Perbaikan Nilai Prokom 2023

Mata Kuliah	: Programa Komputer
Hari / Tanggal	: Senin, 16 Juli 2022
Sifat Ujian	: Mandiri, Diperkenankan Akses Internet / Online
Dosen Pengampu	: Yusuf Priyandari, S.T., M.T., Dr. Eko Liquiddanu, S.T., M.T.
Waktu	: 120 Menit

CPMK 2 / RCPL 1-36

Mampu membuat program komputer dengan bahasa pemrograman tertentu untuk merealisasikan algoritma-algoritma penyelesaian masalah tertentu.

Peraturan Ujian.

1. Kerjakanlah ujian ini secara mandiri, **tidak meminta bantuan** atau **informasi** dari **teman atau orang lain.**
2. Bantuan pencarian solusi melalui internet masih diperkenankan asalkan tidak melanggar peraturan ujian nomor 1.
3. Upload hasil pekerjaan Anda pada folder yang disediakan di *cloud*. Nama file program utama menggunakan nim Anda dengan huruf kapital.
4. Kerjakan soal secara runut sesuai petunjuk soal agar Anda memperoleh nilai (tidak menghasilkan error). Perhatikan dengan seksama ketentuan-ketentuan yang diminta pada setiap soal.
5. Minimal Anda mampu menyelesaikan sampai nomor 3, dan lebih baik sampai menyelesaikan seluruhnya. Selamat bekerja secara mandiri.

Soal Ujian.

1. Program Komputer sederhana menggunakan python ditujukan untuk mengelola aktivitas Tabungan Sampah di sebuah dusun (desa). Tabungan sampah ini menyimpan data anggota (dari warga sekitar dusun), data harga sampah yang bisa memberikan kompensasi nilai bagi warga yang menyetornya, dan transaksi pengumpulan sampah dari anggota (sampah yang bernilai jual).
2. Buatlah sebuah modul bernama "**anggota**" untuk mengelola data warga yang terlibat dalam tabungan sampah.
   a. Di dalam modul tersebut, buatlah fungsi dengan nama **tambah_anggota**, untuk menambah data seorang anggota. Fungsi tersebut membutuhkan *argument* berupa nama, alamat, dan telepon.

      Parameter yang diterima oleh fungsi tersebut kemudian digunakan untuk membentuk data anggota. Penambahan data anggota tersebut disimpan dalam file JSON ***anggotas.json***. Isi format JSON yang diminta oleh soal ini sebagai berikut.

```
{
    "16985": {
        "idanggota": "16985",
        "nama": "Dr. Eman Mustofa, S.IP",
        "alamat": "Gg. Merdeka 12",
        "tanggal": "2023-04-03",
        "telepon": "+627962532798"
    }
}
```

Tampilan masukan data yang dilakukan oleh pengguna program hanyalah tiga hal sebagai berikut:

```
Penambahan data anggota.
Nama : _
Alamat : _
Nomor Telepon:_
```

Setelah proses penambahan berhasil dilakukan, muncul pesan:

```
Berhasil menambahkan data anggota. _
```

dan program menunggu penekanan tombol enter untuk lanjut.

Catatan:
idanggota, dibuat secara acak sebanyak 5 karakter yang bisa berupa karakter angka, huruf, atau gabungan keduanya. Tidak boleh ada idanggota yang sama tersimpan dalam file anggotas.json. Buatkan sebuah fungsi **generate_idanggota** yang memberikan nilai balik idanggota dalam tipe data string.
tanggal, berisi tanggal pada saat dilakukan penambahan data dengan format "yyyy-mm-dd".

Modul dapat diuji/dijalankan dari file program modul tersebut untuk penambahan data anggota.

b. Buatlah fungsi untuk mencari data seorang anggota berdasarkan idanggota pada file anggotas.json. Nama fungsi tersebut adalah **cari_anggota_by_id**.
Sebelum memanggil fungsi tersebut, program menampilkan sebagai berikut:

```
Pencarian data anggota
Masukkan ID Anggota :
```

Dengan demikian, fungsi membutuhkan *argument* berupa idanggota.
Jika menemukan, fungsi akan memberikan *nilai balik* berupa data anggota berbentuk *dictionary* seperti contoh ini: `{'idanggota': '13306', 'nama': 'R. Yusuf Rahmawati', 'alamat': 'Jl. Cihampelas No. 131\nPurwokerto, RI 51978', 'tanggal': '2023-01-05', 'telepon': '+624712437732'}`
Jika tidak menemukan, diberikan *nilai balik* berupa *dictionary* kosong {}.

c. Buatlah fungsi **tampilkan_anggota** untuk menampilkan data seorang anggota hasil pencarian yang telah dilakukan pada soal "b" di atas.
Fungsi tampilkan_anggota membutuhkan *argument* berupa *dictionary* keluaran dari fungsi cari_anggota_by_id. Fungsi ini tidak memberikan nilai balik.
Jika hasil pencarian sebelumnya menemukan data anggota berbentuk *dictionary*, maka fungsi ini menampilkan hasil sebagai berikut:

```
ID Anggota :  13306
Nama :  R. Yusuf Rahmawati
Alamat : Jl. Cihampelas No. 131
Purwokerto, RI 51978
Telepon :  +624712437732
Tanggal Daftar :  2023-01-05
```

Jika hasil pencarian sebelumnya berupa *dictionary* {} kosong, maka fungsi ini akan menampilkan pesan:

```
Tidak ada data anggota !
```

d. Buatlah fungsi *edit_anggota* untuk mengedit data seorang anggota. Seluruh tampilan dan proses penyimpanan pengubahan data anggota berada dalam fungsi, sehingga fungsi tidak membutuhkan argument untuk digunakan.

   Berikut ini tampilan ketika fungsi dipanggil.

   ```
   Ketik ID anggota yang akan diedit : _
   ```

   Gunakan fungsi cari_anggota_by_id untuk memperoleh data anggota dalam file anggotas.json yang akan diedit tersebut. Jika ditemukan, maka program akan menampilkan **satu persatu baris** di bawah ini. Misalkan pengguna bermaksud mengubah `{nama awal}`, maka di bagian setelah tanda panah bisa diketikkan perubahan nama. Setelah ditekan tombol Enter, maka muncul baris berikutnya yang berisi `{alamat awal}` sehingga pengguna bisa mengetikan alamat baru dibagian setelah tanda panah, begitu seterusnya. Namun, jika misal `{nama awal}` tidak akan diubah, maka pengguna cukup menekan tombol Enter saja.

   ```
   Nama : {nama awal} -> _
   Alamat : {alamat awal} -> _
   Telepon : {telepon awal} -> _
   ```

   Catatan tampilan: `{nama awal}`, `{alamat awal}`, dan `{telepon awal}` di atas diganti dengan data nama, alamat dan telepon yang sesuai dengan idanggota yang diperoleh dari hasil pencarian. Setelah berhasil menyimpan file hasil editan *anggotas.json*, maka ditampilkan

   ```
   Data berhasil diubah.
   ```

   Dan setelahnya program selesai.

   Jika idanggota yang diketikkan oleh pengguna, ternyata tidak ada dalam data, maka ditampilkan :

   ```
   Data anggota tidak ditemukan !
   Cari lagi (Y/y = Ya, T/t = Tidak)?
   ```

   Jika dijawab Y/y atau Enter, maka **program membersihkan layar dan kembali ke awal menanyakan ID Anggota yang akan diedit**. Sebaliknya, jika dijawab t/T maka program selesai atau jika nantinya modul dipakai pada program utama maka akan kembali ke menu utama.

e. …

3. Buatlah menu pada program utama untuk menggunakan modul dan fungsi-fungsi yang Anda buat.

   ```
   ========================================
   ** Program Pengelolaan Tabungan Sampah **
   ========================================
   Pilihan menu :
   1. Pengelolaan Keanggotaan
      1a. Penambahan Data Anggota
      1b. Pencarian Data Anggota
      1c. Pengubahan Data Anggota
   9. Exit
   Masukkan pilihan Anda : _
   ```

   a. Perhatikan, menu akan kembali ditampilkan setelah menyelesaikan sebuah proses dengan tampilan pada proses-proses sebelumnya telah dibersihkan dari layer *console / terminal*.
   b. Gunakan modul dan fungsi yang telah dibuat sebelumnya.

4. Buatlah sebuah modul bernama "**tabungansampah**" untuk mengelola data transaksi setoran sampah yang bernilai jual dari warga (anggota) ke lokasi pengolahan sampah. Sebagaimana modul anggota, modul ini setidaknya minimal memiliki proses penambahan transaksi tabungan sampah dan menampilkan tabel tabungan sampah seorang anggota
   a. Di dalam modul tersebut, buatlah fungsi dengan nama **tambah_tabungan**, untuk menambah data transaksi setoran sampah seorang anggota.

      Fungsi tersebut membutuhkan *argument* berupa idanggota, kode sampah yang ditabung, dan kuantitas. Parameter yang diterima oleh fungsi tersebut kemudian digunakan untuk membentuk data tabungan. Penambahan data tabungan tersebut disimpan dalam file JSON ***tabungan***13969***.json***. Perhatikan kode dibelakang kata "tabungan" adalah IDAnggota. Isi format ***tabungan***13969***.json*** yang diminta oleh soal ini sebagai berikut.

```
[
   {
        "tanggal": "2023-06-15",
        "idtransaksi": "2590151",
        "tipetransaksi": "K",
        "sampah": "1",
        "kuantitas": 1.2,
        "nilaisatuan": 100,
        "total": 1200,
        "saldo": 120000
    }
]
```

      Tampilan masukan data yang dilakukan oleh pengguna program sebagai berikut (boleh tidak terlalu rapi tampilannya):

```
Penambahan Tabungan Sampah.
Input ID Anggota : _
=============================================
IDAnggota : {idanggota}| Nama   : {nama}                  |
Telepon   : {telepon}  | Alamat: {alamat}
=============================================
---------------------------------------------------------------------------------------------
Kode| Jenis Sampah      | Harga Satuan (Rp)
---------------------------------------------------------------------
1   | Kardus            | 500
2   | Botol plastic     | 300
3   | Logam besi        | 800
4   | Tembaga           | 950
---------------------------------------------------------------------
Pilih jenis sampah     : _
Kuantitas sampah       : _
```

      Ketika ID Anggota ditemukan, ditampilkanlah data anggota dengan format seperti di atas yang dilanjutkan menampilkan jenis sampah yang bisa diterima oleh tabungan sampah (file tersedia pada ***produksampah.json***), dan dilanjutkan dengan permintaan mengisi kode jenis sampah dan kuantitas yang ditabung.

      Ketika ID Anggota tidak ditemukan, akan ditampilkan pesan

```
Data anggota tidak ditemukan !
Cari lagi (Y/y = Ya, T/t = Tidak)?
```

Ketika pengguna bermaksud mencari lagi data anggota, pengguna dapat menekan Enter langsung atau mengisikan dengan Y atau y, maka program kembali bersih dan kembali menanyakan input ID Anggota seperti di atas.

Kecuali ketika pengguna menekan tombol **"t atau T"**, maka modul selesai dieksekusi. Nantinya dalam implementasi, akan kembali ke halaman menu utama. Catatan: Anda boleh mengubah parameter pengulangan menggunakan mekanisme yang Anda bisa (Tidak harus y/Y atau t/T)

Setelah mengisikan jenis sampah (diasumsikan pengguna tidak pernah salah memasukkan kode jenis sampah) dan kuantitas sampah, maka ketika program **berhasil menambahkan data transaksi tabungan sampah**, muncul pesan:

```
Pencatatan transaksi tabungan sampah berhasil.
```

Kemudian dilanjutkan dengan tampilan yang menanyakan apakah ada jenis sampah lain yang akan ditabung

```
Ada jenis sampah lain akan ditabung (Y/Y=Ya, T/t=Tidak) ? : _
```

Jika Ya, maka ditampilkan kembali seperti **contoh di bawah ini**. Catatan: Anda boleh mengubah parameter pengulangan menggunakan mekanisme yang Anda bisa (Tidak harus y/Y atau t/T)

```
==============================================
IDAnggota : {idanggota}| Nama  : {nama}                  |
Telepon   : {telepon}  | Alamat: {alamat}
==============================================
----------------------------------------------------------------------------------------------------
Kode| Jenis Sampah      | Harga Satuan (Rp)
--------------------------------------------------------------------
1   | Kardus            | 500
2   | Botol plastic     | 300
3   | Logam besi        | 800
4   | Tembaga           | 950
--------------------------------------------------------------------
Pilih jenis sampah      : _
Kuantitas sampah        : _
```

Jika Tidak, maka penggunaan fungsi tambah_tabungan pada modul **tabungansampah** selesai. Jika dalam implementasi modul pada program utama nantinya akan kembali ke menu utama

Catatan:
tanggal, berisi tanggal pada saat dilakukan penambahan data format "yyyy-mm-dd".

idtransaksi, dibuat secara acak sebanyak 7 karakter (misal 2590151) yang bisa berupa karakter angka, huruf, atau gabungan keduanya. Tidak boleh ada idtransaksi yang sama dalam sebuah file JSON milik tabungan seorang anggota. Untuk menghasilkan idtransaksi yang unik pada setiap file tabungan seorang anggota, buatkan fungsi **generate_idtransaksi** yang memiliki parameter berupa idanggota, atau menerima argument berupa idanggota. Luaran fungsi tersebut adalah idtransaksi bertipe string.

tipetransaksi, "K" jika proses menabung, dan "D" jika proses menarik tabungan.

sampah, jika proses menabung, diisi dengan kode sampah yang ditabung. Isikan dengan "0" jika proses menarik tabungan.
kuantitas, jika proses menabung, diisi dengan kuantitas sampah yang ditabung. jika proses menarik tabungan, isikan dengan 1.
nilaisatuan, jika proses menabung, diisi dengan harga satuan yang sesuai dengan kode sampah yang ditabung yang tersimpan pada *produksampah.json*. Buatkan fungsi dengan nama **cari_harga** untuk mendapatkan hargasatuan sampah dari kode yang diinputkan pengguna. Fungsi ini membutuhkan argument berupa kode atau idsampah, kemudian memberikan nilai balik berupa harga dalam tipe float. Jika proses menarik tabungan, nilaisatuan ini adalah dengan besaran nilai tabungan yang ditarik oleh pengguna.

total, jika proses menabung, diisi dengan hasil perkalian antara kuantitas dengan harga satuan untuk kode sampah yang ditabung (atau nilaisatuan). Jika proses menarik tabungan, isikan dengan besaran nilai tabungan yang ditarik oleh pengguna.

saldo, jika proses menabung, diisi dengan nilai saldo terakhir ditambah dengan total, dan sebaliknya jika proses menarik tabungan maka disikan dengan nilai saldo terakhir dikurang dengan total yang ditarik oleh pengguna. Untuk mendapatkan saldo pada transaksi terakhir seorang anggota, buatkan fungsi **get_saldo** dengan parameter berupa idanggota.

Modul dapat diuji/dijalankan dari file program modul atau dijalankan dari program utama.

b. <>

5. Perbarui program utama sehingga bisa menampilkan menu sebagai berikut.

```
** Program Pengelolaan Tabungan Sampah **
=========================================
Pilih menu :
1. Pengelolaan Keanggotaan
   1a. Penambahan Data Anggota
   1b. Pengubahan Data Anggota
   1c. Pencarian Data Anggota
2. Pengelolaan Tabungan Anggota
   2a. Penambahan Tabungan
   2b. Penarikan Tabungan
   2c. Menampilkan Data Tabungan
3. Exit

Masukkan pilihan Anda : _
```

a. Ujilah program utama Anda dengan melakukan proses "Penambahan Tabungan" (2a) yang mengakses modul dan fungsi yang telah dibuat sebelumnya.

6. Lanjutkan pembuatan program untuk menyelesaikan proses penarikan tabungan (2b). Informasi mengenai proses penarikan tabungan secara umum telah disampaikan pada soal Nomor 4.
   a. Biasakan menggunakan modul dan fungsi.
   b. …

7. Lanjutkan pembuatan program untuk menyelesaikan proses menampilkan tabungan (2c) seorang anggota.
   a. Biasakan menggunakan modul dan fungsi.
   b. Program utama yang diperbarui ini ketika dijalankan menampilkan menu sebagai berikut:

```
==========================================
IDAnggota : {idanggota}| Nama  : {nama}                  |
Telepon   : {telepon} | Alamat: {alamat}
==========================================
Tanggal Transaksi Terakhir    : {tanggal}
Kode Transaksi Terakhir       : {idtransaksi}
Jenis Transaksi Terakhir      : {tipetransaksi} (Ubah menjadi Tabungan /
Penarikan )
Nilai Transaksi Terakhir      : {total}
Saldo Tabungan                : {saldo}
```

   c. Seluruh eksekusi program harus menggunakan modul-modul dan fungsi yang telah dbuat pada jawaban soal-soal sebelumnya.
   d. Jika telah menyelesaikan suatu tugas, program akan membersihkan tampilan dan kembali menampilkan halaman utama program seperti pada bagian soal Nomor 5.
8. Note tambahan untuk perbaikan nilai :
   Ditambah penggunaan
   a. 1. metode pengurutan bubble sort untuk mengurutkan data anggota
   b. 2. pelaporan2 transaksi