Link GitHub : <https://github.com/scrubScr1b/uas-pbo>

Uraian

1. **Sebutkan dan jelaskan 4 pilar dari pemograman berorientasi objek. (Bobot 20%)**
2. Abstraksi: Abstraksi adalah konsep dalam OOP yang memungkinkan kita untuk mengisolasi data dan perilaku yang relevan dari suatu objek, dan mengabaikan detail yang tidak perlu. Dalam hal ini, kita membuat kelas-kelas yang merepresentasikan objek-objek dalam program kita, dan kita hanya fokus pada fitur-fitur yang penting untuk tujuan program tersebut. Misalnya, jika kita sedang membuat program untuk mengelola sebuah perpustakaan, kita bisa membuat kelas "Buku" dengan atribut-atribut seperti judul, penulis, dan nomor ISBN, tetapi kita tidak perlu memperhatikan bagaimana buku tersebut disimpan dalam memori atau bagaimana cara pencarian dilakukan.
3. Pewarisan (Inheritance): Pewarisan adalah mekanisme yang memungkinkan kita untuk membuat kelas baru berdasarkan kelas yang sudah ada, dengan mewarisi atribut dan metode yang dimiliki oleh kelas tersebut. Pewarisan memungkinkan kita untuk membangun hierarki kelas yang lebih spesifik atau spesialis, di mana kelas yang lebih tinggi dalam hierarki (kelas induk atau superclass) memiliki fitur-fitur umum, dan kelas yang lebih rendah dalam hierarki (kelas anak atau subclass) memiliki fitur-fitur yang lebih spesifik. Dengan pewarisan, kita dapat menggunakan kembali kode yang sudah ada, mengurangi duplikasi kode, dan mempermudah pemeliharaan kode.
4. Polimorfisme: Polimorfisme adalah konsep yang memungkinkan kita untuk menggunakan objek-objek dari berbagai kelas yang berbeda dengan cara yang seragam. Dalam OOP, polimorfisme tercapai melalui konsep polimorfisme parameter dan polimorfisme penciptaan objek. Polimorfisme parameter memungkinkan kita untuk menggunakan objek dari kelas anak sebagai argumen untuk parameter yang dideklarasikan dengan tipe kelas induk, sehingga memungkinkan fleksibilitas dalam pemanggilan metode. Polimorfisme penciptaan objek memungkinkan kita untuk membuat objek dengan menggunakan referensi dari kelas induk, tetapi sebenarnya objek yang dibuat adalah objek dari kelas anak. Polimorfisme memungkinkan kita untuk menulis kode yang lebih generik, lebih fleksibel, dan dapat diterapkan pada berbagai objek yang berbeda.
5. Encapsulation (Pengkapsulan): Encapsulation adalah konsep yang menggabungkan data dan metode yang beroperasi pada data tersebut menjadi sebuah entitas tunggal, yang disebut sebagai kelas. Dalam pemrograman berorientasi objek, kita menggunakan modifier akses seperti public, private, dan protected untuk mengatur aksesibilitas data dan metode di dalam kelas. Hal ini membantu menjaga keamanan dan integritas data, serta mencegah manipulasi langsung terhadap data dari luar kelas. Dalam hal ini, hanya metode-metode yang didefinisikan dalam kelas yang dapat berinteraksi dengan data tersebut. Pengkapsulan juga memfasilitasi konsep enkapsulasi data, di manadata hanya dapat diakses melalui metode-metode yang ditentukan, sehingga memungkinkan pemrosesan dan validasi data sebelum diakses atau dimodifikasi.

Dalam keseluruhan, keempat pilar ini membantu dalam merancang program dengan pendekatan yang terstruktur dan modular, mempromosikan reusabilitas, memudahkan pemeliharaan, dan meningkatkan fleksibilitas dalam pengembangan perangkat lunak.

1. **Sebutkan dan jelaskan jenis-jenis exception / error pada java. (Bobot 10%)**
2. NullPointerException

NullPointerException (NPE) adalah exception yang paling sering muncul. Sir Charles Antony Richard Hoare, pencipta Null Reference sendiri mengatakan bahwa penciptaan null reference pada tahun 1965 adalah sebuah kesalahan yang menyebabkan kerugian yang sangat besar hingga terciptalah istilah one billion dollars mistake.

1. NumberFormatException

Diposisi kedua adalah NumberFormatException yang terjadi saat kita mencoba mengubah String menjadi nilai angka namun String tersebut tidak terformat dengan benar. Misalnya, kita ingin mengubah String kosong untuk menjadi Integer, maka proses perubahan ini bisa menyebabkan NFE terjadi.

1. IllegalArgumentException

Exception ini muncul saat kita memanggil suatu method dengan mengirimkan parameter yang tidak sesuai. Misalnya suatu method meminta parameter bertipe X sedangkan kita mengirimkan data bertipe Y sebagai parameternya. Error ini muncul karena method tersebut tidak dapat memproses parameter yang dikirim.

1. RuntimeException

RuntimeException adalah jenis error yang muncul saat program sedang berjalan dan tidak memberikan compilation error. Sebagai salah satu jenis unchecked exception, RuntimeException tidak bisa dicegah lewat kode.

Salah satu contohnya adalah saat kita mencoba membagi sesuatu dengan angka 0, maka akan muncul ArithmeticException (kelas ini meng-extends RuntimeException). Error karena pembagian dengan angka nol ini tidak bisa diperiksa saat proses kompilasi dan hanya akan terdeteksi saat program sudah berjalan.

1. IllegalStateException

Sebuah IllegalStateException terjadi saat kita mencoba memanggil method diwaktu yang salah. Misalnya, saat kita menggunakan URLConnection untuk terhubung kembali ke alamat yang sudah terhubung kita akan mendapat "IllegalStateException: Already Connected".

1. NoSuchMethodException

Error ini muncul saat kita mencoba memanggil method yang tidak ada.

1. ClassCastException

ClassCastException terjadi saat kita mencoba mengubah sebuah kelas menjadi kelas lain yang bukan \*instance-\*nya. Kita tidak bisa mengubah suatu objek menjadi objek bertipe kelas yang tidak ia wariskan.

1. Exception

Exception adalah kelas utama dimana semua jenis Exception mewariskannya. Java tidak pernah memanggil Exception secara langsung. Ia muncul bisa jadi karena kita atau pihak ketiga memanggilnya secara eksplisit.

1. ParseException

ParseException muncul saat kita mencoba mengubah String menjadi data tertentu tapi String tersebut tidak terformat dengan benar. Exception ini bisa kita hindari dengan memeriksa terlebih dahulu String tersebut apakah sudah terformat dengan benar atau belum.

1. InvocationTargetException

InvocationTargetException terjadi apabila sebuah method atau konstruktor yang kita panggil menyebabkan suatu exception terpanggil.

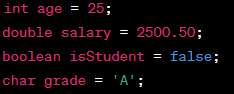
1. **Jelaskan perbedaan dari tipe data promitif dan tipe data objek. (Bobot 10%)**

Tipe data primitif dan tipe data objek adalah dua konsep yang berbeda dalam pemrograman. Berikut adalah penjelasan singkat mengenai perbedaan antara keduanya:

Tipe Data Primitif:

Tipe data primitif adalah tipe data dasar yang telah ditentukan oleh bahasa pemrograman itu sendiri. Mereka biasanya digunakan untuk menyimpan nilai tunggal. Contoh tipe data primitif umum termasuk integer (bilangan bulat), float (bilangan pecahan), boolean (benar/salah), dan char (karakter). Tipe data primitif biasanya memiliki ukuran tetap dalam memori dan diwakili secara langsung sebagai bit atau byte dalam bahasa pemrograman.

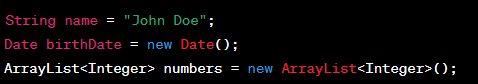
Contoh penggunaan tipe data primitif dalam bahasa Java:



Tipe Data Objek:

Tipe data objek adalah tipe data yang didefinisikan oleh pengguna atau disediakan oleh bahasa pemrograman sebagai kelas atau struktur data kompleks. Mereka dapat menyimpan data dan fungsi (metode) yang terkait bersama. Tipe data objek dibuat menggunakan kelas dan biasanya digunakan untuk menyimpan sekumpulan data terkait yang lebih kompleks atau untuk merepresentasikan konsep dari dunia nyata.

Contoh penggunaan tipe data objek dalam bahasa Java:



Perbedaan antara tipe data primitif dan tipe data objek adalah:

* Tipe data primitif memiliki nilai default yang dapat digunakan tanpa inisialisasi, sementara tipe data objek memerlukan pembuatan objek menggunakan operator new.
* Tipe data primitif disimpan secara langsung di dalam memori, sedangkan tipe data objek disimpan sebagai referensi ke objek di memori.
* Tipe data primitif tidak memiliki metode atau fungsi terkait, sementara tipe data objek dapat memiliki metode dan fungsi yang dapat digunakan untuk memanipulasi data yang dikandungnya.
* Tipe data objek dapat diwarisi (inheritance) dan memiliki hubungan yang lebih kompleks melalui penggunaan konsep pemrograman berorientasi objek, sedangkan tipe data primitif tidak dapat diwarisi atau memiliki hubungan semacam itu.

Pemilihan antara tipe data primitif dan tipe data objek tergantung pada kebutuhan program dan jenis data yang akan disimpan dan dioperasikan.