1.
-Enkapsulasi adalah konsep untuk mendefinisikan sifat scoop pada variabel, object, class, dll. Enkapsulasi dapat mengatur variabel/object apa saja yang bisa diakses.
-Inheritance adalah konsep penurunan sifat induk kepada anaknya, misalnya child class akan memiliki sifat dan method dari parent class.
-Abstrak adalah konsep dimana class dapat didefinisikan ulang dengan bentuk yang berbeda, misalnya class 2D abstrak dapat menjadi class segitiga, persegi, dll

2.
-Coupling adalah ketergantungan modul dengan modul lain.
-Cohesion adalah keterikatan fungsi pada satu modul.
-Komposisi yang baik adalah Loose Coupling, High Cohesion supaya prinsip penggunaan ulang dapat lebih baik

3.
-Creational pattern adalah pola dan solusi dari kejadian pembuatan suatu object/instance.
-Behavioral pattern adalah pola dan solusi dari komunikasi antar object-object.
-Structural pattern adalah pola dan solusi dari pengaturan komposisi class dan object.

4.
a. Manifest berfungsi untuk mengatur permission, fitur, activity, dan fragment yang digunakan.
b. Activity berfungsi untuk mengatur logic, proses bisnis, dan mengontrol suatu tampilan.
c. Drawable berfungsi untuk menyimpan dan menampilkan gambar.

5. Pemrograman yang dapat membuat dan mengatur socket yang merupakan titik komunikasi antar proses pada jaringan komputer.
TCP/IP dan UPD/IP.
Perbedaan:
A)TCP
-sinkronous
-kecepatan lebih lama
-kemungkinan data hilang kecil
B)UDP
-asinkronous
-kecepatan lehih cepat
-kemungkinan data hilang besar

6. C
7. C
8. B
9. D
10.D

11.
```
static void printFibo(int loop){
   if(loop>0){
       n3 = n1 + n2;
       n1 = n2;
       n2 = n3;
       System.out.print(" "+n3);
       printFibo(loop-1);
    }
```

```
 }

public static void main(String[] args) {
    printFibo(20);
  }
```

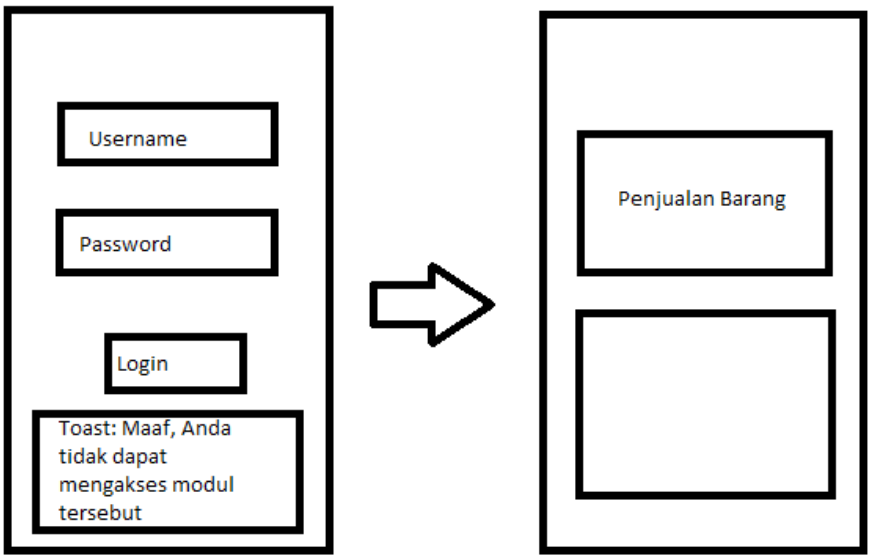
12.

```
static void printReverse(String inp){
          char[] expl = inp.toCharArray();

       for (int i = expl.length - 1; i >= 0; i--){
          System.out.print(expl[i]);
          }
}

public static void main(String[] args) {
    printReverse("aku ingin makan");
  }
```

13.

```
static void printContThree(int max){
          for (int i = 0; i<=max; i++){
                    if(i.toString().contains("3")){
                              System.out.print(i+", ");
                    }
          }
}

public static void main(String[] args) {
    printContThree(100);
  }
```

14.



15.

```
static final String DB_URL = "jdbc:MSSQLSERVER://192.168.1.167";
  static final String USER = "Arif";
  static final String PASS = "123456";
  static final String QUERY = "SELECT USERID, NAMA, STATUS";

  public static void main(String[] args) {
    // Open a connection
    try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
      Statement stmt = conn.createStatement();
      ResultSet rs = stmt.executeQuery(QUERY);
    ) {
      while(rs.next()){
        //Display values
        System.out.print("USERID: " + rs.getInt("USERID"));
        System.out.print(", NAMA: " + rs.getInt("NAMA"));
        System.out.print(", STATUS: " + rs.getString("STATUS"));
      }
    } catch (SQLException e) {
      e.printStackTrace();
    }
  }
```