CADT IDT

W1-S2 *PRACTICE*

DART BASICS

⚠ *Before this practice*

- You need to have completed: `SELF-LEARNING 1 - Dart Syntax & Concepts`

Learning objectives

- Apply type **inference** for variable declarations.
- Handle **nullable** and **non-nullable** variables.
- Differentiate between **final** and **const**.
- Manipulate **strings, lists, and maps.**
- Use **loops** and **conditions** to control flow.
- Define and call functions with positional and **named arguments**, understand **arrow syntax**

No AI tools allowed to solve this practice

How to submit?

✓ **Push your final code** on this GitHub repository (if you are lost, follow this tutorial)

Are you lost?

Read the following documentation to be ready for this practice:

- ✓ Variables
- ✓ Null Safety
- ✓ Built-in types
- ✓ Lists
- ✓ Loops
- ✓ Conditions
- ✓ Functions

REVIEW SELF-LEARNING

In group of 3 or 4, **review the** self-learning work.

- After discussing with your peer, **update your answers** if you need
- Some groups will present **their outcome** to the classroom

2. Nullable and Non-Nullable Variables

`EXPLAIN` : Explain nullable vs non-nullable variables.

`EXPLAIN` : When is it useful to have nullable variables?

`CODE` : Complete the bellow code to illustrate the concepts:

```
// Declare a nullable integer variable and assign it a null value

// Declare a non-nullable integer variable and assign it a value

// Assign a new value to the nullable variable
```

3. Final and const

`EXPLAIN` : Describe the difference between `final` and `const`.

`CODE` : Complete the bellow code to illustrate the concepts:

```
// Declare a final variable and assign it the current date and time

// Can you declare this variable as const? Why?

// Declare a const variable with a integer value

// Can you reassign the value of this final variable? Why?
```

Review, and update your answer regarding your work

EX 1 – Manipulate Types

Are you clear about strings, integer, list, map, set, objects in Dart?
Examine the given data structures and **write the inferred Dart type** for each one (*see example*)

Notes

- *First find by yourself the type*
- *If you need, double check your answer with VSCode.*

Data	Dart Type

`const studentGrades = {`
`  'Sokan': [90, 85, 88],`
`  'Sokea': [70, 80, 75],`
`  'Hay': [95, 92, 89],`
`};`
	`Map<String, List<int>>`

`const pizzaPrices = {`
`  'margherita': 5.5,`
`  'pepperoni': 7.5,`
`  'vegetarian': 6.5,`
`};`
	`Map< String, double>`

`const books = [`
`  {'title': '1984', 'author': 'George Orwell'},`
`  {'title': 'Brave New World', 'author': 'Aldous Huxley'},`
`  {'title': 'Fahrenheit 451', 'author': 'Ray Bradbury'},`
`];`
	`List<Map<String, String>>`

`const company = {`
`  'HR': {'employees': 5, 'budget': 20000},`
`  'IT': {'employees': 10, 'budget': 50000},`
`  'Marketing': {'employees': 7, 'budget': 30000},`
`};`
	`Map<String,Map< String, int>>`

`const coordinates = [`
`  [1, 2],`
`  [3, 4],`
`  [5, 6],`
`];`
	`List<List<int>>`

`const productDetails = {`
`  'id': 101,`
`  'name': 'Laptop',`
`  'price': 999.99,`
`  'inStock': true,`
`};`
	`Map<String, Object>`

`const operations = [`
`  (int a, int b) => a + b,`
`  (int a, int b) => a - b,`
`  (int a, int b) => a * b,`
`];`
	`List<f(int,int)>`

`const distances = {3.1, 5.5, 10.2, 7.8};`	`Set<double>`

`const university = {`
`  'departments': [`
`  {`
`  'name': 'Computer Science',`
`  'students': [`
`  {'name': 'Alice', 'age': 22},`
`  {'name': 'Bob', 'age': 24},`
`  ]`
`  },`
`  {`
`  'name': 'Mathematics',`
`  'students': [`
`  {'name': 'Charlie', 'age': 21},`
`  {'name': 'Dave', 'age': 23},`
`  ]`
`  }`
`  ]`
`};`
	`Map < String, List <Map<String,Object>>>`

EX 2 – Manipulate final and const

In this exercise, you need to decide which variable can be declared as **const** or **final**.

```
// 1 - startText
String iLike = 'I like pizza';

// 2 - toppings
String toppings = 'with tomatoes';
toppings += " and cheese";

// 3 - message
String message = '$iLike $toppings';
print(message);

// 4 - rbgColors
List<String> rbgColors = ['red', 'blue', 'green'];
print(rbgColors);

// 5 - weekDays
List<String> weekDays = ['monday', 'tuesday', 'wednesday'];
weekDays.add('thursday');
print(weekDays);

// 6 - scores
List<int> scores = [45,35,50];
scores = [45,35,50, 78];
print(scores);
```

Guess which variables can be declared as **const**, **final** or **var**, and explain your choices.

Notes

- *Read here to understand the concepts.*
- *Prefer const over final over var.*

	VAR, FINAL, CONST?	**WHY**
`iLike`	CONST	*Because this variable never changes*
`toppings`	Var	*Because this changes a second time*
`message`	Final	*Because this value has assigned one time*
`rbgColors`	Const	*Because this variable cannot be change*
`weekDays`	Final	*Because this variable can be change while run time*
`score`	Const	*Because the variable can be assigned for two times*

EX 3 – Filter a list

Instructions

- You are given a list of integers representing the scores of students in an exam.
- A score of 50 or higher is considered passing.
- Write a Dart program that filters and returns a list of students and the number of students who passed the exam.

Constraints

- You must use the **where** function with a proper **anonymous function** to filter the original list
- More information here

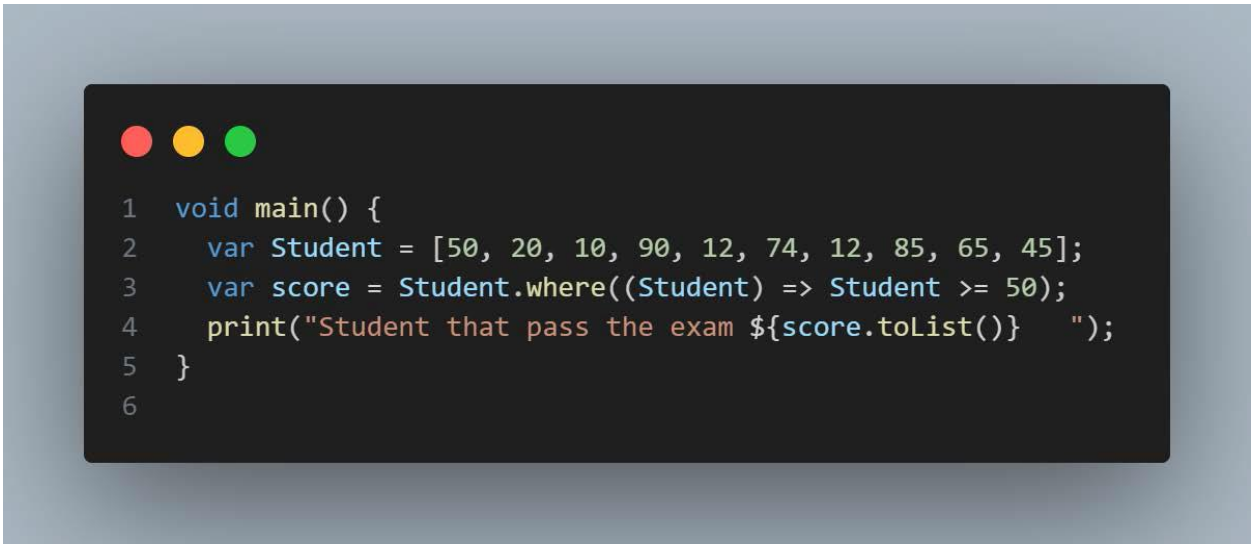
Examples

INPUT

```
[45, 78, 62, 49, 85, 33, 90, 50]
```

OUTPUT

```
[78, 62, 85, 90, 50]
5 students passed
```



```
void main() {
  var Student = [50, 20, 10, 90, 12, 74, 12, 85, 65, 45];
  var score = Student.where((Student) => Student >= 50);
  print("Student that pass the exam ${score.toList()}   ");
}

```

EX 4 – Manipulate maps

Given the following **map** of pizza prices:

```
const pizzaPrices = {
  'margherita': 5.5,
  'pepperoni': 7.5,
  'vegetarian': 6.5,
};
```

Write a program to calculate the total for a given order.

For example, given the following order:

```
const order = ['margherita', 'pepperoni'];
```

The program should print:

```
Total: $13`
```

If a pizza is not on the menu, the program should print:

```
pineapple pizza is not on the menu
```

```
void main() {
  const pizzaPrices = {
    'margherita': 5.5,
    'pepperoni': 7.5,
    'vegetarian': 6.5,
  };

  const order = ['margherita', 'vegetarian','pepperoni', 'gigi'];
  double Total = 0.0;
  for (var item in order) {
    if (pizzaPrices.containsKey(item)) {
      Total += pizzaPrices[item]!;
      print('$item : ${pizzaPrices[item]}');
    } else {
      print('$item is not avalaible on the menu');
    }

  }
  print('\$${Total}');
}
```

BONUS 1 – Write a robot simulator

A robot factory's test facility needs a program to verify robot movements.
The robots have three possible movements:

- turn right
- turn left
- advance

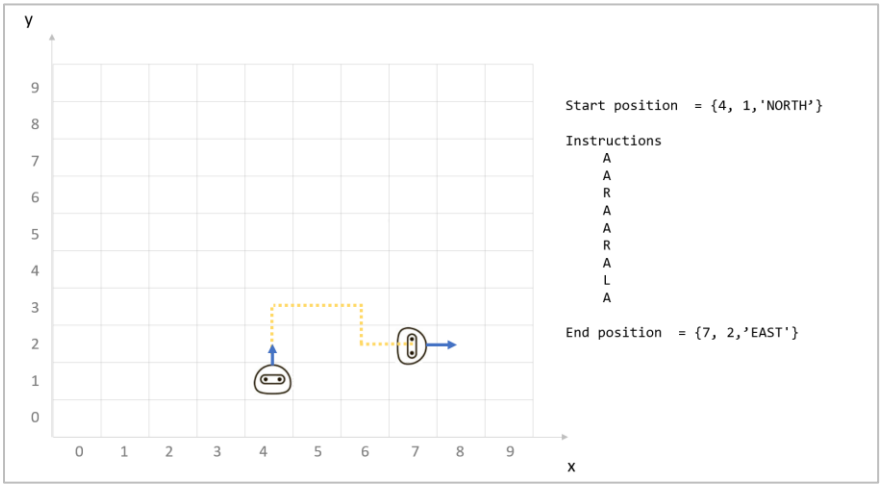
Robots are placed on a hypothetical infinite grid, facing a particular direction (NORTH, EAST, SOUTH, OR WEST) at a set of {X, Y} coordinates, e.g., {3,8}, with coordinates increasing to the north and east.

The robot then receives a number of instructions, at which point the testing facility verifies the robot's new position, and in which direction it is pointing.

As example

- the string "RAALAL" means:
    1. Turn right
    2. Advance twice
    3. Turn left
    4. Advance once
    5. Turn left yet again

- Say a robot starts at {7, 3} facing north.
- Then running this stream of instructions should leave it at {9, 4} facing west.



Note

- You are free to decide how to structure your data in Dart language
- Try to use as much as possible functions to separate your logic

BONUS 2 – Matching Brackets

Instructions

Given a string containing brackets [], braces {}, parentheses (), or any combination thereof, verify that any and all pairs are matched and nested correctly. Any other characters should be ignored.

Examples

INPUT	OUTPUT
`{what is (42)}?`	`Balanced`
`[text}`	`Not balanced`
`{[[(a)b]c]d}`	`Balanced`