Submitted by: **Ayesha Sajid**

Submitted to: **Sir Zulqarnain**

Department: **Computer Science**

Semester: **2nd**

Assignment: **OOP (Midterm)**

Roll number: **207704**

**Question #1**

**Procedural language**

**Definition**

A procedural language is a computer programming language that follows, in order, a set of commands.

**Explanation**

The procedural programming aims at dividing the large program into smaller programs called procedures. The procedures are also alternately referred to as subprograms, subroutines, methods or functions.

In procedural programming language, the program code is organized as set of procedures called functions. These functions operate on the program data called the variables.

In procedural programming, the program data is in the form of variables the functions operate on the programmed data. Each function consist up of computational statements and solves a part of program.

The procedural programming continues to be the preferred choice for the beginners as a first programming language. The C procedural language is extensively used in the industry for the system programming.

**Examples**

Examples of computer procedural languages:

* BASIC
* C
* FORTRAN
* Java
* Pascal

**Object oriented programming**

**Definition**

“Object Oriented programming (OOP) is a programming paradigm that relies on the concept of classes and objects. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects.”

**Explanation**

• Object-oriented programming is about creating objects that contain both data and functions

• OOP is faster and easier to execute.

• When developing in Object-oriented Programming there are many tools available that will aid the programmer while developing. Many languages have software built to help aid developers while coding in their language.

**Examples**

* C++
* Java
* Python
* C#

**DIFFERENCE BETWEEN PROCEDURAL PROGRAMMING AND OBJECT ORIENTED PROGRAMMING:**

**Procedural Programming**

**Object Oriented Programming**

1. **Division of Programs**

* **I**n procedural programming, program is divided into small parts called functions.
* In object oriented programming, program is divided into small parts called objects.

2. **Approach**

* Procedural programming follows top down approach.
* Object oriented programming follows bottom up approach.

3. **Access Specifiers**

* There is no access specifier in procedural programming.
* Object oriented programming have access specifiers like private, public, protected etc.

4. **Addition of new Data**

* Adding new data and function is not easy.
* Adding new data and function is easy.

5. **Data Security**

* Procedural programming does not have any proper way for hiding data so it is less secure.
* Object oriented programming provides data hiding so it is more secure.

6. **Overloading**

* In procedural programming, overloading is not possible.
* Overloading is possible in object oriented programming.

7. **Importance over Function and data**

* In procedural programming, function is more important than data.
* In object oriented programming, data is more important than function.

8. **Bases**

* Procedural programming is based on unreal world.
* Object oriented programming is based on real world.

**Examples**

* Examples: C, FORTRAN, Pascal, Basic etc.
* Examples: C++, Java, Python, C# etc.

**THE NEED OF OBJECT ORIENTED APPROACH**

Object-oriented programming has several advantages over procedural programming:

• OOP is faster and easier to execute

• OOP provides a clear structure for the programs

• OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug

• OOP makes it possible to create full reusable applications with less code and shorter development time

**Features of OOP**

We use object oriented approach due to its following features:

An object-oriented system comes with several great features which are discussed below.

**Encapsulation**

Encapsulation is a process of information hiding. It is simply the combination of process and data into a single entity. Data of an object is hidden from the rest of the system and available only through the services of the class. It allows improvement or modification of methods used by objects without affecting other parts of a system.

**Abstraction**

It is a process of taking or selecting necessary method and attributes to specify the object. It focuses on essential characteristics of an object relative to perspective of user.

**Relationships**

All the classes in the system are related with each other. The objects do not exist in isolation they exist in relationship with other objects.

There are three types of object relationships −

• Aggregation − It indicates relationship between a whole and its parts.

• Association − In this, two classes are related or connected in some way such as one class works with another to perform a task or one class acts upon other class.

• Generalization − The child class is based on parent class. It indicates that two classes are similar but have some differences.

**Inheritance**

Inheritance is a great feature that allows to create sub-classes from an existing class by inheriting the attributes and/or operations of existing classes.

**Polymorphism and Dynamic Binding**

Polymorphism is the ability to take on many different forms. It applies to both objects and operations. A polymorphic object is one who true type hides within a super or parent class.

In polymorphic operation, the operation may be carried out differently by different classes of objects. It allows us to manipulate objects of different classes by knowing only their common properties

**PROS AND CONS OF OBJECT ORIENTED PROGRAMMING AND PROCEDURAL PROGRAMMING:**

There are many advantages to programming in both Object-oriented Programming and Procedural Programming languages. Many argue the superiority of each method, but both serve their purpose and both have their advantages and disadvantages.

**PROS OF OBJECT-ORIENTED PROGRAMMING**

The obvious advantages of using Object-oriented Programming are

• reliability and sustainability across different platforms

• ease of debugging the code.

• Errors that may exist during development can usually be tracked to a point of origin and then fixed eliminating other errors throughout the entire program

• the logical structure of Object-oriented Programming

• The ability to relate code to real world examples makes the process much easier and more simple to code

• majority of program diagrammed out before you ever type out lines of code

**CONS OF OBJECT-ORIENTED PROGRAMMING**

Some cons of object oriented programming are:

• The difficulty of understanding how objects, classes, methods, actions etc relate to each other. Especially for those who are used to a Procedural Programming approach to programming this may be very difficult.

• The requirement to have packages and libraries installed for the code to function properly. These libraries are meant to make the coding process much easier for the developer, but for some who prefer to write out line by line their code it could become confusing and difficult to understand.

**PROS OF PROCEDURAL PROGRAMMING**

Some pros of Procedural Programming are

• **Ability to jump right into coding a program without the need to create any objects or classes**.

Many programmers prefer this method because they want to be able to write small amounts of code in a short period of time without the need to plan it out. Examples could include simple forms or html pages with basic data displayed.

• **Ability to learn the language easily.**

Many developers begin their education by copying code from the internet and then pasting it in an application to run. With Object-oriented Programming there is more to do besides just copying and pasting code, but in Procedural Programming that is generally an acceptable way to build a program.

• The top down structure is also an advantage for those who prefer to work their way through a program vs. planning the program out before the development process.

Many developers who are creative prefer to write their code.

**CONS OF PROCEDURAL PROGRAMMING**

Some cons of procedural programming are:

• Its method of programming is the inability to reuse code throughout the program. Having to rewrite the same type of code many times throughout a program can add to the development cost and time of a project.

• Another disadvantage is the difficulty in error checking.

• In Procedural Programming the code is continuously broken down into smaller manageable pieces until the different problems or functions can be executed. With this approach finding errors in the code becomes more and more difficult as the code gets longer and longer.

• The difficulty for developers who use Procedural Programming as their approach to programming to move from language to language. Without a knowledge of Object-oriented Programming it is very difficult for a developer to be qualified in many languages who use this method.

**Question #2**

**Objects**

**Definition**

Object is an single instance of class

When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

An object is nothing but a self-contained component that consists of methods and properties to make a data useful. It helps you to determines the behavior of the class

**Syntax**

**Object creation**

className objectName;

**Accessing member functions**

objectName.memberFunctio();

**Use of Object**

Here are the important uses of an object

* It helps you to know the type of message accepted and the type of returned responses.
* You can use an object to access a piece of memory using an object reference variable.
* It is used to manipulate data.
* Objects represent a real-world problem for which you are finding a solution.
* It enables data members and member functions to perform the desired task.

**Class**

**Definition**

Class is the blue print for the object.

**Explanation**

A class in C++ is the building block, that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object.

**Example**

Consider the Class of Cars. There may be many cars with different names and brand but all of them will share some common properties like all of them will have 4 wheels, Speed Limit, Mileage range etc. So here, Car is the class and wheels, speed limits, mileage are their properties.

**Components of Class**

A Class is a user defined data-type which has following memebers

* Data members
* Member functions.

**Data members**

Data members are the data variables

**Member functions**

Member functions are the functions used to manipulate these variables and together

Data members and member functions defines the properties and behavior of the objects in a Class.

**Example explanation**

In the above example of class Car, the data member will be speed limit, mileage etc and member functions can be apply brakes, increase speed etc.

**Syntax of class**

A class is defined in C++ using keyword class followed by the name of class. The body of class is defined inside the curly brackets and terminated by a semicolon at the end.

**class className;**

**{**

**Access specifiers**; //can be private, protected or public

**Data Members**; //Variables to use

**Member functions()** //Methods to access data members

{

}

**};**// class ends with semi colon

**Uses of Class**

Here are the important uses of class:

* Class is used to hold both data variables and member functions.
* It enables you to create user define objects.
* Class provides a way to organize information about data.
* You can use class to inherit the property of other class.
* Classes can be used to take advantage of constructor or destructor.
* It can be used for a large amount of data and complex applications.

**Differences between objects and class**

* A class is a template for creating objects in program whereas the object is an instance of a class.
* A class is a logical entity while object is a physical entity.
* A class does not allocate memory space on the other hand object allocates memory space.
* You can declare class only once but you can create more than one object using a class.
* Classes can't be manipulated while objects can be manipulated.
* Classes doesn't have any values, whereas objects have its own values.
* You can create class using "class" keyword while hand you can create object using "new" keyword in Java.

**Inheritance**

**Introduction**

One of the most important concepts in object-oriented programming is that of inheritance.

**Definition**

Inheritance allows us to define a class in terms of another class, which makes it easier to create and maintain an application.

(or)

The capability of a class to derive properties and characteristics from another class is called Inheritance

**Explanation**

When creating a class, instead of writing completely new data members and member functions, the programmer can designate that the new class should inherit the members of an existing class. This existing class is called the **base class**, and the new class is referred to as the **derived class.**

The idea of inheritance implements the is a relationship.

**Example**,

Mammal IS-A animal, dog IS-A mammal hence dog IS-A animal as well and so on.

**Base and derived classes**

**Sub Class, derived class or child class:**

**Definition #1**

The class that inherits properties from another class is called Sub class or Derived Class.

**Definition #2**

  The class that inherits from another class

**Super Class, base class or parent class**

**Definition #1**

The class whose properties are inherited by sub class is called Base Class or Super class

**Definition #2**

The class being inherited from

**Modes of Inheritance**

1. **Public mode**: If we derive a sub class from a public base class. Then the public member of the base class will become public in the derived class and protected members of the base class will become protected in derived class.
2. **Protected mode**: If we derive a sub class from a Protected base class. Then both public member and protected members of the base class will become protected in derived class.
3. **Private mode**: If we derive a sub class from a Private base class. Then both public member and protected members of the base class will become Private in derived class.

**Syntax of inheritance**

class parent\_class

{

//Body of parent class

};

class child\_class : access\_modifier parent\_class

{

//Body of child class

};

**Types of Inheritance in C++**

1) Single inheritance  
2) Multilevel inheritance  
3) Multiple inheritance  
4) Hierarchical inheritance  
5) Hybrid inheritance

**1)Single inheritance**

In Single inheritance one class inherits one class exactly

**2)Multilevel Inheritance**

In this type of inheritance one class inherits another child class

**3)Multiple Inheritance**

In multiple inheritance, a class can inherit more than one class. This means that in this type of inheritance a single child class can have multiple parent classes.

**4)Hierarchical Inheritance**

In this type of inheritance, one parent class has more than one child class.

**4)Hierarchical Inheritance**

In this type of inheritance, one parent class has more than one child class. For example:

**Code reusability**

**Introduction**

Code reusability is one of the most important advantage of inheritance.

**Definition**

Reusability in computer programming is the measurement of the likelihood that a given unit of code can be successfully incorporated into another program.

**Uses of Reusability**

* The reusability of a code hinges on the ability of other programmers to be able to look at the code and understand it and its properties
* Reusability is the quality of a code being used in different platforms for multiple functions
* For reusability to work, a known method is known as modularization, wherein a code is broken into modules.

**Misuses of Reusability**

* When reusability is applied to code or program, it is compulsory that the code is a closed source programming
* For reusability to work, it is not compulsory the code is broken into modules

**Code Reuse Technical Challenges**

Software quality plays a major role in how easily software can be reused.

There are four major traits of software quality that impact reuse.

1. **Security**

To be reused, code needs to be secure. You can ensure secure code in a number of ways, including using a CWE list.

1. **Reliability**

**T**o be reused, code needs to be reliable. You can ensure reliable code by ensuring availability, fault tolerance, and recoverability.

1. **Performance Efficiency**

To be reused, code needs to be efficient. You can ensure efficiency by improving response times and monitoring processor, memory, and utilization.

1. **Maintainability**

To be reused, code needs to be maintainable. One way to ensure that code is maintainable is to ensure it is compliant.

**Polymorphism**

**Introduction**

Poly mean many

Morphism means shapes

 it occurs when we have many classes that are related to each other by inheritance.

**Definition**

C++ polymorphism means that a call to a member function will cause a different function to be executed depending on the type of object that invokes the function.

**Polymorphism is the feature of oop**

Polymorphism is a feature of OOPs that allows the object to behave differently in different conditions. In C++ we have two types of polymorphism:

1) Compile time Polymorphism – This is also known as static (or early) binding.

2) Runtime Polymorphism – This is also known as dynamic (or late) binding

**1) Compile time Polymorphism**

Function overloading and Operator overloading are perfect example of Compile time polymorphism.

**2) Runtime Polymorphism**

Function overriding is an example of Runtime polymorphism.  
**Function Overriding**:

When child class declares a method, which is already present in the parent class then this is called function overriding, here child class overrides the parent class

**Overloading**

**Definition**

The process of having two or more functions with the same name, but different parameters, is known as function overloading.

**Explanation**

An overloaded declaration is a declaration that is declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and obviously different definition (implementation).

When you call an overloaded function or operator, the compiler determines the most appropriate definition to use, by comparing the argument types you have used to call the function or operator with the parameter types specified in the definitions. The process of selecting the most appropriate overloaded function or operator is called overload resolution.

**Function Overloading in C++**

When we have multiple definitions for the same function name in the same scope. The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. We cannot overload function declarations that differ only by return type.

**Operators Overloading in C++**

We can redefine or overload most of the built-in operators available in C++. Thus, a programmer can use operators with user-defined types as well.

**Syntax defined**

Overloaded operators are functions with special names: the keyword "operator" followed by the symbol for the operator being defined. Like any other function, an overloaded operator has a return type and a parameter list.

Box operator+(const Box&);

**Overload-able operators**

Following is the list of operators which can be overloaded

* +
* -
* \*
* /
* %
* ^
* &
* !
* ~
* ,
* =
* <
* >
* <=
* >=
* ++
* <<
* --
* >>
* ==
* !=
* &&
* ||
* +=
* -=
* \*=
* %=
* /=
* >>=
* <<=
* \*=
* \=
* []
* ()
* New
* New[]
* Delete
* Delete[]

**Operators which can not be overloaded**

* **:**
* **.\***
* **.**
* **?:**

**QUESTION #3:**

**C++**

**Introduction**

C++ is a cross-platform language that can be used to create high-performance applications.

C++ was developed by Bjarne Stroustrup, as an extension to the C language.

C++ gives programmers a high level of control over system resources and memory.

The language was updated 3 major times in 2011, 2014, and 2017 to C++11, C++14, and C++17.

**Applications developed using C++**

* YouTube. ...
* Amazon.com. ...
* Windows OS. ...
* Microsoft Office. ...
* MySQL. ...
* Mozilla Firefox.

**C++ Key Features**

* OOP (Object-Oriented Programming) C++ is an object-oriented language, unlike C which is a procedural language.
* Platform or Machine Independent/ Portable.
* Simple.
* High-level programming language.
* Popular.
* Case sensitive.
* Compiler-Based.
* DMA (Dynamic Memory Allocation)

**C++ Pros**

* Faster compile time
* Semi object-oriented
* Used in OS programming
* Wide Support
* Very Powerful

**C++ cons**

* Pointer can be misused in C++.
* Complicated to learn
* No automatic Garbage collection

**Java**

**Introduction**

Java is an extensively used programming language that acts as a server-side language for back-end, big data, and android development.

It is also employed in desktop computing, games, other mobile computing, and numerical computing. Moreover, Java programs can run on various platforms such as Windows, Macintosh, and Unix computers. Java web development is very famous for creating a top-notch web apps.

**Applications developed using Java**

Twitter, Cash App, Spotify, Signal, Uber, Netflix, and more are the very popular apps developed using Java.

**Java Key Features**

* Object-Oriented
* Architecture-neutral
* Multi-threaded
* Platform-independent

**Java Pros**

* Great libraries
* High Performance
* Native threads
* Excellent SDK — JDK

**Java Cons**

* Slow
* Requires significant memory space
* Java for web development is the best choice.

**.Net**

**Introduction**

.Net is an open-source platform through which one can develop various sorts of applications. With this cross-platform, one can use various languages, editors, and libraries to build for the web, mobile, desktop, games, and IoT.

.Net support multiple programming languages such as C#, C++, VB.Net, and F#.

**Applications developed using .Net**

* Accenture, Starbucks, Stack Overflow, Microsoft, and many others are the well-known brands using .Net.

**.Net Key Features**

* Common Language Runtime engine (CLR)
* Language independence
* Base Class Library
* Uncomplicated deployment

**.Net Pros**

* Stable code
* Great 3rd party libraries
* Highly productive
* Great community

**.Net Cons**

* Limited Object-Relational Support
* Stability Problems for New Releases

**Python**

**Introduction**

Python is a general-purpose programming language similar to HTML, CSS, and JavaScript. It is used for the back-end, software, and web development, data science, and writing system scripts over other things. The technology simple to learn syntax maintains readability and therefore lessens the expense of program maintenance

**Applications developed using Python**

YouTube, Google, Quora, Pinterest, Instagram, and more are very famous apps built using Python.

**Python Key Features**

* Object-Oriented Approach
* Supports GUI
* Extensive Array of Library
* Supports Multiple Languages

**Python Pros**

* Functional programming
* Simple to learn
* Dynamic typing
* Readable code

**Python Cons**

* Memory Consumption
* Runtime Errors

**Question #4**

**Android**

Android is an open source and Linux-based operating system for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies.

Android programming is based on Java programming language so if you have basic understanding on Java programming then it will be a fun to learn Android application development.

**Environment setup**

We can start Android application development on either of the following operating systems −

* Microsoft Windows XP or later version.
* Mac OS X 10.5.8 or later version with Intel chip.
* Linux including GNU C Library 2.7 or later.

Second point is that all the required tools to develop Android applications are freely available and can be downloaded from the Web. Following is the list of software's you will need before you start your Android application programming.

* Java JDK5 or later version
* Android Studio

Here last two components are optional and if you are working on Windows machine then these components make your life easy while doing Java based application development. So let us have a look how to proceed to set required environment.

**Mobile application**

A mobile application (also called a mobile app) is a type of application designed to run on a mobile device, which can be a smartphone or tablet computer. Even if apps are usually small software units with limited function, they still manage to provide users with quality services and experiences.

Contrary to applications designed for desktop computers, mobile applications move away from integrated software systems. Instead, each mobile app provides an isolated and limited functionality. For example, it can be a game, a calculator, or a mobile web browser.

Because of the limited hardware resources of the early mobile devices, mobile apps avoided multi-functionality. However, even if the devices used today are far more sophisticated, mobile apps remain narrowly functional. This is how mobile app owners allow consumers to handpick exactly the functions their devices should have.

**Mobile app development**

Mobile app development is a process that draws a lot from traditional software development. However, it’s focused on creating software that takes advantage of the unique features of mobile device hardware.

The most straightforward scenario for building a mobile app is taking a desktop-based application and importing it to a mobile device. However, as the app becomes more robust, this technique can become problematic.

A better approach involves developing specifically for the mobile environment. It’s a technique that takes advantage of all the benefits mobile devices offer. The process takes into account their limitations and helps business owners balance cost with functionality.

For example, applications that use location-based features such as maps are always built from the ground up with mobile in mind. Location-based services delivered on a desktop app make less sense because desktop users aren’t moving around.

Modern smartphones and tablets are equipped with features such as Bluetooth, Near Field Communication (NFC), GPS, gyroscopic sensors, cameras, and many more. Developers can use these features to create apps with technologies such as Virtual or Augmented Reality, barcode scanning, location-based services, and many more. The most successful and popular mobile applications use smartphone features in the best possible way.

The issue of hardware in mobile devices introduces another complication:

While developers building apps for iOS can only expect the apps to be run on two types of devices (iPhones and iPads), Android developers can’t say the same. In fact, for them, every smartphone and tablet may be running on different hardware and various versions of the operating system.

**Mobile app development technologies**

To help you understand the process of building a mobile application here’s a closer look at all the different technology considerations business owners must make before building an app.

**Native apps**

Such apps are built for a single mobile operating system. That’s why they’re called native – they’re native to a particular platform or device. The majority of mobile apps today are built for systems like Android or iOS. To put it simply, you can’t install and use an Android app on iPhone, and vice versa.

**Benefits of native app**

The main benefit of native apps is their high performance and excellent user experience. After all, developers who build them use native device UI. Access to a broad range of APIs also helps to accelerate the development work and extend the boundaries of app usage. Native applications can only be downloaded from app stores and installed directly into devices. That’s why they first need to pass a strict publishing process. 32157

**Draw Back of native apps**

The most important drawback of native apps is their cost. To build, support, and maintain an app for Android and iOS you basically need two development teams. As you can imagine, this may result in a higher price tag on the project.

**Web apps**

Web apps are software applications that behave similarly to native mobile apps and work on mobile devices. However, there are significant differences between native apps and web apps. For starters, web apps use browsers to run, and they’re usually written in CSS, HTML5, or JavaScript. Such apps redirect the user to the URL and then offer them the option to install the app. They simply create a bookmark on their page. That’s why they require minimum device memory.

**Draw back**

Since all of the personal databases will be saved on the server, users can only use the application if they have an internet connection. This is the main drawback of web apps – they always require a good internet connection. Otherwise, you risk delivering a subpar user experience.

**Benefits**

Moreover, developers don’t have that many APIs works with, except for the most popular features like geolocation. The performance will be linked to browser work and network connection as well.

**Hybrid apps**

These apps are built using web technologies such as JavaScript, CSS, and HTML 5. Why are they called hybrid? Hybrid apps basically work like web apps disguised in a native wrapper.

**Benefits**

Hybrid apps are easy and fast to develop, which is a clear benefit. You also get a single codebase for all the platforms. This lowers the cost of maintenance and streamlines the updating process. Developers can also take advantage of many APIs for features such as gyroscope or geolocation.

**Draw back**

On the other hand, hybrid applications may lack speed and performance. Also, you might experience some design issues as the app might not look the same on two or more platforms.

**Native programming languages for IOS development**

There are two native programming languages for iOS development are:

* Objective-C
* Swift

**Objective-C**

**introduction**

The Objective-C language is a simple computer language designed to enable sophisticated object-oriented programming. Objective-C is defined as a small but powerful set of extensions to the standard ANSI C language. Its additions to C are mostly based on Smalltalk, one of the first object-oriented programming languages

Objective-C is designed to give C full object-oriented programming capabilities, and to do so in a simple and straightforward way.

Objective-C is a general-purpose, object-oriented programming language that adds Smalltalk-style messaging to the C programming language. This is the main programming language used by Apple for the OS X and iOS operating systems and their respective APIs, Cocoa and Cocoa Touch

**OOp environment**

Most object-oriented development environments consist of several parts:

* An object-oriented programming language
* A library of objects
* A suite of development tools
* A runtime environment

**Swift**

**Background**

Apple’s launch this past June of Swift, a new programming language for writing iOS apps, created a great deal of buzz and excitement throughout the iOS developer community.

Since its launch, many iOS developers have been struggling with the question of if, how, and when to transition from Objective-C to Swift. The answer to that question will of course be different for every team and every project.

**Introduction**

Swift is a general-purpose, multi-paradigm, object-oriented, functional, imperative and block structured language. It is the result of the latest research on programming languages and is built using a modern approach to safety, software design patterns by Apple Inc.. It is the brand new programming language for iOS application, macOS application, watchOS application, tvOS application. Soon it became one of top 5 programming language and gained popularity among Apple developer community over the few years of time replacing the old school Objective C.

**Programming languages for android app development**

Two native programming languages for android app development are;

* JavaScript
* Kotlin

**React Native (Javascript)**

React Native is a real JavaScript framework that offers real, localized mobile applications for iOS and Android. It is based on the response, Facebook’s JavaScript library, to make the user border, but instead of targeting the browser, it targets the mobile platform. JavaScript is also known as the cockroach of the Information technology industry. It uses a similar UI mechanism as native iOS or Android apps. Feedback applications are as free as any other native app.

It is considered to be the best framework for mobile app development for both native OS. This means that the JavaScript language is used to create a cross-platform app that is developed that runs on different platforms instead of creating separate apps for each system. It expresses the ability to reuse native presentation code and help build faster and cheaper applications Cross-platform development helps save time and coding heads.

2. **Kotlin**

Kotlin is an innovative and stable typing programming language that enhances productivity and enhances developer happiness. Kotlin can be used in conjunction with Java to create more efficient and high-performance apps. Kotlin has very clean syntax and easy code. It is varied and versatile. It produces compact and cleaner code than Java. The modern features of the Kotlin language allow you to focus on both expressing your ideas and writing less code. Less code is written to test and maintain less code. Overall, this is considered a clean and tidy version of Java.

**Best programming language for Android games**

Java is the best language for Android games. Some people code in C ++, but then have trouble dealing with cross-device compatibility. As mentioned earlier, since Java runs on virtual machines, you don’t have to jump through hoops to sync Java games to different devices. So, we can say that the java language is best for games development.

**Top 3 frameworks for hybrid app development**

Top 3 frameworks for hybrid app development are:

* React Native
* Xamarin
* Ionic

**React Native**

React Native is one of the top cross-platform mobile development frameworks. It is based on React — a best-in-class JavaScript library for creating user interfaces — and targets mobile platforms.

The great thing about React Native is that you can use a universal JavaScript language as well as Java, Swift, or C — languages specific for native platforms to build cross-platform solutions. Thus, this framework can convert the source code into native elements, contributing to native user experience.

Web developers can easily use React Native to create mobile apps without obtaining new programming skills

**Xamarin**

Xamarin is a leading free, open-source platform for building hybrid mobile apps. Xamarin-based solutions are compatible with any mobile platform and provide the same performance and user experience as native solutions.

This popular Microsoft cross-platform mobile app framework relies on the C# programming language complemented with .NET framework and is available under the MIT license as part of Visual Studio. Its development toolset includes the following components:

* Xamarin Studio/Visual Studio (its own IDE)
* Xamarin SDKs
* Xamarin Test Cloud
* Xamarin.Mac to build native Mac apps

**Ionic**

ionic is definitely among the best hybrid app frameworks as it bridges the properties that are in demand: it’s open-source, easily maintained, scalable, and easy to read. Ionic provides a library of HTML, CSS and JS components optimized for mobile development that helps build interactive apps.

Ionic has a rich set of features and tools, including out-of-the-box support for material design, native-style mobile UI elements and layouts. It also provides its customers with regular updates and resources from the development community.

Ionic-based hybrid applications possess native functionalities, specific gestures and customizable tools that enhance user-friendliness. However, they rely on wrappers such as Cordova or PhoneGap to run as native.