Essay

First of what is Kotlin?

Kotlin is a statically typed programming language that runs on the Java Virtual Machine (JVM). It was developed by JetBrains, the company behind IntelliJ IDEA, and officially announced in 2011. Since its inception, Kotlin has gained significant traction in the software development community due to its concise syntax, interoperability with Java, and strong tooling support.

Here are some common uses of Kotlin

Android App Development\*\*: Kotlin has become an officially supported language for Android development by Google. Many Android developers have embraced Kotlin due to its modern features, which can help improve productivity and code readability compared to Java. Web Development: Kotlin can be used for web development, both on the frontend and backend. Kotlin can be compiled to JavaScript using tools like Kotlin/JS, allowing developers to write frontend code in Kotlin. Additionally, Kotlin can be used with frameworks like Ktor and Spring Boot to build backend services for web applications. Data Science and Machine Learning: Kotlin’s interoperability with Java libraries makes it suitable for data science and machine learning tasks. Developers can leverage Java libraries like Deeplearning4j or Weka in Kotlin projects, allowing them to take advantage of existing Java ecosystem tools and libraries.

Then there is Scripting: Kotlin can also be used for scripting tasks, similar to other scripting languages like Python or Ruby. Its concise syntax and strong type system make it a viable option for writing scripts for automation, tooling, or command-line utilities.Education: Kotlin is increasingly being used in educational settings to teach programming due to its modern features and ease of learning. Its syntax is less verbose compared to Java, making it more approachable for beginners while still providing a strong foundation in object-oriented programming principles.

In my assignment I was given instruction to create an app in which every time I enter a age between 20-100 it is supposed to tell me which historical figure had passed on in between the ages of 20-100

In Kotlin, factors can be announced utilizing either the 'val' catchphrase or the 'var' watchword. The choice between 'val' and 'var' decides whether the variable is permanent (consistent) or changeable.

A computer screen with text on it

Description automatically generated

Having a likable user interface (UI) is crucial for several reasons like having User Engagement to have A likable UI can enhance user engagement by providing an enjoyable and satisfying user experience. When users find an interface visually appealing and easy to use, they are more likely to spend more time interacting with the application or website, User Satisfaction for a well-designed UI contributes to overall user satisfaction. When users enjoy interacting with an interface, they are more likely to have positive perceptions of the product or service. This can lead to increased user loyalty and retention.

Increased Conversions: A likable UI can lead to higher conversion rates, whether the goal is to sell products, collect leads, or encourage user engagement. When users feel positively about an interface, they are more likely to take desired actions and convert into customers or subscribers.

A computer screen with text on it

Description automatically generated

The importance of using variables for your code is to factors pronounced utilizing val are permanent, meaning their esteem cannot be changed once it's doled out .It is comparable to pronouncing a 'final' variable in Java. Once initialized, you cannot reassign a modern esteem to a val.

Where as var factors announced utilizing var are changeable, meaning their esteem can be changed after initialization.

It's practically equivalent to to pronouncing a normal variable in Java.

Choosing between val and var depends on whether you would like the variable's esteem to be steady or mutable.

Utilize Val after you know the esteem won't alter after initialization. This may move forward code coherence and security, as you're sure the value won't be inadvertently adjusted somewhere else within the code.

Utilize var once you have to be reassigned the variable's esteem afterward within the program. This gives adaptability but requires additional caution to guarantee that the variable is overhauled suitably without causing unintended side impacts.

And as you can see I have utilized val and var to assign them function to the button, edit text and the text view

In KOTLIN when working with Android improvement, you regularly allot IDs to UI components such as buttons, Edit Text , and Text Views i. These IDs are at that point referenced in your Kotlin code to associated with these UI components programmatically.

Such as using the Find view by Id and then we're setting a click listener for the button. When the button is clicked, we retrieve the text entered in the Edit Text view and update the Text View accordingly. Using if-else statements when handling inputs in a program is crucial for controlling the flow of execution based on different conditions. Here's why it's important and how it's commonly used: Conditional Logic: if-else statements allow you to execute different blocks of code depending on whether a condition is true or false. This is essential when you need your program to react differently based on the input provided. Error Handling: When dealing with user inputs, it's common to validate them to ensure they meet certain criteria. if-else statements can be used to check if the

A computer screen with text on it

Description automatically generated

input is within acceptable ranges or if it meets certain conditions. If the input doesn't meet the requirements, you can provide appropriate error messages or take corrective actions.

In Menu Selection many applications, users are presented with menus or options to choose from. if-else statements can be used to determine which option the user has selected and execute the corresponding functionality. State Management: Inputs can sometimes change the state of the program or object. if-else statements are useful for checking the current state and deciding what actions to take based on that state. Algorithmic Decisions: Algorithms often require making decisions based on input data. if-else statements are fundamental in implementing these decision-making processes within the algorithm.

then using User Interfaces for if-else statements are frequently used to control the visibility, enabled state, or behavior of UI elements based on user inputs or other conditions. In this case I used it to list the Historical Figures that were needed as an input for the app. The use of the input Number was to restrict ages below or beyond 20-100.