課題５　概念論述

1. オブジェクト指向とは何かを述べてください

a. 特徴３つ、また、その説明を含めてください

b. 具体例を含めてください

　オブジェクト指向とは「ある役割を持ったモノ」ごとにクラス（プログラム全体の設計図）を分割し、モノとモノとの関係性を定義していくことでシステムを作り上げようとするシステム構成の考え方のこと。
　オブジェクト：直訳すれば「物」「対象」という意味。プログラミングにおいてはデータと処理の集まりを意味している。
　クラス：データと処理をひとつにまとめる機能。モノ（オブジェクト）の設計書のようなもの。
　プロパティ：モノ（オブジェクト）が持っているデータのこと。（車に例えれば「メーカー」「排気量」「色」など）
　メソッド：モノ（オブジェクト）が持っている処理のこと。（車に例えれば「走る」「曲がる」「止まる」などアクションを起こす処理のこと）
　オブジェクト指向３つの特徴：
●**継承**＝同じようなプログラムを１か所にまとめてコードを再利用しやすくする仕組み
似たようなオブジェクトを複数作る時に、全てのプロパティやメソッドをいちいちプログラミングするのは非常に手間が掛かりますが、継承を使うことにより、同じ機能を実装できます。

ゲーム作成に例えると、「車」に「トラック」も追加しようとした場合。基本操作は同じため追加も簡単にできるのです。

●**カプセル化**＝他のプログラムからできるだけ変更できない仕組み
オブジェクトが持つデータや処理のうち、別のオブジェクトから直接利用される必要のないものを隠すことを言い、利用する場合は外部から操作するために作られた処理を設けることを言います。
プログラムが壊れにくくなると共に、大人数で開発をするときすべてのコードを認識する必要がなくなります。

●**ポリモーフィズム**＝継承したコードの一部を変更して利用するための仕組み
クラスによって同一のメソッドで異なる処理が行えるという性質をいいます。世の中の家電は説明書を見なくてもだいたい使い方がわかりますよね。
車においてもアクセルが右というふうに決まっています。

プログラムも同じ処理の名前で動いてくれると、処理名を覚える必要もないし、ミスも減らすことが出来るのです。

2. Github flow とは何かを述べてください

a. 下記の文言を必ず含めてください

i. リポジトリ

ii. main

iii. リモート

iv. ブランチ

Github とは Git におけるリポジトリの分岐モデルであり、ルールのことを指します。
それぞれのブランチを明確に定義し、複数人での開発時にそれぞれが好き勝手にブランチを作成し混乱することを防ぎます。 プロジェクト開発するとき、master の
mian ブランチがあり、開発用の develop ブランチ、そして機能毎の feature ブランチや緊急用の hotfix ブランチが存在し、これらのブランチを使い分けて開発・リリースを進めていきます。
GitHub Flow の開発の流れ
1. fork で開発リポジトリを自分のリモートリポジトリにコピーしてきます。
2. master の main ブランチから、作業ブランチを切ります。
3. ローカルで開発、コミットして Fork したブランチへ Push します
4. 本家リポジトリへ PR を出し、フィードバックを受けながら開発を進めます
5. レビュワーに承認されたら、本家リポジトリへマージします
6. デプロイ

3.サーバーサイドエンジニア・フロントエンジニアとはどのような違いがあるかを述べてください。

サーバーサイドというのは、インターネット上でサーバー側から行う処理に対して必要なプログラムの設計・開発に携わり、サーバーで扱うデータ管理を専門的に行うエンジニアのことを指します。Web 上で扱われるアプリでも、主にサーバーで行う処理の開発・状態の保存や維持をなすエンジニアを、サーバーサイドエンジニアと言います。これに対して、Web ページの設計や画面上の動きなどユーザーが直接目にする領域は、フロントエンドエンジニアが担当しています。

4.AWS とは何ですか。特徴を述べてください。

AWS とは、クラウドコンピューティングを使ったサービスです。
AWS とは Amazon Web Services の略で、Amazon が提供している 100 以上のクラウドコンピューティングサービスの総称です。
クラウドコンピューティングとは、インターネットを介してサーバー・ストレージ・データベース・ソフトウェアといったコンピューターを使った様々なサービスを利用することを指します。クラウドコンピューティングでは、手元に 1 台の PC とインターネットに接続できる環境さえあれば、サーバーや大容量のストレージ、高速なデータベースなどを必要な分だけ利用できるわけです。
AWS の特徴

１. 低コスト

初期費用や運用コストはサービスを利用する上で大きな判断基準になります。利用したサービス以上の料金を支払う必要がないだけでなく、サービス利用の手続きが単純なため、サービスの利用に対する判断を迅速に行えます。判断次第でコストダウンが可能です。

2.サービスの豊富さ

AWS は、IoT やデータ解析といった、IT インフラ全般でサービスを提供しており、その数は 200 にも上ります。IaaS や PaaS から SaaS・BaaS・FaaS のようなコンテンツをはじめとして、多種多様なラインナップが用意されています。

3.拡張性

AWS は、サービスの拡張性にも優れています。一般的なサービスでは、システムの拡張や縮小をする場合、担当者との交渉や契約手続きが必要です。AWS では、画面上の操作だけで設定変更ができます。これにより、突発的なイベントへの対応がスムーズになります。イベント期間だけ、サーバーを増やす、サービスを追加するといったことが可能です。

4.セキュリティ

AWS は、ハードウェアをセキュリティに係るコンプライアンス要件に準拠したデータセンターで運営しています。AWS のサービスだけなく、連携された外部システムのセキュリティに対しても、サポートする仕組みがあります。最適なセキュリティ対策が可能です。

5. Docker とは具体的に何ができる技術ですか。また Docker を導入するメリットを述べてください。

Docker とは、Docker 社（旧 dotCloud）が開発するコンテナのアプリケーション実行環境を管理するオープンソースソフトウェア（OSS）です。2013 年にオープンソースのプロジェクトとして公開されました。

コンテナは、実行環境を他のプロセスから隔離し、その中でアプリケーションを動作させる技術です。コンテナが利用するリソースは他のプロセスやコンテナから隔離されています。そのため、コンテナに構築されたアプリは独立したコンピュータでアプリが動作しているように見えます。

コンテナを用いることで、異なるサーバでも、同じ構成の環境を簡単に構築することができます。PC 全体を仮想化する仮想マシンとよく比べられますが、仮想マシンよりも軽量で高速に動作し、実行に必要なリソースも少なくて済みます。

Docker は、コンテナに含まれるアプリケーションをパッケージ化して実行する機能、コンテナを管理するためのツールとプラットフォームを備えています。

Docker には次の 3 つのメリットがあります。

開発ライフサイクルの改善に強い

Docker を使用すると、開発者はコンテナを利用して標準化された環境で作業できるようになります。その結果、開発ライフサイクルが改善され、CI/CD（継続的インテグレーション／継続的デリバリー）につながります。

開発環境が簡単に用意でき、かつ本番環境と共通化できる

Docker は「Docker Registry」「Docker Export／Import」という機能を有しています。これは、コンテナの元となる Docker イメージを異なるホスト間で共有する機能です。例えばチームで開発をする際、開発用マシンで作成した Docker イメージを他のメンバーのマシンに簡単にコピーできます。これにより、アプリケーションの移植性や相互運用性を高めることができます。

アプリケーション実行環境を高速にデプロイできる

コンテナは、Docker を実行するホスト上で他のアプリケーションと同じプロセス単位で管理されます。そのためコンテナでは、ハイパーバイザー型の仮想マシンでいうところの OS のブート処理が不要です。より少ないリソースでより多くのことができるため、アプリケーションの実行環境を高速にデプロイすることが可能になります。