NAME : MAKVANA NEEL

ROLL NO : CE064

ID : 20CEUOS086

BATCH : A4

Lab 04 code and comments:

Tutorial 1:

```dart
import 'package:lab4_tutorial1/lab4_tutorial1.dart' as lab4_tutorial1;

import 'dart:math';

/* Constructors are of 5 types:
(Constuctor with class name only is unnamed constructor)
1)Default Constructor
2)Custom Constructor(With parameter --> Generic Constructor)
  (i)Long-form constructor
  (ii)Short-form constructor
  (iii)named Constructor
3)Factory constructor
*/
class Password {
  final String value;
  const Password([value = '1234']) : this.value = value;
  bool isValid() {
    if (value.length >= 8) {
      return true;
    }
    return false;
```

```
  }

  @override
  String toString() {
    return "value:$value";
  }
}

class MyClass {
  var myProperty = 1;
}

class Email {
  var _address = '';
  String get value => _address;
  set value(String address) => _address = address;
}

class SomeClass {
  static int myProperty = 0;
  static void myMethod() {
    print('Hello, Dart!');
  }
}

class Student {
  Student({firstName = 'abc', lastname = 'xyz', grade = 0})
      : this.firstName = 'abc',
        this.lastName = 'xyz',
        this.grade = 100;
  final String firstName, lastName;
  final int grade;
}

class Shperes {
  static const PI = (22 / 7);
  const Shperes({int radius = 0}) : this.radius = radius;
  final int radius;
  double get volume => (4 / 3) * PI * radius * radius * radius;
  double get area => 2 * PI * radius;
}

/*
4 types of a variable:class variable,instance variable,global variable,local
variable
*/

class MySingleton {
  MySingleton.__();
  static final MySingleton _instance = MySingleton.__();
  factory MySingleton() => _instance;
```

```
}

//JSON -->Javascript object notation
void main(List<String> arguments) {
  // print('Hello world: ${lab4_tutorial1.calculate()}!');
  /*
  // Class
  // Classes are used to combine data and functions ins_ide a single structure.
  // Object creation is also called as an instantiating a class and object is an
insatnce of a class.
  final user = new User();//new is optional over here
  user._id = 1;
  user._name = 'abc';
  print(user); // Insatnce of 'User'
  print(user.toJson());
  final user1 = User()
  .._id = 2
  .._name = "xyz";
  // this .. is a cascade operator and ; is at the end only
  */
  /*
  // Mini exrecises
  Password p1 = Password();
  p1.value = 'swami@0123';
  print(p1.isVal_id());
  p1.value = '1234';
  print(p1.isVal_id());
  */
  /*
  //_named constructor
  User user2 = User(id:0,name:'anonymous');
  print(user2);
  const obj1 = User.anonymous();
  const obj2 = User.anonymous();
  // canonical objects
  print(identical(user2,obj2));//This is false
  print(identical(obj1,obj2));//This is true
  print(user2.id);
  */
  /*
  final jb = User(id:1,name:'JB Lorenzo');
  final map = {'id':10,'name':'jekils'};
  final jekils = User.fromJson(map);//Factory constructor
  */
  final email = Email();
  email.value = 'abc@xyz.com';
  final emailString = email.value;
  print(emailString);

  final value = SomeClass.myProperty;
  SomeClass.myMethod();
```

```dart
  // value.myMethod(); // This is not valid
  /*
  // Refernce of object(not deep copy)
  final myObject = MyClass();
  final another = myObject;
  myObject.myProperty = 2;
  print(another.myProperty);
  */

  final mySingleton = MySingleton();
  // Challanges:
  // Challange 1:
  final Student bert = Student(firstName: 'bert', grade: 95);
  final Student ernie = Student(firstName: 'ernie', grade: 85);
  // Challange 2:
  Shperes s1 = Shperes(radius: 12);
  print(s1.area);
  print(s1.volume);
}
```

Tutorial 2 :

```dart
import 'package:lab4_tutorial2/lab4_tutorial2.dart' as lab4_tutorial2;

import 'dart:math';


/*
class User {
  String? name;
  int? id;
}
*/

bool isPositive(int? anInteger) {
  if (anInteger == null) {
    return false;
  }
  return !anInteger.isNegative;
}

class User {
```

```dart
  User(this.name);
  final String name;
  // If we don't write late then this will give an error
  // Using late means that Dart doesn't initialize the variable right away. It only
initializes it when you access it the first time. This is also known as lazy
initialization.
  late final int _secretNumber = _calculateSecret();
  int _calculateSecret() {
    return name.length + 42;
  }
  /*
    //This will work
    User(this.name) {
    _secretNumber = _calculateSecret();
    }
    late final int _secretNumber;
  */
}

class User1 {
  // Here we have to initialize name
  //Using initializing formals
  User1(this.name);
  /* //(Using an initializer list)
    User(String name)
    : _name = name;
    String _name;
  */
  /* //(Using default parameter values)
    User([this.name = 'anonymous']);
    String name;
    //or
    User({this.name = 'anonymous'});
    String name;
  */
  /* //(required name parameters)
    User({required this.name});
    String name;
  */
  String name;
}

class Name {
  Name({givenName = '', surname = '', surnameIsFirst = false});
  String givenName, surname;
  bool surnameIsFirst;
}

int? fun() {
  var random = new Random();
  int? num = random.nextInt(1);
```

```dart
  if (num == 0) {
    num = null;
  }
  return num;
}

void main(List<String> arguments) {
  print('Hello world: ${lab4_tutorial2.calculate()}!');
  /*
  // Into 。safety
  print(isPositive(3)); // true
  print(isPositive(-1)); // false
  // print(isPositive(Null)); // This will give error as null is not an integer
  */

  /*
  // Nullable and non-nullable
  // Nullable types end with the '?'

  // Non nullable:(type which can't take null value)
  // dart types are non nullable means we can't assign null to it that's why we get
an error in above function call
  // int postalCode = null //error

  //Nullable types:
  int ? myInt;
  print(myInt);
  */

  /*
  // Mini exercises:
  // Exercise:1
  String? preofession;
  print(preofession);
  preofession = "basketball player";
  const iLove = 'Dart';//iLove is inferred as String
  */

  String? name;
  // print(name.length);//This will results into an error
  name = "xyz";
  print(name.length);

  /*
  // Null aware operators:
  // 1) If-null operator(??)
  String? message;
  final msg = message ??
      'No message'; //If message is null then 'No message' is the value of msg
  print(msg);
```

```
  // 2)Null-aware assignment operator(??=)
  int? x;
  x ??= 10; //Same as x = x ?? 10;

  // 3)Null aware access operator//null aware method operator
  print(x?.isNegative);

  // 4)Null assertion operator(!) or bang operator
  int num =
      13!; //It tells that right hand side value is not null and program will crash
if it will be null at runtime

  // 5)Null aware cascade operator(?..)
  User user = User()
    ..id = 42
    ..name = 'abc';
  // If object is nullable then
  User? user1 = User()
    ?..id = 42
    ..name = 'xyz';

  // We can have the chain of the operator
  String? lengthString = user?.name?.length.toString();

  // 6)Null aware index operator(?[]):
  List<int>? myList = [1, 2, 3];
  myList = null;
  int? myItem = myList?[2];
  print(myItem);
  */

  // Challenges:
  // Challenge 1:
  int temp = fun() ?? 0;
  print(temp);

  // Challenge 2:

}
```