SeedTechSchool

総合テスト　問5　回答

2021年10月23日

松永　隆寛

※問題文および参考リンクを黒字、回答を赤字で示した。

1. **オブジェクト指向とは何かを述べてください**

➾オブジェクト(=モノ)を中心に考えたプログラミングスタイル(orプログラミング

パラダイム)

➾そのキモは、ある程度まとまった機能や性質を抽象化し、オブジェクトとして

　まとめて整理することで、より効率的に開発できるようにすること。

➾現実の例で言えば、例えばスズキ㈱などメーカーは、3つの同じような製造ラインの

工場(磐田工場、湖西工場、相良工場)を持つとして、1つの工場で作った製造

マニュアル(=オブジェクト)を、他の工場の同じ工程でも使い回すことで

効率化している。

これを、いくら工場が違うとは言っても、同じ機能を持つ工程で同じ作業を

しているのに、わざわざ各工場で1からマニュアル作っていたら非効率であるし、

いろいろずれてきてまとめるのが面倒。

➾オブジェクトは、クラスで設計される。クラスは、プロパティ(構成要素)、メソッド

(ふるまい)から成る。

　　➾後述の3つの特徴を有する。いずれもシステムをより安全に効率よく開発する

ことに資する。

　　➾なお、以下のURLにある通り、オブジェクト指向型もプログラミングパラダイムの

一種であり、プログラムの見方・考え方の一種である。ということなので、定義は

決まっておらず、人により定義が異なり一概に決まっていない。

「手続き型言語とは？オブジェクト指向言語と比較しわかりやすく解説！メリットやPythonにおすすめの型を紹介！」

<https://agency-star.co.jp/column/procedural-linguistics>

a. 特徴3つ、また、その説明を含めてください

　➾カプセル化　　　：オブジェクトのメンバー(変数などの要素)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。

　　　　　　　　　　　→こうすることで、無用なエラーなどの可能性を潰すことが

可能なため、大規模システム開発などで特に有効。

継承　　　　　　：一度定義したクラスを引き継いで新しいクラスを定義すること。

　　　　　　　　　→こうすることで、一度定義したクラスを再度一から作る必要

　　　　　　　　　　がなくなる上、共通する部分を整理できるようになり、

　　　　　　　　　　開発の効率化に資する。

ポリモーフィズム：同じ命令を送ったとしても、それぞれのクラスが独立した

固有の処理を行う特性をいう。

→こうすることで、同じメソッドであれば共通化でき、

　コードがきれいにまとまる。ひいては、開発の効率化に資する。

　　➾注意：この3つは、オブジェクト指向が備える代表的な要素であるだけで、

オブジェクト指向の定義ではない。

b. 具体例を含めてください

　➾カプセル化　　　：そのクラスにアクセスできる範囲を絞ることで、

　　　　　　　　　　　他でコーディング修正した際に影響範囲を小さくできる。

　　　　　　　　　　　例えば、ある1つのクラスからデータを読み出す際の方法が

　　　　　　　　　　　2つあった場合、他でコーディング修正した際の修正量が

　　　　　　　　　　　単純に2倍になる可能性がある。それだけエラーが増えやすく

　　　　　　　　　　　なる。なので、読み出す方法は1つに絞る、などが、カプセル化

　　　　　　　　　　　の1つの手法。

継承　　　　　　：クラス　動物　{ 鳴く() { } };

　　　　　↓

クラス　犬　{ 鳴く() { “ワン”} };

クラス　猫　{ 鳴く() { “にゃー”} };

　　　　　 →子クラスに親クラスのメソッドを継承することで、

　　　　　　 　新しくメソッドを定義しなくてもよい。

ポリモーフィズム：(個別に関数を作成)

　　　　　 　動物の種類ごとに異なるアウトプットが出力される。

クラス 犬 { 鳴く() { "ワン" } };

クラス 猫 { 鳴く() { "ニャー" } };

クラス アヒル { 鳴く() { "クワッ" } };

　　　　　　　 　↓「鳴く」というメソッドを共通化する

　　　　 (共通化後)

　　　　　 　共通化前と同じく、動物の種類ごとに異なるアウトプット

が出力される。

動物 = { 犬,猫,アヒル };

for (動物) { 動物.鳴く(); }

➾コードが3行→2行にまとめられた。

参考リンク：

「Part3 オブジェクト指向プログラミングの三本柱」

<https://xtech.nikkei.com/it/article/lecture/20061204/255715/>

「Qiita　オブジェクト指向の3大要素　まとめ」

<https://qiita.com/yuto-ktok/items/2c8172397ba9dc76fe7a>

「ポリモーフィズムとはなにか？」

<https://webpia.jp/polymorphism/>

「オブジェクト指向の多様性(ポリモーフィズム)についてわかりやすく解説」

<https://engineer-life.dev/polymorphism/>

「オブジェクト指向のカプセル化とは？初心者向けにわかりやすく解説！」

<https://savvycode.dev/programming-oriented-object-encapslation/>

「手続き型言語とは？オブジェクト指向言語と比較しわかりやすく解説！メリットやPythonにおすすめの型を紹介！」

<https://agency-star.co.jp/column/procedural-linguistics>

**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 flowとは**

・Github flowとは、Githubを利用した開発の流れを意味する。

・いくつか専門用語があり、以下に示す。

i. リポジトリ：データを保管しておく貯蔵庫

ii. main：Merge後のデータを保管するブランチをMainブランチとする。

iii. リモート：リモートリポジトリは、サーバ上に置かれる貯蔵庫

iv. ブランチ：書いたコードを載せるトラックのようなもの

ⅴ. 各種：修正保存する＝Add、ローカルに保存＝Commit、ローカル→リモートに

　　移す＝Push、リモートリポジトリがブランチをまとめる行為＝Merge

・開発には、個人開発とチーム開発がある。

・個人開発では、Add、Commit、Pushまでの使用が一般的。

・チーム開発では、ブランチが登場する。

・Mainブランチからトピックブランチを切る。

　　・Mergeする前に、チームメイトに承認を依頼する「Pull request」を送る。

・デザイナーやエンジニアは各々が任されたブランチで作業し、完成したらMain

ブランチにMergeさせて、完成させる。

　　・Mainブランチをプルする。

　　・以降、同様に繰り返し、全体を完成させる。

1. **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。**



　参考URL

　「フロントエンドとサーバーサイドの違い。仕事内容・年収・需要を比較」

<https://www.sejuku.net/blog/100525>

**4. AWSとは何ですか。特徴を述べてください。**

・Amazonが提供するクラウドサービス

・サーバ、ストレージ、ソフト、などシステム構築・運用に必要なサービスを

必要なときに必要な分だけ利用可能

・AWSだけで資格がある

・インフラエンジニア目指す人は重要

・サービス総数は100以上

・初期費用なし

・常に最新のセキュリティ

・定期的にアップグレードされる

**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：「コンテナ型」の仮想環境を構築できるOSS

・１つのハードウェアで、複数のOSを使用できる。

・例えば、MacOSが標準OSのハードでも、LinuxやWindowsOSを使用できる。

・「コンテナ型」とは、アプリケーション起動に必要なアプリケーション、ライブラリ、

設定ファイルを一つの箱に入れたようなもの。コンテナエンジンの上で動作させる。

・コンテナ型との対比でハイパーバイザ型がある。

　　ハイパーバイザ型は、ホストOS上でゲストOSが別に起動する形になり、メモリや

CPUを食う。一方、コンテナ型は、ホストOSのカーネルを共有することで動き、

ゲストOSを起動しないので、比較してメモリやCPUを食わない。