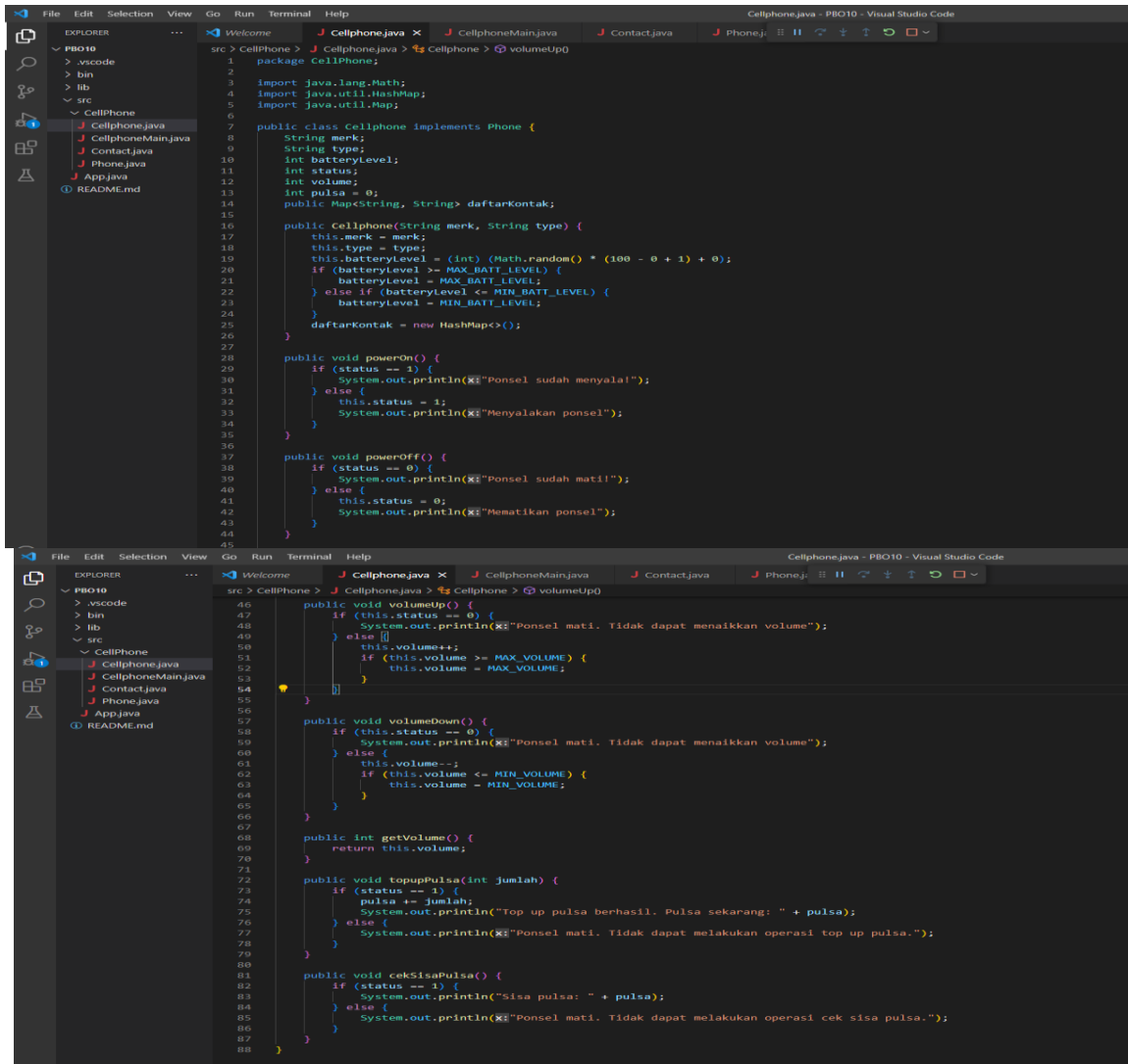


Nama : Mochammad Abdurrochman Ari Wibowo

NIM : A11.2021.13297

Laporan Tugas PBO Pertemuan ke-10

- **CellPhone.java**



```
CellPhone.java - PBO10 - Visual Studio Code
src > CellPhone > CellPhone.java > CellPhone > volumeUp()
package CellPhone;

import java.lang.Math;
import java.util.HashMap;
import java.util.Map;

public class CellPhone implements Phone {
    String merk;
    String type;
    int batteryLevel;
    int status;
    int volume;
    int pulsa = 0;
    public Map<String, String> daftarkontak;

    public CellPhone(String merk, String type) {
        this.merk = merk;
        this.type = type;
        this.batteryLevel = (int) (Math.random() * (100 - 0 + 1) + 0);
        if (batteryLevel >= MAX_BATT_LEVEL) {
            batteryLevel = MAX_BATT_LEVEL;
        } else if (batteryLevel <= MIN_BATT_LEVEL) {
            batteryLevel = MIN_BATT_LEVEL;
        }
        daftarkontak = new HashMap<>();
    }

    public void powerOn() {
        if (status == 1) {
            System.out.println(Ki"Ponsel sudah menyala!");
        } else {
            this.status = 1;
            System.out.println(Ki"Menyalakan ponsel");
        }
    }

    public void powerOff() {
        if (status == 0) {
            System.out.println(Ki"Ponsel sudah mati!");
        } else {
            this.status = 0;
            System.out.println(Ki"Mematikan ponsel");
        }
    }

    public void volumeUp() {
        if (this.status == 0) {
            System.out.println(Ki"Ponsel mati. Tidak dapat menaikkan volume");
        } else {
            this.volume++;
            if (this.volume >= MAX_VOLUME) {
                this.volume = MAX_VOLUME;
            }
        }
    }

    public void volumeDown() {
        if (this.status == 0) {
            System.out.println(Ki"Ponsel mati. Tidak dapat menaikkan volume");
        } else {
            this.volume--;
            if (this.volume <= MIN_VOLUME) {
                this.volume = MIN_VOLUME;
            }
        }
    }

    public int getVolume() {
        return this.volume;
    }

    public void topupPulsa(int jumlah) {
        if (status == 1) {
            pulsa += jumlah;
            System.out.println("Top up pulsa berhasil. Pulsa sekarang: " + pulsa);
        } else {
            System.out.println(Ki"Ponsel mati. Tidak dapat melakukan operasi top up pulsa.");
        }
    }

    public void cekSisaPulsa() {
        if (status == 1) {
            System.out.println("Sisa pulsa: " + pulsa);
        } else {
            System.out.println(Ki"Ponsel mati. Tidak dapat melakukan operasi cek sisa pulsa.");
        }
    }
}
```

Program di atas adalah implementasi kelas **Cellphone** yang mengimplementasikan interface **Phone**. Program ini merupakan representasi sederhana dari sebuah ponsel (cell phone) dengan beberapa fungsi dasar seperti menghidupkan dan mematikan ponsel, mengatur volume, melakukan top up pulsa, dan memeriksa sisa pulsa.

Berikut adalah penjelasan singkat mengenai setiap bagian program:

1. Mendefinisikan package **CellPhone**.

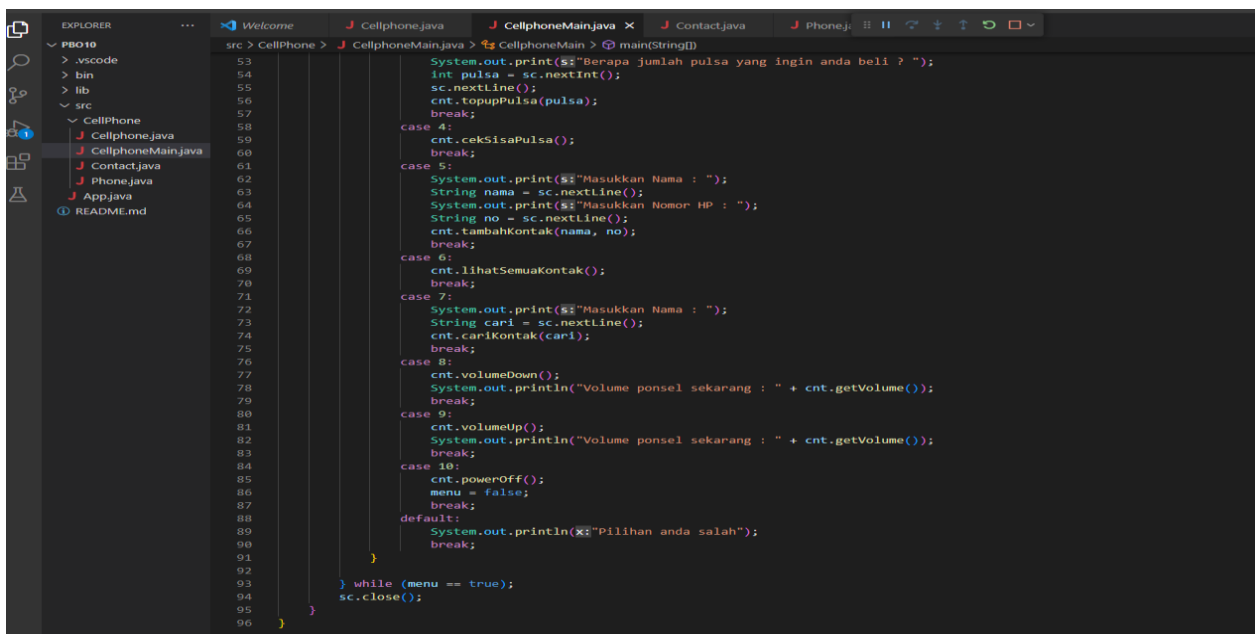
2. Mengimport kelas **Math** dan **HashMap** dari package **java.lang** dan **java.util** secara berturut-turut.
3. Mendefinisikan kelas **Cellphone** dan mengimplementasikan interface **Phone**.
4. Mendeklarasikan beberapa variabel member seperti **merk**, **type**, **batteryLevel**, **status**, **volume**, dan **pulsa**.
5. Mendeklarasikan objek **daftarKontak** yang merupakan **Map** untuk menyimpan daftar kontak ponsel.
6. Membuat konstruktor **Cellphone** yang menerima parameter **merk** dan **type**. Konstruktor ini akan menginisialisasi nilai-nilai awal seperti **batteryLevel** secara acak, dan mengatur batas atas dan batas bawah baterai (**MAX_BATT_LEVEL** dan **MIN_BATT_LEVEL**).
7. Mendefinisikan metode **powerOn()** yang digunakan untuk menghidupkan ponsel. Jika ponsel sudah menyala, akan mencetak pesan "Ponsel sudah menyala!".
8. Mendefinisikan metode **powerOff()** yang digunakan untuk mematikan ponsel. Jika ponsel sudah mati, akan mencetak pesan "Ponsel sudah mati!".
9. Mendefinisikan metode **volumeUp()** untuk menaikkan volume ponsel. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat menaikkan volume".
10. Mendefinisikan metode **volumeDown()** untuk menurunkan volume ponsel. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat menurunkan volume".
11. Mendefinisikan metode **getVolume()** yang mengembalikan nilai volume saat ini.
12. Mendefinisikan metode **topupPulsa()** untuk melakukan top up pulsa ponsel. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat melakukan operasi top up pulsa".
13. Mendefinisikan metode **cekSisaPulsa()** untuk memeriksa sisa pulsa ponsel. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat melakukan operasi cek sisa pulsa".

- **CellPhoneMain.java**

```

1 package CellPhone;
2
3 import java.util.Scanner;
4
5 public class CellPhoneMain {
6     public static void main(String[] args) {
7         String merk, tipe;
8         char hidup;
9         boolean menu = true;
10
11         Scanner sc = new Scanner(System.in);
12         System.out.println("Masukkan Merk dan Tipe ponsel==");
13         merk = sc.nextLine();
14         System.out.println("Tipe :");
15         tipe = sc.nextLine();
16
17         // cellphone cp = new Cellphone(merk, tipe);
18         Contact cnt = new Contact(merk, tipe);
19         System.out.println("Ponsel : " + merk + " " + tipe);
20         System.out.println("Hidupkan ponsel ? (y/t)");
21         hidup = sc.next().charAt(0);
22         if (hidup == 'Y' || hidup == 'y') {
23             cnt.powerOn();
24         } else {
25             cnt.powerOff();
26         }
27
28         do {
29             System.out.println("Apa yang ingin anda lakukan pada ponsel ? ==");
30             System.out.println("1. Menyalakan Ponsel");
31             System.out.println("2. Mematikan Ponsel");
32             System.out.println("3. TopUp Pulsa");
33             System.out.println("4. Cek Pulsa");
34             System.out.println("5. Tambah Kontak");
35             System.out.println("6. Lihat Semua Kontak");
36             System.out.println("7. Cari Kontak");
37             System.out.println("8. Volume Down");
38             System.out.println("9. Volume Up");
39             System.out.println("10. Matikan dan Keluar");
40             System.out.println("=====");
41             System.out.print("Pilih : ");
42             int pilih = sc.nextInt();
43             sc.nextLine();
44         } while (menu);
45     }
46 }

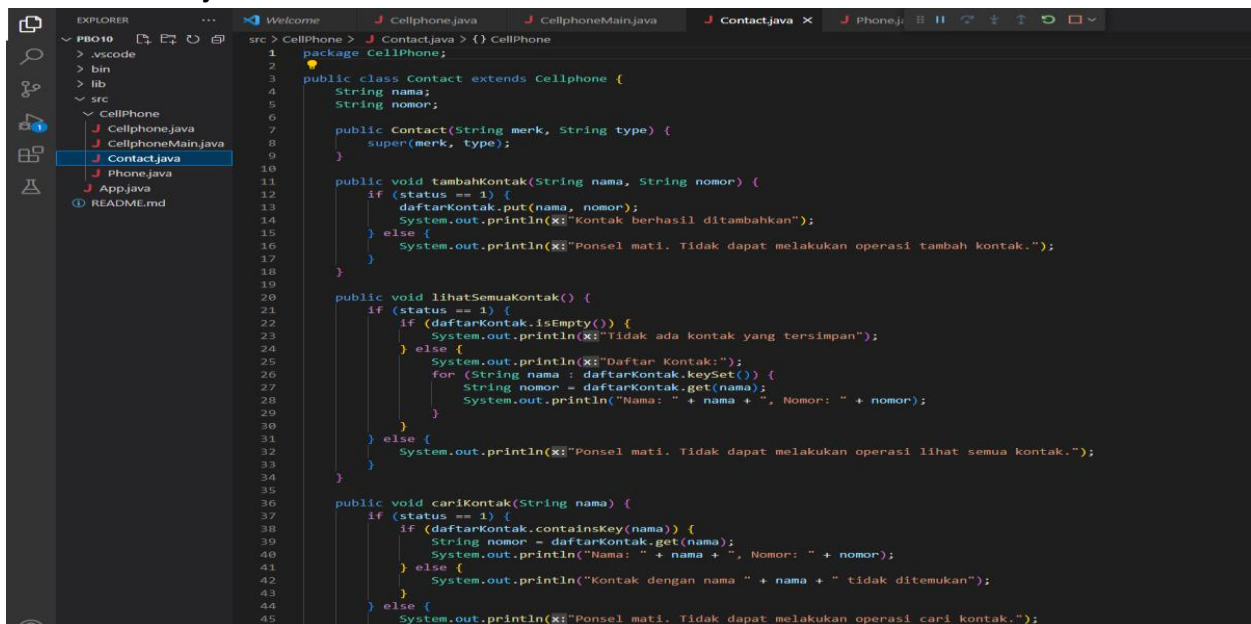
```

A screenshot of the Visual Studio Code editor. The Explorer sidebar on the left shows a project structure with folders .vscode, bin, lib, and src. Inside src, there is a folder named CellPhone containing files CellPhone.java, CellPhoneMain.java, Contact.java, Phone.java, App.java, and a README.md file. CellPhoneMain.java is selected and open in the main editor. The code in CellPhoneMain.java shows a main method with a switch-case menu. The menu options include: printing the number of pulses, topping up, checking balance, entering name and HP number to add a contact, viewing all contacts, searching for a contact, and changing volume. A while loop ensures the menu is displayed as long as the user is not powered off (menu == true).

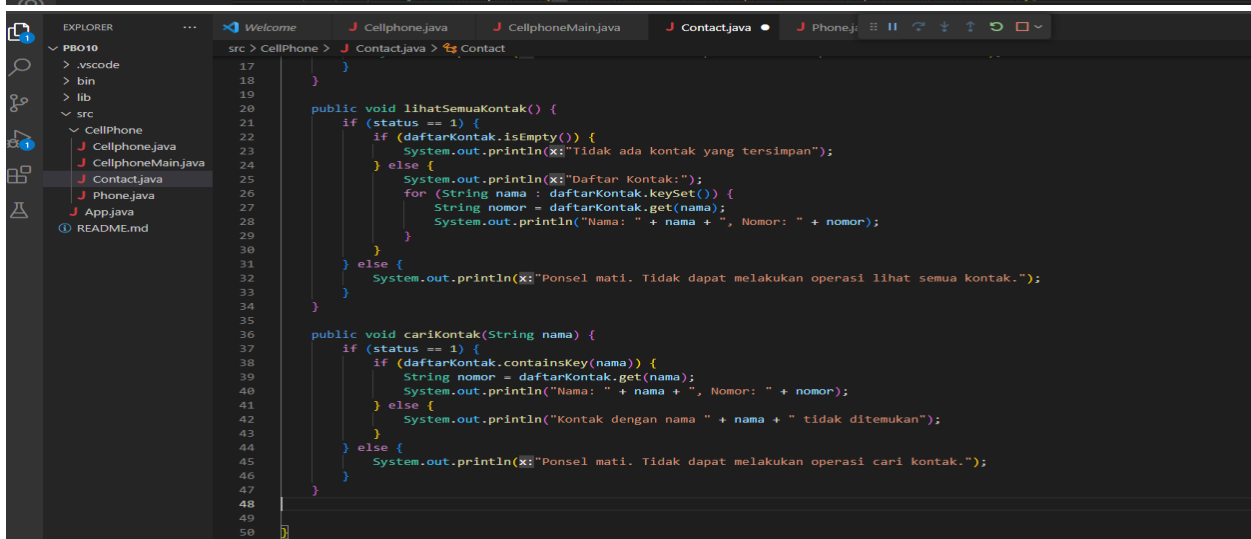
```
src > CellPhone > J CellPhoneMain.java > main(String[] args)
53     System.out.print($2"Berapa jumlah pulsa yang ingin anda beli ? ");
54     int pulsa = sc.nextInt();
55     sc.nextLine();
56     cnt.topupPulsa(pulsa);
57     break;
58
59     case 4:
60     cnt.cekSisaPulsa();
61     break;
62
63     case 5:
64     System.out.print($2"Masukkan Nama : ");
65     String nama = sc.nextLine();
66     System.out.print($2"Masukkan Nomor HP : ");
67     String no = sc.nextLine();
68     cnt.tambahKontak(nama, no);
69     break;
70
71     case 6:
72     cnt.lihatSemuaKontak();
73     break;
74
75     case 7:
76     System.out.print($2"Masukkan Nama : ");
77     String cari = sc.nextLine();
78     cnt.cariKontak(cari);
79     break;
80
81     case 8:
82     cnt.volumeDown();
83     System.out.println("Volume ponsel sekarang : " + cnt.getVolume());
84     break;
85
86     case 9:
87     cnt.volumeUp();
88     System.out.println("Volume ponsel sekarang : " + cnt.getVolume());
89     break;
90
91     case 10:
92     cnt.powerOff();
93     menu = false;
94     break;
95
96     default:
97     System.out.println($2"Pilihan anda salah");
98     break;
99
100 } while (menu == true);
101 sc.close();
102 }
```

CellPhoneMain.java adalah main class dimana seluruh class yang lain akan di gunakan disini. Diberikan menu dengan menggunakan swich case agar user dapat memilih secara langsung apa yang di ingin dilakukan user, serta diberikan perulangan DoWhile agar menu selalu tampil sebelum user menginputkan pilihan menu.

- **Contact.java**

A screenshot of the Visual Studio Code editor showing the Contact.java file. The Explorer sidebar shows the same project structure as the previous image, with Contact.java selected. The code in Contact.java defines a Contact class that extends CellPhone. It includes methods for adding a contact (tambahKontak), viewing all contacts (lihatSemuaKontak), and searching for a contact (cariKontak). Each method has a status parameter (1 for success, 0 for failure) and uses a HashMap (daftarKontak) to store contacts.

```
src > CellPhone > J Contact.java > Contact
1 package CellPhone;
2
3 public class Contact extends CellPhone {
4     String nama;
5     String nomor;
6
7     public Contact(String merk, String type) {
8         super(merk, type);
9     }
10
11     public void tambahKontak(String nama, String nomor) {
12         if (status == 1) {
13             daftarKontak.put(nama, nomor);
14             System.out.println($2"Kontak berhasil ditambahkan");
15         } else {
16             System.out.println($2"Ponsel mati. Tidak dapat melakukan operasi tambah kontak.");
17         }
18     }
19
20     public void lihatSemuaKontak() {
21         if (status == 1) {
22             if (daftarKontak.isEmpty()) {
23                 System.out.println($2"Tidak ada kontak yang tersimpan");
24             } else {
25                 System.out.println($2"Daftar Kontak:");
26                 for (String nama : daftarKontak.keySet()) {
27                     String nomor = daftarKontak.get(nama);
28                     System.out.println("Nama: " + nama + ", Nomor: " + nomor);
29                 }
30             }
31         } else {
32             System.out.println($2"Ponsel mati. Tidak dapat melakukan operasi lihat semua kontak.");
33         }
34     }
35
36     public void cariKontak(String nama) {
37         if (status == 1) {
38             if (daftarKontak.containsKey(nama)) {
39                 String nomor = daftarKontak.get(nama);
40                 System.out.println("Nama: " + nama + ", Nomor: " + nomor);
41             } else {
42                 System.out.println("Kontak dengan nama " + nama + " tidak ditemukan");
43             }
44         } else {
45             System.out.println($2"Ponsel mati. Tidak dapat melakukan operasi cari kontak.");
46         }
47     }
48
49
50 }
```

This is another screenshot of the Visual Studio Code editor showing the same Contact.java file. The code is identical to the previous screenshot, showing the Contact class with its methods and attributes.

```
src > CellPhone > J Contact.java > Contact
17 }
18
19
20 public void lihatSemuaKontak() {
21     if (status == 1) {
22         if (daftarKontak.isEmpty()) {
23             System.out.println($2"Tidak ada kontak yang tersimpan");
24         } else {
25             System.out.println($2"Daftar Kontak:");
26             for (String nama : daftarKontak.keySet()) {
27                 String nomor = daftarKontak.get(nama);
28                 System.out.println("Nama: " + nama + ", Nomor: " + nomor);
29             }
30         }
31     } else {
32         System.out.println($2"Ponsel mati. Tidak dapat melakukan operasi lihat semua kontak.");
33     }
34 }
35
36 public void cariKontak(String nama) {
37     if (status == 1) {
38         if (daftarKontak.containsKey(nama)) {
39             String nomor = daftarKontak.get(nama);
40             System.out.println("Nama: " + nama + ", Nomor: " + nomor);
41         } else {
42             System.out.println("Kontak dengan nama " + nama + " tidak ditemukan");
43         }
44     } else {
45         System.out.println($2"Ponsel mati. Tidak dapat melakukan operasi cari kontak.");
46     }
47 }
48
49
50 }
```

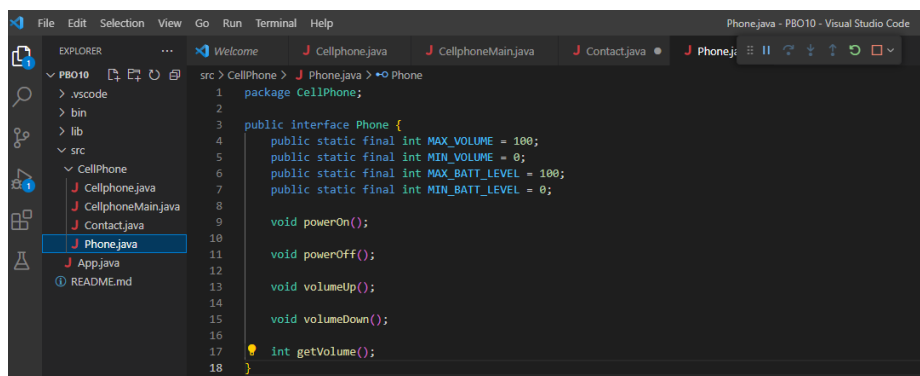
Program di atas adalah kelas **Contact** yang merupakan subkelas dari **Cellphone**. Kelas ini digunakan untuk mengelola kontak pada ponsel.

Berikut adalah penjelasan singkat mengenai setiap bagian program:

1. Mendefinisikan package **CellPhone**.
2. Mendefinisikan kelas **Contact** yang merupakan subkelas dari **Cellphone**.
3. Mendeklarasikan variabel **nama** dan **nomor** untuk menyimpan informasi kontak.
4. Membuat konstruktor **Contact** yang menerima parameter **merk** dan **type** untuk menginisialisasi objek **Cellphone**.
5. Mendefinisikan metode **tambahKontak()** untuk menambahkan kontak baru. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat melakukan operasi tambah kontak".
6. Mendefinisikan metode **lihatSemuaKontak()** untuk melihat semua kontak yang tersimpan. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat melakukan operasi lihat semua kontak".
7. Mendefinisikan metode **cariKontak()** untuk mencari kontak berdasarkan nama. Jika ponsel dalam keadaan mati, akan mencetak pesan "Ponsel mati. Tidak dapat melakukan operasi cari kontak".

Kelas **Contact** ini menambahkan fungsionalitas tambahan terkait manajemen kontak pada ponsel. Dengan menggunakan objek **Contact**, pengguna dapat menambahkan kontak baru, melihat semua kontak yang tersimpan, dan mencari kontak berdasarkan nama.

- **Phone.java**



Program di atas adalah sebuah interface yang bernama **Phone**. Interface ini mendefinisikan kontrak atau blueprint untuk sebuah ponsel yang memiliki beberapa fungsi dasar. Interface **Phone** ini memberikan kerangka dasar untuk mengimplementasikan fungsi-fungsi dasar yang ada pada sebuah ponsel. Kelas-kelas yang mengimplementasikan interface ini harus mengimplementasikan semua metode yang dideklarasikan dalam interface ini.