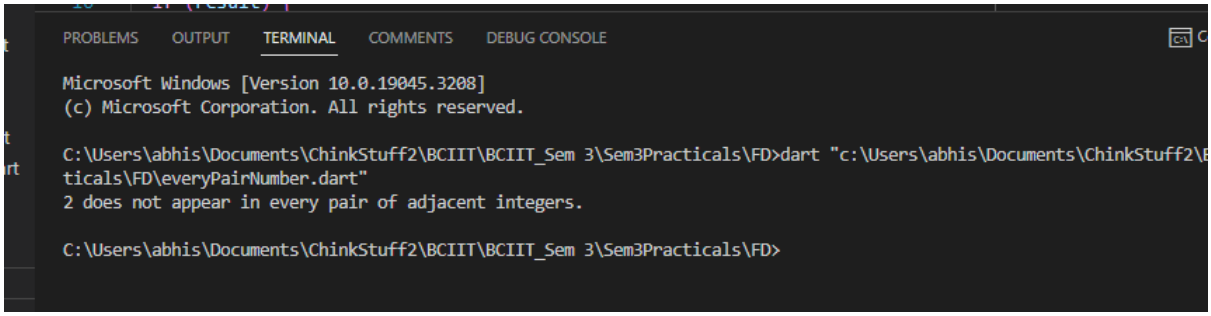
```
  int reversed = 0;

  while (number != 0) {
    int digit = number % 10;
    reversed = reversed * 10 + digit;
    number ~/= 10;
  }

  return reversed;
}
```



Q5)

```
bool checkNumberInPairs(List<int> numbers, int target) {
  for (int i = 0; i < numbers.length - 1; i++) {
    if (numbers[i] != target && numbers[i + 1] != target) {
      return false;
    }
  }
  return true;
}

void main() {
  List<int> numbers = [1, 2, 2, 3, 3, 2, 2, 1];
  int target = 2;
```

```
  bool result = checkNumberInPairs(numbers, target);

  if (result) {
    print("$target appears in every pair of adjacent integers.");
  } else {
    print("$target does not appear in every pair of adjacent integers.");
  }
}
```

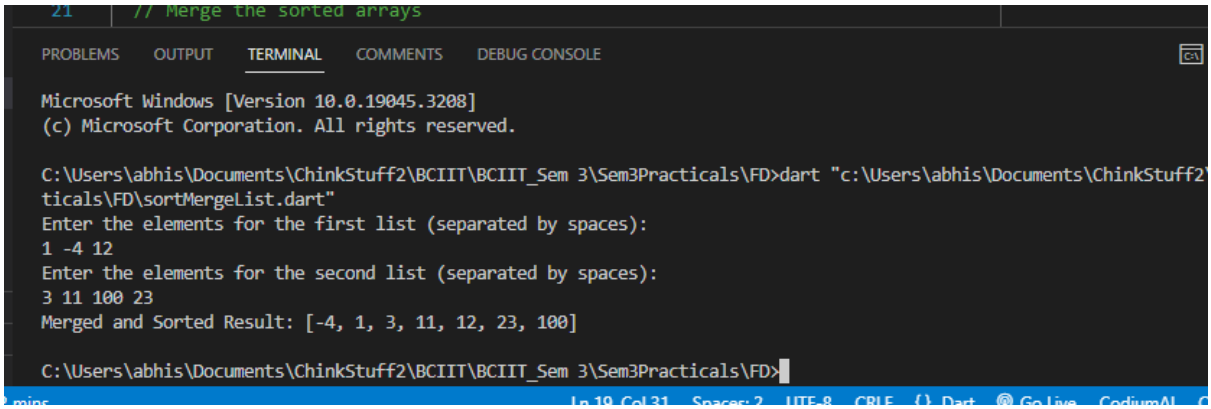


Q6)

```
import 'dart:io';

void main() {
  List<int> array1 = [];
  List<int> array2 = [];

  // Input for the first list
  print("Enter the elements for the first list (separated by spaces):");
  String input1 = stdin.readLineSync() ?? '';
  array1 = input1.split(' ').map((str) => int.tryParse(str) ?? 0).toList();

  // Input for the second list
  print("Enter the elements for the second list (separated by spaces):");
```

```
  String input2 = stdin.readLineSync() ?? '';
  array2 = input2.split(' ').map((str) => int.tryParse(str) ?? 0).toList();

  // Manually sort both arrays
  array1 = manualSort(array1);
  array2 = manualSort(array2);

  // Merge the sorted arrays
  List<int> result = mergeSortedArrays(array1, array2);

  print("Merged and Sorted Result: $result");
}

List<int> manualSort(List<int> list) {
  for (int i = 0; i < list.length - 1; i++) {
    for (int j = i + 1; j < list.length; j++) {
      if (list[i] > list[j]) {
        int temp = list[i];
        list[i] = list[j];
        list[j] = temp;
      }
    }
  }
  return list;
}

List<int> mergeSortedArrays(List<int> array1, List<int> array2) {
  List<int> result = [];

  int i = 0;
  int j = 0;
```

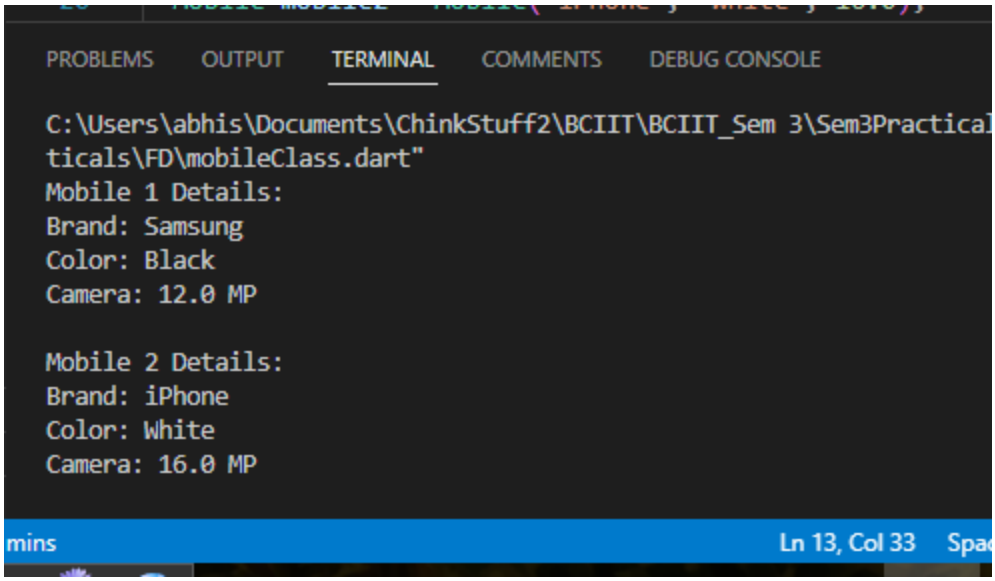
```
  while (i < array1.length && j < array2.length) {
    if (array1[i] < array2[j]) {
      result.add(array1[i]);
      i++;
    } else {
      result.add(array2[j]);
      j++;
    }
  }

  // Add any remaining elements from both arrays
  while (i < array1.length) {
    result.add(array1[i]);
    i++;
  }

  while (j < array2.length) {
    result.add(array2[j]);
    j++;
  }

  return result;
}
```



Q7)

```
class Mobile {
  String brand;
  String color;
  double camera;

  // Constructor to initialize the mobile object
  Mobile(this.brand, this.color, this.camera);

  // Method to print mobile details
  void printDetails() {
    print("Brand: $brand");
    print("Color: $color");
    print("Camera: $camera MP");
  }
}

void main() {
  // Initialize three mobile objects
  Mobile mobile1 = Mobile("Samsung", "Black", 12.0);
  Mobile mobile2 = Mobile("iPhone", "White", 16.0);
  Mobile mobile3 = Mobile("Google Pixel", "Silver", 12.0);

  // Print details of the mobile objects
  print("Mobile 1 Details:");
  mobile1.printDetails();

  print("\nMobile 2 Details:");
  mobile2.printDetails();
```

```
  print("\nMobile 3 Details:");
  mobile3.printDetails();
}
```

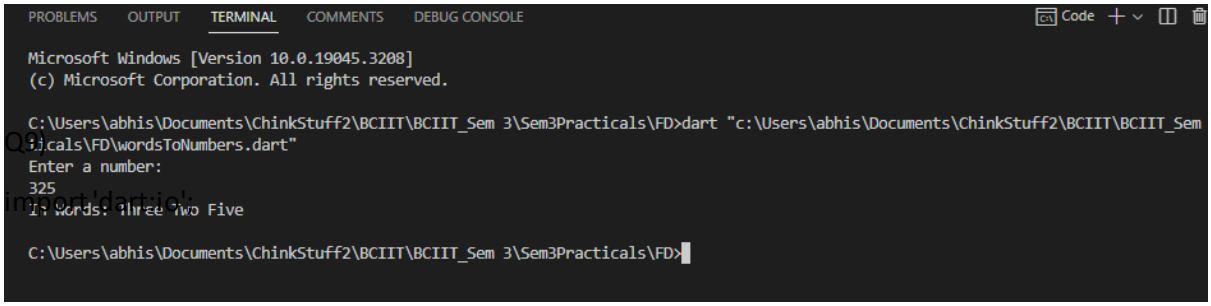


Q8)

```
import 'dart:io';

void main() {
  print("Enter a number:");
  int number = int.parse(stdin.readLineSync() ?? '');

  String words = numberToWords(number);

  print("In Words: $words");
}

String numberToWords(int number) {
  List<String> wordsList = [];
```

```
  while (number > 0) {
    int digit = number % 10;
    wordsList.add(digitToWord(digit));
    number ~/= 10;
  }

  return wordsList.reversed.join(' ');
}

String digitToWord(int digit) {
  switch (digit) {
    case 0:
      return "Zero";
    case 1:
      return "One";
    case 2:
      return "Two";
    case 3:
      return "Three";
    case 4:
      return "Four";
    case 5:
      return "Five";
    case 6:
      return "Six";
    case 7:
      return "Seven";
    case 8:
      return "Eight";
    case 9:
      return "Nine";
```

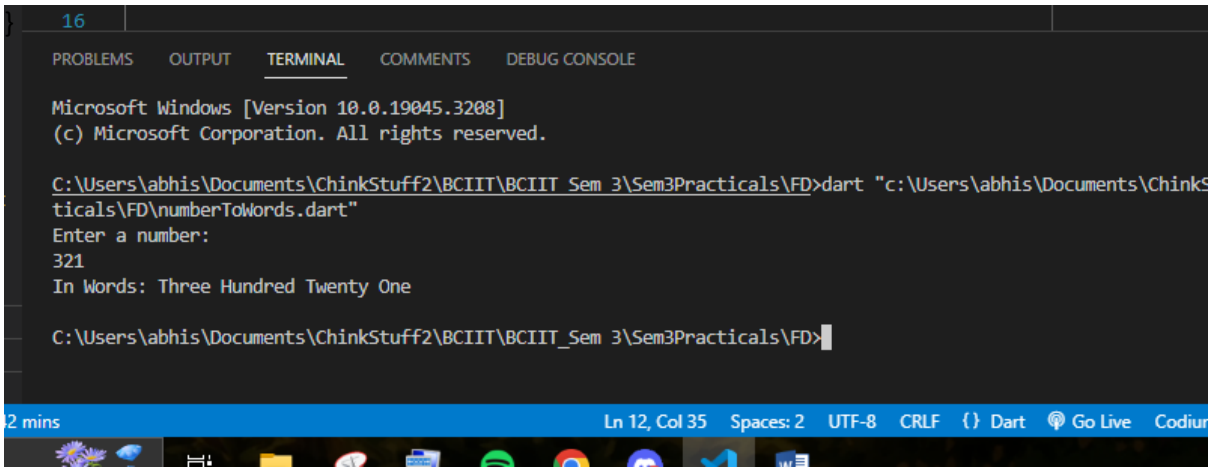
```
    default:
      return "";
  }
}
```




```
void main() {
  print("Enter a number:");
  int number = int.parse(stdin.readLineSync() ?? '');

  String words = numberToWords(number);

  print("In Words: $words");
}

String numberToWords(int number) {
  if (number == 0) {
    return "Zero";
  }

  List<String> units = [
    "",
    "One",
    "Two",
    "Three",
    "Four",
```

```
  "Five",
  "Six",
  "Seven",
  "Eight",
  "Nine",
  "Ten",
  "Eleven",
  "Twelve",
  "Thirteen",
  "Fourteen",
  "Fifteen",
  "Sixteen",
  "Seventeen",
  "Eighteen",
  "Nineteen"
];

List<String> tens = [
  "",
  "",
  "Twenty",
  "Thirty",
  "Forty",
  "Fifty",
  "Sixty",
  "Seventy",
  "Eighty",
  "Ninety"
];

String result = "";
```

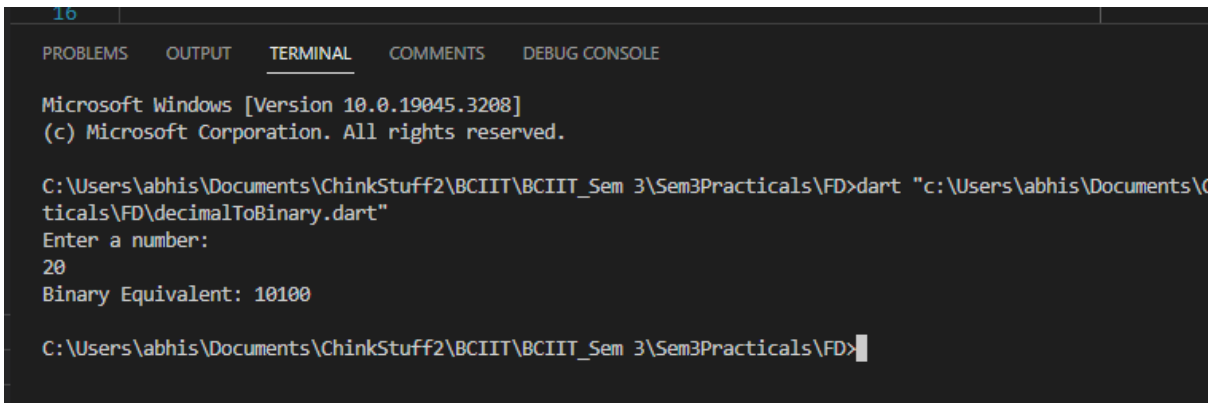
```
  if (number >= 100) {
    result += units[number ~/ 100] + " Hundred ";
    number %= 100;
  }

  if (number >= 20) {
    result += tens[number ~/ 10] + " ";
    number %= 10;
  }

  if (number > 0) {
    result += units[number];
  }

  return result.trim();
```



Q10)

```
import 'dart:io';

void main() {
  print("Enter a number:");
```

```
  int number = int.parse(stdin.readLineSync() ?? '');

  String binaryEquivalent = decimalToBinary(number);

  print("Binary Equivalent: $binaryEquivalent");
}

String decimalToBinary(int number) {
  if (number == 0) {
    return "0";
  }

  String binary = "";
  while (number > 0) {
    int remainder = number % 2;
    binary = "$remainder$binary";
    number ~/= 2;
  }

  return binary;
}
```



All code pushed to my github repo:
https://github.com/ShanksCodes/Sem3Practicals/tree/main/FD