1. **Object Oriented Programming** (OOP) dalam bahasa Indonesia diartikan sebagai **Pemograman Berorientasi Object** merupakan metode pemrograman yang berorientasi pada objek. Beberapa objek tersebut akan saling berinteraksi untuk menyelesaikan masalah yang rumit.
2. **Stream** adalah dasar dari operasi input-output (IO) di java yang ada di dalam package java.io sebagai package utama. Stream digunakan untuk menangkap input dari device ataupun mengeluarkan output untuk device.

   Lambda adalah fungsi yang didefinisikan tanpa menggunakan identifier. Lambda digunakan untuk menyederhanakan kode program saat ingin menggunakan method dalam class interface, sedangkan class interface tersebut hanya memiliki 1 method.
3. Perbedaan Abstract Class dan Interface.

Abstract Class	Interface
Bisa berisi abstract dan non-abstract method.	Hanya boleh berisi abstract method.
Kita harus menuliskan sendiri modifiernya.	Kita tidak perlu susah2 menulis public abstract di depan nama method. Karena secara implisit, modifier untuk method di interface adalah public dan abstract.
Bisa mendeklarasikan constant dan instance variable.	Hanya bisa mendeklarasikan constant. Secara implisit variable yang dideklarasikan di interface bersifatpublic, static dan final.
Method boleh bersifat static.	Method tidak boleh bersifat static.
Method boleh bersifat final.	Method tidak boleh bersifat final.
Suatu abstact class hanya bisa mengextend satu abstract class lainnya.	Suatu interface bisa meng-extend satu atau lebih interface lainnya.
Suatu abstract class hanya bisa meng-extend satu abstract class dan meng-implement beberapa interface.	Suatu interface hanya bisa meng-extend interface lainnya. Dan tidak bisa meng-implement class atau interface lainnya.

4. **Functional Interface** adalah sebuah interface yang hanya mempunyai satu method abstract saja
5. ***SOLID** principle* merupakan singkatan dari 5 buah prinsip yang sangat berhubungan dengan OOP. SOLID principle akan sangat berguna apabila diimplementasikan ke dalam sebuah program skala besar, alasannya dengan SOLID principle membuat *class* yang ada didalam program bisa di*maintenance* tanpa banyak melibatkan *class* yang lainnya.
6.
7.
8. **Unit Test**: pengujian bagian terkecil dari sebuah code, bagian terkecil ini adalah bisa sebuah fungsi, module atau class dari sistem tersebut.
   Integration Test: pengujian apakah gabungan dari bagian (fungsi) dari sebuah aplikasi atau system dapat bekerja sama dengan benar.
9. **Test-driven development (TDD)** adalah proses pengembangan perangkat lunak yang bergantung pada pengulangan siklus pengembangan yang sangat singkat. Persyaratan untuk melakukan sebuah perubahan menjadi kasus uji yang sangat spesifik, maka software yang sedang dikembangkan diharuskan memenuhi test baru.