Câu 1: Hãy cho biết các nền tảng cho thiết bị di động thông minh hiện nay? Với mỗi nền tảng hãy cho biết đặc điểm, ưu và khuyết điểm.

- Nền tảng: Android, IOS, Window phone.

- Ưu điểm:
  + Android là hệ điều hành mã nguồn mở, có thể tùy chỉnh theo nhu cầu. Chạy trên nhiều loại thiết bị. Người dùng có thể tùy chỉnh giao diện, cài đặt ứng dụng từ nguồn ngoài, thay đổi launcher, và sử dụng nhiều tiện ích mở rộng khác.

  + IOS được tối ưu hóa cho phần cứng của Apple, giúp các thiết bị iPhone và iPad có khả năng hoạt động mượt mà, ít xảy ra hiện tượng giật, lag. iOS tích hợp sâu với các sản phẩm và dịch vụ khác của Apple như MacBook, Apple Watch, Apple TV. Người dùng có thể chuyển đổi và đồng bộ dữ liệu dễ dàng giữa các thiết bị. Bảo mật cao, thường xuyên cập nhật và vá các lỗ hổng bảo mật. iOS cũng có các tính năng như mã hóa dữ liệu. Cập nhật thường xuyên, ngay cả đối với mẫu cũ. Chất lượng cao, giảm thiểu các ứng dụng có mã độc, khó hack.

- Nhược điểm:
  + Android bảo mật không cao. Có nhiều phiên bản Android khác nhau do các nhà sản xuất tùy biến, dẫn đến tình trạng phân mảnh, khó tối ưu trải nghiệm người dùng. Sau khoảng 4 năm sẽ không còn được cập nhật phiên bản.

  + IOS giá thành cao. Khả năng tùy biến thấp. Hạn chế hệ sinh thái bên ngoài.

Câu 2: Liệt kê các nền tảng phát triển ứng dụng di động phổ biến hiện nay và so sánh sự khác biệt chính giữa chúng.

Nền tảng	Ngôn ngữ	Hiệu năng	Tính năng hệ thống	Độ phức tạp phát triển	Tính đa nền tảng
Native	Java/Kotlin, Swift	Cao	Tốt nhất	Cao	Không
React Native	JavaScript	Tương đối cao	Tương đối tốt	Trung bình	Có
Flutter	Dart	Cao gần native	Tốt	Trung bình	Có
Xamarin	C#	Tương đối cao	Tương đối tốt	Trung bình	Có
Ionic	JavaScript, HTML, CSS	Trung bình	Hạn chế	Dễ	Có
Cordova	JavaScript, HTML, CSS	Thấp	Hạn chế	Dễ	Có

Câu 3: Điều gì làm cho Flutter trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng đa nền tảng? So sánh với các nền tảng khác như React Native và Xamarin.

- Lý do khiến cho Flutter trở thành một lựa chọn phổ biến:
    - Flutter sử dụng Dart và tạo giao diện đồ họa (UI) trực tiếp với công cụ đồ họa Skia. Thay vì phụ thuộc vào cầu nối như React Native hay Xamarin để liên kết với các thành phần giao diện hệ thống, Flutter có thể vẽ mọi yếu tố giao diện trên nền tảng riêng. Điều này giúp tối ưu hóa hiệu năng, khiến ứng dụng Flutter đạt hiệu năng gần như tương đương với ứng dụng native.
    - Flutter sử dụng một hệ thống widget tùy chỉnh, cho phép xây dựng giao diện đẹp mắt và đồng nhất trên mọi nền tảng. Các nhà phát triển có thể dễ dàng tái tạo giao diện phức tạp, với sự linh hoạt cao và dễ kiểm soát chi tiết về thiết kế.
    - Flutter hỗ trợ phát triển đa nền tảng (iOS, Android, web, và desktop) từ một mã nguồn duy nhất. Điều này giúp giảm thiểu công sức bảo trì mã nguồn, tối ưu hóa thời gian và chi phí phát triển cho các đội ngũ nhỏ hoặc dự án có ngân sách hạn chế.
    - Vì Flutter kiểm soát toàn bộ giao diện qua widget, các ứng dụng Flutter giữ được tính nhất quán cao về giao diện và trải nghiệm người dùng trên mọi nền tảng.
    - Hot Reload là tính năng nổi bật của Flutter, cho phép các thay đổi trong mã nguồn được áp dụng ngay lập tức trên giao diện mà không cần khởi động lại ứng dụng. Điều này giúp tăng hiệu suất làm việc và dễ dàng kiểm tra, chỉnh sửa giao diện.
    - Flutter có sự hậu thuẫn mạnh mẽ từ Google, cùng với cộng đồng ngày càng phát triển, cung cấp nhiều thư viện và tài liệu phong phú. Điều này giúp các nhà phát triển dễ dàng tìm kiếm sự hỗ trợ và tài liệu cần thiết.
- So sánh Flutter với React Native và Xamarin:

Tính năng	Flutter	React Native	Xamarin
Ngôn ngữ	Dart	JavaScript	C#
Hiệu năng	Gần như native, rất tối ưu	Tốt nhưng phụ thuộc cầu nối	Khá tốt, nhưng kém hơn native
Kiến trúc UI	Widget riêng, độc lập với hệ thống	Dùng thành phần native	Thành phần native
Hot Reload	Có, rất mượt mà	Có, nhưng đôi khi không ổn định	Có, nhưng không tốt như Flutter

Hỗ trợ đa nền tảng	iOS, Android, web, desktop	iOS, Android, một phần web	iOS, Android, Windows
Cộng đồng	Rất lớn, được Google hỗ trợ	Rất lớn, Facebook hỗ trợ	Khá lớn, nhưng tập trung vào Microsoft
Giao diện	Nhất quán nhờ hệ thống widget riêng	Sử dụng thành phần hệ điều hành	Nhất quán cho iOS và Android
Hỗ trợ tính năng native	Có nhiều plugin, dễ tùy chỉnh	Dùng cầu nối, nhiều plugin	Hỗ trợ tốt các API native
Độ khó học	Trung bình (Dart mới mẻ hơn)	Dễ (JavaScript phổ biến)	Khá cao (C# và môi trường Microsoft)

Câu 4: Liệt kê các ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng trên Android và giải thích tại sao chúng lại được chọn.

- **Java**: là ngôn ngữ đầu tiên được Google sử dụng để phát triển ứng dụng Android từ khi hệ điều hành này ra mắt. Là một ngôn ngữ hướng đối tượng, dễ học và có cú pháp rõ ràng, giúp dễ dàng phát triển và bảo trì ứng dụng. Là một ngôn ngữ phổ biến với cộng đồng lớn, nhiều tài liệu và thư viện hỗ trợ, giúp các nhà phát triển dễ dàng tìm kiếm tài nguyên và giải pháp khi gặp vấn đề. Tương thích tốt với nhiều nền tảng và thiết bị, giúp dễ dàng phát triển ứng dụng cho đa dạng thiết bị Android.
- **Kotlin:** Năm 2017, Google đã công bố Kotlin là ngôn ngữ chính thức cho Android. Kotlin có khả năng thay thế hoàn toàn Java và là ngôn ngữ mặc định cho các dự án Android mới. Kotlin có cú pháp ngắn gọn, giảm bớt các đoạn mã rườm rà, giúp tăng tốc độ phát triển và giảm lỗi lập trình. Kotlin có các tính năng như Null Safety giúp tránh lỗi phổ biến liên quan đến giá trị null, tăng độ an toàn và độ ổn định của ứng dụng. Kotlin tương thích hoàn toàn với Java, cho phép các nhà phát triển chuyển đổi dần từ Java sang Kotlin hoặc sử dụng cả hai ngôn ngữ trong cùng một dự án.
- **C++:** là một ngôn ngữ bậc thấp, tối ưu cho các tác vụ yêu cầu hiệu năng cao như xử lý đồ họa, âm thanh, và tính toán phức tạp. Các ứng dụng yêu cầu sử dụng CPU và bộ nhớ nhiều, chẳng hạn như trò chơi và các ứng dụng đa phương tiện phức tạp, có thể hưởng lợi từ hiệu năng của C++. Android cung cấp NDK cho phép các nhà phát triển sử dụng C++

để xây dựng phần mềm, đặc biệt là các thư viện native cho những tác vụ đòi hỏi hiệu năng cao.

- **Dart (Flutter):** Dart là ngôn ngữ chính thức của Flutter, một framework của Google cho phép phát triển ứng dụng chạy trên cả Android, iOS, và các nền tảng khác từ một mã nguồn duy nhất. Dart có tính năng Hot Reload, giúp các nhà phát triển thay đổi mã nguồn và thấy ngay kết quả trên giao diện mà không cần khởi động lại ứng dụng, giúp tăng tốc độ phát triển. Dart và Flutter cung cấp các widget tùy chỉnh giúp thiết kế giao diện nhất quán và đẹp mắt trên nhiều nền tảng, phù hợp cho các ứng dụng có yêu cầu cao về giao diện người dùng.
- **Python (thông qua Kivy hoặc BeeWare):** Python có cú pháp rõ ràng, dễ học, giúp các nhà phát triển mới bắt đầu dễ dàng tiếp cận lập trình ứng dụng. Với Kivy hoặc BeeWare, Python có thể được dùng để phát triển ứng dụng đa nền tảng, bao gồm Android. Các công cụ này giúp Python trở thành lựa chọn tốt cho các ứng dụng nhỏ và dự án phát triển nhanh. Python có một hệ sinh thái thư viện phong phú, đặc biệt trong các lĩnh vực như trí tuệ nhân tạo và xử lý dữ liệu, giúp ích khi cần tích hợp các tính năng phức tạp trong ứng dụng Android.
- **JavaScript (React Native, Apache Cordova):** Các framework như React Native và Apache Cordova cho phép phát triển ứng dụng từ một mã nguồn duy nhất, chạy trên cả Android và iOS. JavaScript là ngôn ngữ phổ biến và quen thuộc với nhiều nhà phát triển web, giúp họ dễ dàng chuyển sang phát triển ứng dụng di động mà không cần học thêm ngôn ngữ mới. Các framework JavaScript hỗ trợ nhiều thư viện và công cụ giúp phát triển ứng dụng một cách nhanh chóng, phù hợp cho các dự án cần triển khai trong thời gian ngắn hoặc có ngân sách hạn chế.

Câu 5: Liệt kê các ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng trên iOS.

- **Swift:** được Apple giới thiệu vào năm 2014 như một ngôn ngữ thay thế hiện đại cho Objective-C, và nhanh chóng trở thành ngôn ngữ chính cho phát triển iOS. Swift được tối ưu hóa để có hiệu năng gần như tương đương với C++ và vượt trội so với nhiều ngôn ngữ cấp cao khác. Điều này giúp Swift phù hợp cho các ứng dụng có yêu cầu xử lý cao. Swift có cú pháp ngắn gọn, rõ ràng, và dễ học hơn so với Objective-C. Nó giúp các nhà phát triển viết mã nhanh chóng, an toàn và ít lỗi hơn. Swift có các tính năng như Null Safety và các công cụ kiểm tra lỗi mạnh mẽ, giúp giảm thiểu các lỗi phổ biến liên quan đến Null và bảo mật mã nguồn.
- **Objective-C:** Objective-C là ngôn ngữ ban đầu để phát triển ứng dụng iOS và vẫn được hỗ trợ đầy đủ bởi Apple. Nhiều ứng dụng và thư viện lớn cũ của iOS vẫn sử dụng Objective-C. Objective-C có khả năng tương thích và tích hợp tốt với mã nguồn C và C++, giúp các nhà phát triển có thể sử dụng các thư viện C/C++ trong ứng dụng iOS. Vì lịch sử phát triển lâu dài, Objective-C có một kho tài liệu và thư viện phong phú. Các nhà phát triển lâu năm trên nền tảng Apple vẫn ưa thích sử dụng Objective-C cho các dự án lớn. Dù Swift đang dần thay thế, Objective-C vẫn là một lựa chọn ổn định cho các dự án cần duy trì và cập nhật mã cũ.
- **C++:** C++ là một ngôn ngữ bậc thấp, tối ưu cho các tác vụ cần xử lý nặng như game, xử lý đồ họa, và các ứng dụng khoa học. iOS cho phép sử dụng C++ thông qua Objective-

C++ (phiên bản của Objective-C hỗ trợ mã C++), giúp các nhà phát triển tận dụng được các thư viện và mã C++ có sẵn. Khi phát triển game hoặc các ứng dụng cần hiệu năng cao chạy trên nhiều nền tảng, C++ là lựa chọn tốt để chia sẻ mã nguồn giữa các hệ điều hành.

- **Dart (Flutter):** Flutter, framework của Google sử dụng Dart, cho phép phát triển ứng dụng một mã nguồn duy nhất chạy trên cả iOS, Android, web và desktop. Điều này giúp tiết kiệm chi phí và thời gian khi phát triển đa nền tảng. Dart và Flutter có tính năng Hot Reload, giúp các nhà phát triển thấy ngay thay đổi trên giao diện mà không cần khởi động lại ứng dụng, tăng tốc độ phát triển. Flutter cung cấp hệ thống widget tùy chỉnh, giúp tạo ra giao diện nhất quán trên mọi nền tảng.
- **JavaScript (React Native):** React Native là framework JavaScript cho phép phát triển ứng dụng iOS và Android từ một mã nguồn chung, giúp tiết kiệm thời gian và tài nguyên phát triển. JavaScript là một ngôn ngữ phổ biến, dễ học, và thân thiện với nhiều nhà phát triển web, giúp họ chuyển sang phát triển ứng dụng di động dễ dàng. Với hệ sinh thái thư viện phong phú, React Native hỗ trợ nhiều tính năng, giúp xây dựng ứng dụng phức tạp mà không cần phải code từ đầu.
- **Python (Kivy hoặc BeeWare):** Python có cú pháp đơn giản và dễ hiểu, thích hợp cho các nhà phát triển muốn xây dựng nhanh các ứng dụng cơ bản. Thông qua các framework như Kivy hoặc BeeWare, Python có thể được dùng để phát triển ứng dụng đa nền tảng, bao gồm iOS. Python có nhiều thư viện hỗ trợ các lĩnh vực chuyên sâu như trí tuệ nhân tạo, phân tích dữ liệu, giúp ích khi cần tích hợp các tính năng này vào ứng dụng.

Câu 6: Hãy thảo luận về những thách thức mà Windows Phone đã phải đối mặt và nguyên nhân dẫn đến sự sụt giảm thị phần của nó.

- **Thiếu ứng dụng phổ biến**: Một trong những vấn đề lớn nhất của Windows Phone là kho ứng dụng nghèo nàn. Trong khi iOS và Android có hàng triệu ứng dụng, Windows Store lại thiếu các ứng dụng phổ biến và cập nhật thường xuyên. Nhiều nhà phát triển không mặn mà với việc phát triển ứng dụng cho nền tảng này do số lượng người dùng hạn chế.
- **Thiếu sự hỗ trợ từ nhà phát triển**: Các công ty và nhà phát triển phần mềm thường ưu tiên xây dựng ứng dụng cho iOS và Android trước, do đó Windows Phone ít khi được ưu tiên. Nhiều ứng dụng quan trọng như Snapchat, Instagram, và một số dịch vụ ngân hàng thậm chí không có mặt trên Windows Phone, làm giảm sức hấp dẫn với người dùng.
- **Sự không nhất quán trong chiến lược phát triển**: Microsoft đã thay đổi chiến lược và tầm nhìn của Windows Phone qua từng năm, gây nhầm lẫn cho người dùng và nhà phát triển. Từ việc chuyển từ Windows Phone 7 sang 8 và sau đó là Windows 10 Mobile, mỗi phiên bản đều yêu cầu các ứng dụng được viết lại, tạo ra gánh nặng cho các nhà phát triển.
- **Cạnh tranh với Android và iOS**: Microsoft đã khó khăn trong việc xác định rõ điểm khác biệt của Windows Phone so với iOS và Android. Dù giao diện “Live Tiles” là một ý tưởng độc đáo, nó không đủ để thu hút người dùng từ các nền tảng đã quen thuộc và ổn định hơn.

Câu 7: Khám phá các ngôn ngữ và công cụ để phát triển ứng dụng web trên thiết bị di động.

 React và **React Native**:

- **React**: Framework JavaScript được Facebook phát triển, nổi tiếng với tính năng **component-based** và **Virtual DOM** giúp phát triển giao diện nhanh chóng.
- **React Native**: Dựa trên React, cho phép viết ứng dụng đa nền tảng (iOS, Android) sử dụng cùng một mã nguồn JavaScript. Điều này giúp tận dụng kiến thức về React và phát triển ứng dụng di động gốc.

 Vue.js và **Quasar Framework**:

- **Vue.js**: Một framework JavaScript nhẹ và linh hoạt, phổ biến nhờ cú pháp đơn giản, dễ học và cộng đồng lớn. Vue.js thường dùng để phát triển các ứng dụng đơn trang (SPA).
- **Quasar Framework**: Dựa trên Vue.js, giúp phát triển các ứng dụng đa nền tảng như SPA, PWA, ứng dụng desktop và mobile.

 Angular và **Ionic**:

- **Angular**: Framework do Google phát triển, có kiến trúc **MVC (Model-View-Controller)**, được sử dụng nhiều cho ứng dụng web doanh nghiệp và các ứng dụng lớn.
- **Ionic**: Framework dựa trên Angular, hỗ trợ phát triển ứng dụng đa nền tảng. Ionic cho phép viết một mã nguồn duy nhất, sau đó biên dịch ra ứng dụng cho iOS, Android, và cả web.

 Flutter:

- **Dart**: Flutter sử dụng ngôn ngữ Dart của Google để viết mã. Nó cho phép phát triển ứng dụng di động nhanh chóng với tính năng **Hot Reload** và giao diện thống nhất trên cả iOS và Android.
- **Flutter Web**: Gần đây, Flutter cũng hỗ trợ phát triển ứng dụng web, giúp mở rộng khả năng đa nền tảng của framework.

 Apache Cordova (PhoneGap):

- Cordova là công cụ giúp gói các ứng dụng web vào trong một trình duyệt riêng, cho phép chạy các ứng dụng này trên iOS, Android như một ứng dụng native.
- **PhoneGap** (một biến thể của Cordova) giúp tiếp cận các API phần cứng của điện thoại như camera, định vị, và cảm biến.

 Progressive Web Apps (PWA):

- **PWA** là các ứng dụng web được thiết kế để hoạt động như một ứng dụng gốc trên thiết bị di động. Chúng có thể được thêm vào màn hình chính, chạy ngoại tuyến, và có tính năng giống như ứng dụng native.
- **Công cụ hỗ trợ PWA**: Các công cụ như **Workbox** (thư viện của Google) giúp dễ dàng cài đặt và quản lý các tính năng ngoại tuyến và bộ nhớ cache.

Câu 8: Nghiên cứu về nhu cầu nguồn nhân lực lập trình viên trên thiết bị di động hiện nay và những kỹ năng được yêu cầu nhiều nhất.

+ Thu nhập của Junior di động trong khoảng: 10-18 triệu.

+ Thu nhập của Mid-level di động trong khoảng: 18-35 triệu.

+ Thu nhập của Senior di động trong khoảng: 35-60 triệu.

Kỹ năng lập trình:

- Native Android (Java/Kotlin)
- Native iOS (Swift/Objective-C)
- Cross-platform (React Native, Flutter)
- Backend integration (RESTful APIs, GraphQL)
- Cơ sở dữ liệu mobile (SQLite, Realm)