Quiz

1. What will be the output when the following code is executed?

```
class Vehicle {
  void drive() {
    print("Driving vehicle...");
  }
}

class Car extends Vehicle {
  void honk() {
    print("Car honking...");
  }
}

void main() {
  Car myCar = Car();
  myCar.drive();
  myCar.honk();
}
```

A) Driving vehicle... Car honking...
B) Car honking... Driving vehicle...
C) Car honking...
D) Driving vehicle...

Answer: A) Driving vehicle... Car honking…

2. What is the purpose of an abstract class in Dart?
A) To prevent instantiation of the class
B) To provide a blueprint for creating objects
C) To restrict access to certain class members
D) To encapsulate private data within the class

Answer: B) To provide a blueprint for creating objects

3. What is a Superclass in Dart inheritance?
A) A class that is inherited by another class
B) A class that is derived from other classes
C) A class that is created without any inheritance
D) A class that has no subclasses

Answer: A) A class that is inherited by another class

4. What will be the output when the following code is executed?

```
class Animal {
  void eat() {
    print("Animal eating...");
  }
}

class Cat extends Animal {
  void meow() {
    print("Meow!");
  }
}

void main() {
  Animal myCat = Animal();
  myCat.meow();
}
```
A) Animal eating...
B) Meow!
C) Animal eating...Meow!
D) Compilation error

Answer: D) Compilation error

5. What enables code reuse and promotes a hierarchical structure among classes?
A) Encapsulation
B) Polymorphism
C) Inheritance
D) None of the above

Answer: C) Inheritance

6. What is another term for a parent class in Dart inheritance?
A) Derived class
B) Subclass
C) Superclass
D) Base class

Answer: C) Superclass & D)Base class

7. What is the method in the superclass called when method overriding occurs?
A) Method overridden
B) Method overrided
C) Method inherited
D) Method superclass

Answer: A) Method overridden

8. What will be the output when the following code is executed?

```
class Shape {
  void draw() {
    print("Drawing shape...");
  }
}

class Circle extends Shape {
  @override
  void draw() {
    print("Drawing circle...");
  }
}

void main() {
  Circle myCircle = Circle();
  myCircle.draw();
}
```

A) Drawing shape...
B) Drawing circle...
C) Compilation error
D) No output

Answer: B) Drawing circle…

9. What will be the output when the following code is executed?

```
class Animal {
  void eat() {
    print("Animal eating...");
  }
}

class Lion extends Animal {
  @override
  void eat() {
    super.eat();
    print("Lion eating meat...");
  }
}

void main() {
  Lion myLion = Lion();
  myLion.eat();
}
```

A) Animal eating... Lion eating meat...
B) Lion eating meat... Animal eating...
C) Lion eating meat...
D) Compilation error

Answer: A) Animal eating... Lion eating meat...

10. What is the purpose of method overriding in Dart?
A) To prevent access to certain methods in the superclass
B) To modify the behaviour of methods in the superclass
C) To create new methods in the subclass without any relation to the superclass
D) To define private methods that cannot be accessed outside the subclass

Answer: B) To modify the behaviour of methods in the superclass

11. Which annotation is used by a Subclass to modify behaviour of Superclass?
A)@annotate
B)@generate
C)@overriden
D)@override

Answer: D)@override

12. Can a subclass override a method in the superclass and change its return type?
A) Partially true
B) Yes
C) Partially false
D) No

Answer: D) No

13. What is Dart inheritance?
A) The process of modifying an existing class
B) The process of creating a new class from scratch
C) The process of deriving properties and characteristics of another class
D) The process of reusing code from an external library

Answer: C) The process of deriving properties and characteristics of a new class

14. What will be the output?

```
abstract class Animal {
  void makeSound();
}

class Dog extends Animal {
  @override
  void makeSound() {
    print("Dog barking...");
  }
}

void main() {
  Animal myAnimal = Dog();
  myAnimal.makeSound();
}
```

A) Dog barking...
B) Compilation error
C) Animal making sound...
D) No output

Answer: A) Dog barking…

15. Which of the following statements best describes the purpose of debugging in Dart?
A) To optimise the performance of the code
B) To identify and fix errors or bugs in the code
C) To enhance the readability of the code
D) To document the code for future reference

Answer: B) To identify and fix errors or bugs in the code

16. Can an abstract class be instantiated directly?
A) Yes, an abstract class can be instantiated without any issues
B) No, an abstract class cannot be directly instantiated
C) Yes, but only if all the abstract methods are implemented
D) Yes, but it requires the use of the new keyword

Answer: B) No, an abstract class cannot be directly instantiated

17. Which keyword is used to import code from an external file in Dart?
A) using
B) require
C) import
D) include

Answer: C) import

18. What is the purpose of breakpoints in the debugging process in Flutter?
A) Breakpoints are used to terminate the execution of a Flutter application.
B) Breakpoints are used to pause the execution of a Flutter application at a specific line of code.
C) Breakpoints are used to slow down the execution of a Flutter application for performance analysis.
D) Breakpoints are used to skip certain parts of code during debugging.

Answer: B) Breakpoints are used to pause the execution of a Flutter application at a specific line of code.

19. In Dart, how do we declare an abstract class?
A) Using the abstract keyword before the class declaration
B) Using the abstract keyword before the method declaration
C) Using the abstract keyword after the class declaration
D) Using the abstract keyword after the method declaration

Answer: A) Using the abstract keyword before the class declaration

20. How does debugging help in understanding complex program flow in Dart?
A) By simplifying the code structure
B) By providing real-time execution visualisation
C) By automatically optimising the program flow
D) By generating detailed documentation

Answer: B) By providing real-time execution visualisation

Assignment

Define an **abstract** class **Account** with the following properties:

- ***accountNumber*** *(integer)*
- ***balance*** *(double)*

It should also have the following methods:

- ***deposit(double amount)****:* Adds the specified amount to the account balance.
- ***withdraw(double amount)****:* Abstract method that deducts the specified amount from the account balance.

Define a class **SavingsAccount** that extends the Account class. It should have an additional property called ***interestRate*** *(double)*.

Implement the ***withdraw()*** method in the SavingsAccount class as follows:

- Subtract the specified amount from the account balance.
- Apply the interest rate to the remaining balance.

Define a class **CurrentAccount** that **extends** the **Account** class. It should have an additional property called ***overdraftLimit*** (double).

Implement the ***withdraw()*** method in the CurrentAccount class as follows:

- Allow withdrawals up to the overdraft limit.
- If the withdrawal amount exceeds the overdraft limit, print a message indicating insufficient funds.

In main()

- Create an instance of the SavingsAccount class by providing values for the account number, initial balance, and interest rate.
- Use the instance to perform operations like depositing and withdrawing money.
- Create an instance of the CurrentAccount class by providing values for the account number, initial balance, and overdraft limit.
- Use the instance to perform operations like depositing and withdrawing.

Solution:

```
abstract class Account {
  int accountNumber;
  double balance;

  Account(this.accountNumber, this.balance);

  void deposit(double amount) {
    balance += amount;
    print('Deposit of $amount successful. New balance: $balance');
  }


  void withdraw(double amount) {
    if (balance >= amount) {
      balance -= amount;
      print('Withdrawal of $amount successful. New balance: $balance');
    } else {
      print('Insufficient funds. Cannot withdraw $amount');
    }
  }
}

class SavingsAccount extends Account {
  double interestRate;

  SavingsAccount(int accountNumber, double balance, this.interestRate)
      : super(accountNumber, balance);

  @override
  void withdraw(double amount) {
    if (balance >= amount) {
      balance -= amount;
      balance += balance * interestRate;
      print('Withdrawal of $amount successful. New balance: $balance');
    } else {
      print('Insufficient funds. Cannot withdraw $amount');
    }
  }
}

class CurrentAccount extends Account {
  double overdraftLimit;
  CurrentAccount(int accountNumber, double balance, this.overdraftLimit)
      : super(accountNumber, balance);

  @override
  void withdraw(double amount) {
    if (balance + overdraftLimit >= amount) {
      balance -= amount;
      print('Withdrawal of $amount successful. New balance: $balance');
    } else {
      print('Insufficient funds. Cannot withdraw $amount');
    }
  }
}

void main() {
  // Create a savings account
  SavingsAccount savings = SavingsAccount(123456, 5000, 0.05);
  savings.deposit(2000);
  savings.withdraw(3000);
  savings.withdraw(10000); // Will apply interest rate

  // Create a current account
  CurrentAccount current = CurrentAccount(789012, 10000, 5000);
  current.deposit(3000);
  current.withdraw(5000);
  current.withdraw(12000); // Will display insufficient funds
}
```

Live Test

There is a base class called **Media** and it has a method called ***play()*** that prints "Playing media...".

You need to create a derived class called **Song** that inherits from the **Media** class and adds an additional attribute called ***artist*** (string). The Song class should **override** the ***play()*** method to print the artist name along with the media play message like "Playing song by $artist...'".

In main() create one instance of Media and one of Song. Call their play() methods that print proper messages.

Solution:

```
class Media {
  void play() {
    print('Playing media...');
  }
}

class Song extends Media {
  String artist;

  Song(this.artist);

  @override
  void play() {
    print('Playing song by $artist...');
  }
}

void main() {
  Media media = Media();
  media.play();

  Song song = Song('James');
  song.play();
}
```