下記 5 点、すべてにお答えください。

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 （即物的に言えば） データとそれに関わるメソッドを クラス(class) にまとめてプログラミングすることです。

オブジェクト指向言語がもつ代表的な 3 つの特徴のことで、「カプセル化、継承、ポリモーフィズム」の 3 つ。

カプセル化

カプセル化とは，オブジェクトのメンバーを保護することで，オブジェクトの安全性を高めることができる。修飾子でアクセス範囲を制限することで勝手に値に触れることを防止することができる。また，値を操作するための手段となるメソッドを用意することで，決められた手段以外のアクセスを防ぐことができる．さらに，操作するためのメソッドにおいて入出力の値を管理することにより，不正な値が入ってくることを防ぐことや，ふさわしくない出力が生じないようにすることができる。

カプセル化により，そのオブジェクトを設計者が想定した通りに機能するようにすることができるようになる．これは「ユーザーが使うのに必要な部分だけ可視化してそれ以外は隠す」ということです。プログラミング言語的に言えば public（公開） と private（非公開） を適切に設定することです。

この性質を理解し、影響範囲の小さく疎なプログラミングを心がけることが、大規模で安全なシステム構築に繋がる。

```
public class Calculator{
    /** ボタン「1」はpublic（公開） */
    public void button1(){
        //処理
    }
    /** ボタン「2」はpublic（公開） */
    public void button2(){
        //処理
    }
    /** 内部で行う計算処理はprivate（非公開） */
    private double calculate(){
        //処理
    }

    /** 結果の表示はpublic（公開） */
    public double showResult(){
        //処理
    }
}
```

継承

「継承」はプログラムの再利用性を高める考え方であり、特定のオブジェクトの機能を引き継いで使用することです。

「継承」を行わないと同じようなプログラムを複数作成後、1 つのプログラムに修正などを行った時に他のプログラムも同じように修正などを行うことになります。同じようなプログラムを 1 つにまとめて、1 つにまとめたプログラムから継承し再利用することで、修正箇所があった場合 1 つにまとめたプログラムだけの修正を行うだけで済みます。

さらに具体的にいうと「継承」はクラス定義の共通部分を別クラスにまとめる仕組みです。

実際にプログラム開発中に継承を行う時「スーパークラス」と「サブクラス」という言葉がでてきます。継承されるクラスを「スーパークラス」といい、継承し新しく作成したクラスを「サブクラス」といいます。「サブクラス」では「スーパークラス」の変数定義やメソッドを使用することができます。

そのため、「継承」はコードの再利用、拡張性を高めたり、同じような機能を持つ重複したコードを記述しなくてもよくなるため、トラブルを防止することが可能となります。

```
class User:
    def __init__(self, name):
        self.name = name

    def hello(self):
        print("Hello " + self.name)
```

```
class SuperUser(User): # ユーザークラスを継承するには、()の中にUserクラスを書けばOKです。
    def __init__(self, name, age):
        # super()を使うことで親クラスのメソッドを呼び出すことができます。
        super().__init__(name) # こクラスから親クラスのコンストラクタを呼び出す
        self.age = age
```

ポリモーフィズム(多態性)
オーバライドやオーバロードによって、メソッドを状況によって使い分けることができること
※ オーバライド: 親クラスのメソッドを子クラスで上書きすること
※ オーバロード: 同一クラス内で引数の違いによってメソッドの呼び分けること

```
// Person型変数に代入可
Person westernPerson = new WesternPerson("Albert", "Einstein", 26);
Person easternPerson = new EasternPerson("信長", "織田", 47);
```

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

① main ブランチは、常時デプロイ可能な状態を保つ
② main ブランチから作業内容が分かるようなブランチ名をつける
③ ブランチを作成したらファイルを変更し、作成したローカルリポジトリのブランチにコミットする
④ コミットができたら作成したブランチと同じ名前のブランチを GitHub へプッシュします
⑤ GitHub へプッシュできたらプルリクエストを送ります
⑥ プルリクエストを送ったらコードレビューをし、main ブランチにマージします
⑦ main ブランチにマージしたら、その main ブランチをすぐに本番サーバにデプロイします。
⑧ デプロイ後は速やかに作業していたブランチを削除する

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。
　一言いうと、サーバーサイドエンジニアはバックエンド（サーバーでプログラムの実行・管理）の仕事をしています。そして、フロントエンジニアはフロントエンド（ユーザーが見ている画面のデザイン）の仕事をしています。

・フロントエンドエンジニアの仕事内容
フロントエンジニアが使う主な言語は HTML/CSS、JavaScript（jQuery）2 つの言語を使って、ユーザーが実際に使う画面を作成します。
フロントエンジニアの仕事は、Web デザイナーが作ったデザインをともに画像ファイルなどを組み合わせながら、コードを記述し Web ページを作成することです。
作成が終わるとサーバーにデータをアップロードし、表示崩れの有無やサーバーと通信できるかを確認をしてリリースをします。

・サーバーサイドエンジニアの仕事内容
サーバーサイドで使われる言語で代表的なものは、Ruby、PHP、Python、Java などです。
サーバーサイドエンジニアの仕事内容は、サーバー側で実行する処理に必要なプログラムを開発がメインの仕事内容です。Web 上でユーザーが操作したときに操作内容に応じたプログラムの開発と保守をおこないます。

他にもサーバー側で扱う顧客情報などのデータ管理やデータベースにあるファイルへのアクセスなど、目に見えない処理を開発するのがサーバーサイドエンジニアです。

フロントエンドエンジニアの年収は 300 万円～700 万円となっています。サーバーサイドエンジニアの年収は 400 万円～800 万となっています。年収 1000 万以上を目指すこともできます。自分の研修経験および年収現状から見ると、サーバーサイドのほうは技術レベルがより高いと思います。

4. AWS とは何ですか。特徴を述べてください。

・AWS とは

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。

・特徴

①　変動的なコスト

AWS は変動的なコスト運用を行っているのが特徴です。AWS のコストのシステムは利用した分だけお金がかかる従量課金制をとっています。つまり、サーバを利用していない間は利用料金がかからないため、無駄なコストを削減することができるのが大きな特徴の一つです。また、ハードウェアやソフトウェアを購入する必要がないため、初期費用もかからないのも特徴的な点です。

②　目的に合わせて利用できる

AWS では、コンピューティングやストレージ、データベースだけでなく最新のテクノロジーなど、利用できるサービスは多岐に渡っています。さらにサービス内にも多くの機能があるため、AWS には利用可能なサービスや機能が非常に多くなっています。すべてのサービスを用いるのではなく、その中から必要なサービスだけを導入し、目的に合わせて利用できるという点も AWS の特徴的な点であるといえます。

③　高いセキュリティレベル

AWS は非常にセキュリティレベルが高いのも特徴の一つです。重要情報の扱いについては問題視される部分が多く、クラウディングサービスの弱点でもありました。しかし AWS はさまざまなセキュリティ認証を受け、高いセキュリティレベルを実現しているため、利用者からの信用を獲得しています。AWS は常に最新のセキュリティが施されているセキュアな環境であるといえるでしょう。

④　すぐに始められる

AWS はインターネットを介したサービスなので、利用開始時にハードウェアの購入や設置の必要がありません。すぐに必要な環境を整えることができるのが特徴の一つです。つまり、契約後すぐに AWS を始めることができるため、結果としてビジネスのスピードアップにもつながっていきます。スペックの変更に時間を要さないという柔軟な点も特徴の一つとして挙げられます。

⑤　質の高いパフォーマンス
クラウディングサービスは常に最新技術がアップデートされていくのが強みです。AWS でも常に定期的に最新化が行われるため、質の高いパフォーマンスを維持することができるのが AWS の特徴です。
また、この最新化は AWS 側が処理してくれるため、利用者側の手間がかからないという点も注目すべき点であり、AWS 利用のメリットの一つです。

⑥　無料期間が設けられている
AWS には「無料利用枠」と呼ばれる無料でサービスを利用できる期間があります。「12 か月無料枠」「無期限無料枠」「トライアル」の 3 種類があります。12 か月無料枠は AWS のアカウントを作成してから 1 年間有効な無料枠で、無期限無料枠とは AWS アカウントを使用している限り、無料で使用することができる無料枠です。トライアルは該当サービスを初めて使う際に一定期間だけ無料で使える無料枠となっています

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。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。

・メリット
①起動までの時間を大きく短縮できる
一般的な仮想サーバーでは、ゲスト OS 単位で仮想サーバーが生成されるため、OS の起動から利用可能な状態になるまで分単位の時間を要しますが、コンテナ型仮想化では、コンテナ起動に掛かる時間は数秒程度です。この身軽さ、素早さは大きな利点です。

②簡単に環境構築ができる
開発したシステムを動かすためには、同一環境を別のマシンで準備しなければなりません。同一環境の構築の手間、環境確認や環境テストなど、多くの手順が必要です。しかし、Docker は環境が丸ごと提供されるため、少ない手順でスピーディーに同じ環境を再現できます。

③Docker イメージの配布により開発環境構築の時間が短縮できる
作成した Docker イメージはパッケージとして配布できます。これにより、システム開発などで別のアプリケーションを動かす際にも、Docker イメージを利用すれば同一環境となり、環境のずれを防止できます。Docker イメージは開発環境を準備する時間の短縮にもおすすめです。

④ハードウェアリソースを削減できる

コンテナ自体は、アプリケーションの実行に必要なものだけに限定され、またホスト OS のカーネル部分だけを使用するため、無駄がなくパフォーマンスが向上し、リソースの削減も実現します。

⑤Docker Hub が利用できる
Docker の特徴としてはコンテナ共有化サービスの『Docker Hub』があります。『Docker Hub』を通じて世界中の開発者の成果物が共有できます。これによって、よりレベルの高い優れたイメージを効果的、効率的に活用できます。

⑥インフラ調達は後回しにできる
一般的なシステム開発では、サーバーなどのインフラ環境を先に整えますが、Docker コンテナでは、まずはアプリケーション作りから始めて問題がありません。アプリケーション開発が進んだところで、コンテナ単位で本番環境に乗せればよいので、システムエンジニアはインフラを気にせず開発に専念できます。

⑦ロールバックが速い
ロールバックの時間が速く簡単なことも、Docker のメリットの１つです。ロールバックとは、日本語で「後退」「巻き戻し」という意味で、PC の状態を指定したところまで戻すことを言います。
開発やテストを行う上で、バージョンアップや処理を繰り返すことは多いです。そのため、バージョンアップする前や処理を実行する前の状態に戻すのに時間がかかると、作業効率が悪くなります。
ロールバックが速く簡単なことは開発・テスト作業を行う上で重要です。Docker コンテナは状態の差分を保存するため、ロールバックの時間が速く簡単なのがメリットです。起動までの時間を大きく短縮できる