1. **Data Types**

a. Primitive Data types

// If you will assign any value, return will be null

- String

String myText = ' I can create a text on Apex'; // single quotes make anything string
myText.capitalize();// available functions with . notation

- Integer

Integer myNumber = 4;// whole number up until ~2 billion

- Long

Long myLongNumber = 3000000000L; // bigger than ~2 billion, use L at the end of L; (why we use Integer instead of Long; due to performance)

- Decimal

Decimal myDecimalNumber = 3.55 // decimal up to ~2 billion

- Double

Double myDoubleNumber = 3000000000.56 // bigger than ~2 billion

- Date

Date myDate = Date.newInstance(2022,12,21);
Date currentDate = Date.today();
currentDate.format();

- DateTime // Date, DateTime and Time logics are the same

- Time // Date, DateTime and Time logics are the same

- Boolean

Boolean myChoice = true; // True, False or null

- Id

Id myId = '001Dn0000092Z0KIAU;// 15 or 18 alphanumeric characters

- Blob

Blob myBlob = Blob.valueOf('Test');// (eg.11010010) – attachment, webservices or files

b. Collections

- List //

```
List<String> myList = new List<String>();
myList.add('item1');
myList.add('item2');
myList.remove(0); // indexed ordered list starting from 0
```

- Set // values are unique

```
Set<Integer> mySet= new Set<Integer>();
mySet.add(5);
mySet.add(4);
mySet.add(5);// this will not add since all values are unique in the Set
mySet.remove(4); // there is no index
```

- Map// key value pairs; keys shpuld be unique, value can be the same (Eg. Student Id and name

```
Map< String, String> myMap= new Map<String, String>();
myMap.put('S0001', 'Michael');
myMap.put('S0002', 'Michael'); // key is unique, so value can be the same
myMap.remove('S0001'); // put key value to remove
```

c. Constants

- Final

```
Final Decimal PI_Number = 3.14; // SNAKE_CASE
Final Decimal DISCOUNT_RATE = .05  // 5% discount
```

- Enum

```
Enum String SEASON {'WINTER', 'SUMMER'} // similar to global picklist values
Season.Winter
```

2. Control Flow Statements

a. If Else

```
If(x>10){ //If(first logic - true or false)

    //code block to execute if logic is true
} else if(x==6){

                    //code block to execute if logic is true
 } else if(x==3){

    //code block to execute if logic is true
 } else {

    //code block to execute if none of the logics above true
 }
```

b. Switch

```
Swithc on x{
When 1{
//code block to execute
}
When 2, 4, 6 {
//code block to execute

}
When 9. 10{
//code block to execute

}
When else {
//code block to execute

}
```

c. **Loops**

- **Do While**

```
Do {
//the code will run at leas one time
// do not forget to add exit logic like increase or decrease otherwise CPU limit will stop the loop
} while (condition)
```

- **While**

```
While(condition){
//it may not be run if condition will not met
// do not forget to add exit logic like increase or decrease otherwise CPU limit will stop the loop

}
```

- **For**
  - **Traditional For**

```
For (Integer myInteger = 1; myInteger<5; myInteger++)
//( Initial statement ; exit condition; increase or decrease)
{
//code block to execute
}
```

  - **List Set For**

```
List<Account> myAccountList = List<Account>();
For( Account newAccount : myAccountList){
//code block to execute
}
```

  - **SOQL For**

```
For( Account newAccount :[SELECT Name FROM Account]){
//code block to execute
}
```

3. **Classes**

    a. **Class**

```
public class ClassName{ // class declaration and ClassName is required

    }
```

    b. **Method**

```
public void MethodName(){ // return type and MethodName() is required

        }
```

Encapsulation

Encapsulation in Apex is the technique of organizing code into classes and methods that hide the implementation details of a class from other parts of the code, helping to improve maintainability and reusability, and enforce data integrity.

- **Abstraction**

Abstraction in Apex is the technique of defining a class that represents the common characteristics and behaviors of a group of related objects, without providing the implementation details for those characteristics and behaviors.

- **Inheritance**

Inheritance in Apex is the ability of a class to inherit properties and methods from a parent class, allowing the child class to reuse the code and functionality of the parent class and extend or modify it as needed.

- **Polymorphism**

Polymorphism in Apex is the ability of a class to have multiple methods with the same name but different implementations, or for an object to behave differently depending on its type or the context in which it is used.

- **Inner Class**

An inner class in Apex is a class that is defined within another class, and has access to the private variables and methods of the outer class. Inner classes are often used to provide additional functionality or to organize related code within a larger class.

- **Try Catch Finally Error Handling**

The try, catch, and finally blocks in Apex are used to handle exceptions that may be thrown during the execution of code. The try block contains the code that may throw an exception, the catch block contains the code that handles the exception, and the finally block contains code that is always executed regardless of whether an exception was thrown.