TUGAS BESAR

ALGORITMA DAN STRUKTUR DATA

CASE 2

Disusun oleh:

Farhan Nanda Prihamudi (1206210006)

M. Sirojul M (1206210008)

Rendika Nurtanto Suhartanto (1206210011)

Ryanta Meylinda Savira (1206210014)

PROGRAM STUDI SAINS DATA

FAKULTAS TEKNOLOGI INFORMASI DAN BISNIS

INSTITUT TEKNOLOGI TELKOM SURABAYA

2022

KATA PENGANTAR

Puji syukur kami ucapkan kehadirat Allah SWT atas segala rahmat-Nya sehingga laporan tugas besar ini dapat tersusun sampai selesai dengan baik. Laporan ini dibuat dengan tujuan memenuhi UAS pada Mata Kuliah Algoritma dan Struktur Data. Selain itu, penyusunan laporan ini bertujuan untuk menambah wawasan dan pengetahuan kepada pembaca.

Penulis menyampaikan ucapan terima kasih kepada Ibu Regita Putri Permata, S.Stat., M.Stat. selaku dosen mata kuliah Algoritma dan Struktur Data. Berkat tugas yang diberikan ini, kami dapat menambah wawasan berkaitan dengan tugas yang diberikan. Penulis juga mengucapkan terima kasih yang sebesarnya kepada semua pihak yang membantu dalam proses penyusunan makalah ini.

Penulis menyadari bahwa dalam penyusunan dan penulisan masih melakukan banyak kesalahan. Oleh karena itu penulis memohon maaf atas kesalahan, Penulis memohon kritik dan saran kepada para pembaca, agar penulisan makalah ini akan lebih baik lagi kedepannya.

Surabaya, 11 Juli 2022

Penyusun

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Python adalah bahasa pemrograman yang ditafsirkan untuk tujuan umum. Tidak seperti bahasa lain yang sulit dibaca dan dipahami, python lebih berfokus pada keterbacaan kode untuk membuat sintaks lebih mudah dipahami. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun mereka yang sudah mahir dalam bahasa pemrograman lain. sudah didefinisikan dan dideklarasikan dalam bahasa pemrograman tertentu (misalnya char, integer, string, dll.).

Dalam pemrograman struktur data, stack dan antrian adalah dua jenis struktur data non-primitif dengan tipe data abstrak yang digunakan untuk menyimpan item data baik dalam array maupun dalam daftar string berdasarkan struktur data kejadian nyata. Stack juga dikenal sebagai stack dimana data dapat dimasukkan dan diambil hanya dari satu sisi. Oleh karena itu, stack bersifat LIFO (Last In First Out). Tumpukan dapat dipahami sebagai kumpulan data seolah-olah data ditempatkan di atas data lain.

Queue juga dikenal sebagai antrian di mana data masuk di satu sisi dan keluar di sisi lain. Oleh karena itu, sifat queue yaitu FIFO (First In First Out). Pengertian Queue adalah sekumpulan data dimana penambahan item hanya dapat dilakukan pada salah satu ujung yang disebut sisi belakang (rear) dan penghapusan (retrieving item) dilakukan pada ujung yang lain. Contoh queue paling sederhana dapat dilihat pada antrian. Prinsip pengoperasian queue adalah prinsip first in first out (FIFO) atau first in first out. Sedangkan prinsip “last in first” atau “last in first” (LIFO), digunakan pada stack atau tumpukan. Dalam queue, ada pintu masuk di satu ujung dan pintu keluar di ujung lainnya. Lalu ada pointer yang menggunakan awal dan akhir.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka kami menemukan beberapa permasalahan yang akan menjadi bahasan pada penyusunan makalah ini, diantaranya sebagai berikut :

1. Apa yang dimaksud dengan stack, queue, dan OOP ?
2. Bagaimana cara kerja OOP ?
3. Bagaimana pendeklarasian OOP?
4. Bagaimana operasi stack dan queue pada data antrian yang sudah tersedia?

1.3 Tujuan

Tujuan yang didapat adalah untuk memudahkan pengertian dan pemahaman mengenai stack, queue, linked list dan OOP (Object Oriented Programming) secara teori dan praktek sehingga diharapkan nantinya pembaca dapat memahami dengan sangat baik mengenai praktek pemrograman beserta langkah dalam menyelesaikan permasalah yang terjadi. Serta untuk memenuhi tugas pada mata kuliah algoritma dan struktur data.

1.4 Manfaat Penelitian

Penelitian dilakukan karena adanya masalah yang perlu dipecahkan. Keuntungan praktis membantu memecahkan masalah dengan cara yang praktis. Ini bertujuan untuk berkontribusi pada pengembangan teori tentang pemahaman dan metode stack, queue, linked list, dan OOP (Pemrograman Berorientasi Objek).

1.5 Batasan Masalah

Berdasarkan latar belakang penelitian dan identifikasi masalah dapat diidentifikasi bahwa hasil pembelajaran Algoritma Struktur Data berupa kemampuan memberikan analisis stack, queue, linked list, dan OOP (Object Oriented Programming) dalam data yang dapat diperbaiki dan dianalisis melalui beberapa upaya, yaitu menganalisis data antrian yang sudah tersedia menggunakan cara stack, queue, dan linked list, ataupun juga dapat menggunakan OOP (Object Oriented Programming) untuk memudahkan menyelesaikan masalah.

BAB II

TINJAUAN PUSTAKA

2.1 Pengertian

2.1.1 OOP

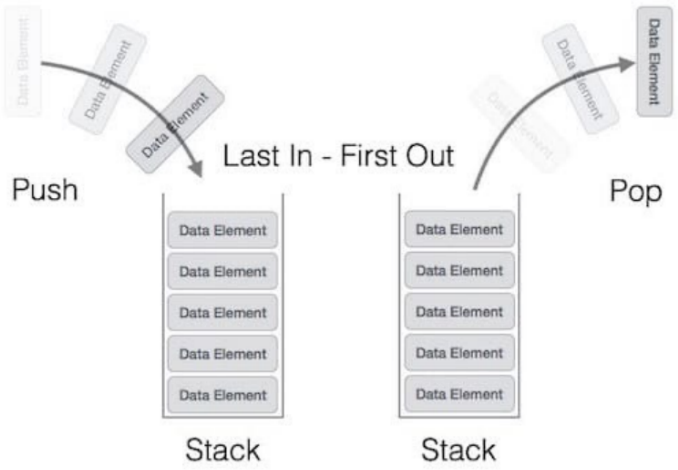
Menurut artikel Gillis dan Lewis, pemrograman berorientasi objek atau OOP didefinisikan sebagai pemrograman berorientasi objek. Ini adalah model pemrograman komputer dengan pengaturan desain perangkat lunak data atau objek-sentris. Tidak seperti pemrograman lain yang berfokus pada fungsi dan logika, OOP berfokus pada objek atau data yang memiliki atribut atau perilaku unik. Fokus OOP pada objek adalah apa yang dapat dilakukan pengembang dan berguna untuk pemrograman yang besar dan kompleks. Pendekatan OOP ini mudah diperbaiki dan dipelihara.

OOP juga dapat didefinisikan sebagai paradigma pemrograman berdasarkan konsep kelas dan objek. Paradigma ini digunakan untuk menyusun program perangkat lunak menjadi potongan kode atau kelas kode yang sederhana dan dapat digunakan kembali. Kelas-kelas ini akan digunakan nanti dalam pemrograman untuk membuat objek individu.

OOP dapat digunakan dalam berbagai bahasa pemrograman seperti JavaScript, C, Java, dan Python. Anda dapat menggunakan kelas dalam pemrograman OOP untuk menentukan atribut yang dimiliki instance objek, seperti warna. Secara umum, OOP lebih sederhana karena pengembang dapat fokus bekerja dengan objek daripada logika atau fungsi.

2.1.2 Stack

Stack atau tumpukkan adalah kumpulan data yang berada di atas data lain. Oleh karena itu, stack merupakan struktur data yang menggunakan konsep LIFO (last in first out). Di stack, item yang didorong terakhir adalah item yang pertama kali muncul.



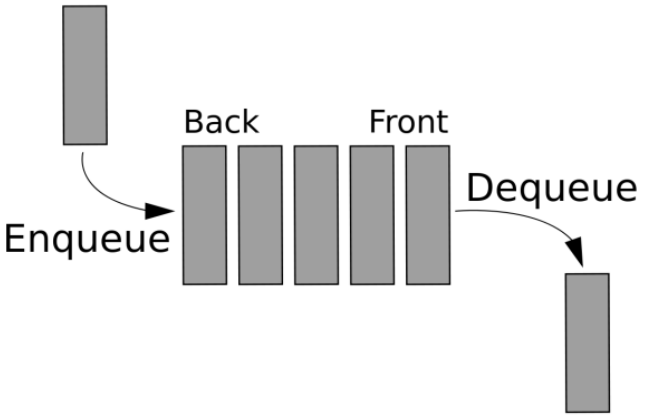
CONTOH STACK :

- Setumpuk koran, dimana koran yang paling terakhir ditambahkan dan ditaruh diatas adalah koran yang dapat dilihat
- Seseorang yang sedang mencuci piring, piring yang dicuci pertama pasti akan diletakan dibawah dan akan terus berlanjut sampai stack piring yang terakhir dicuci. Lalu, piring pasti akan di taruh di rak piring dan pasti yang diambil adalah piring yang paling atas yaitu piring yang terakhir dicuci dan yang pertama dicuci pasti akan terakhir.
- stack Batu bata yang sedang diturunkan dari mobil pasti yang diambil adalah batu bata yang paling atas, padahal batu bata pertama yang dimasukkan kedalam mobil adalah batu bata yang berada dibawahnya.

2.1.3 Queue

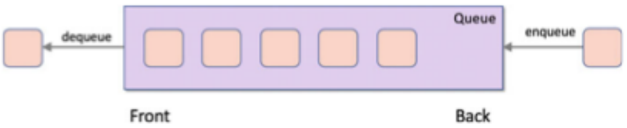
Queue adalah struktur data yang memiliki sifat FIFO (first in, first out) yang memiliki operasi yang bisa dilakukan adalah Enqueue, Dequeue.

- Enqueue merupakan proses penambahan item. Penambahan item dilakukan pada index yang terakhir.
- Dequeue merupakan proses pengambilan item. Pengambilan item dilakukan pada index awal.



Queue ADT dibuat dari sebuah list yang dibatasi sifat-sifat dan aksi / operasi yang dilakukan pada list tersebut. Aksi / Operasi yang bisa dilakukan dengan Queue ADT:

- isEmpty(), untuk mengecek apakah Queue sedang kosong atau tidak
- enqueue(x), untuk melakukan insert data x ke Queue
- dequeue(), untuk melakukan delete 1 data di Queue yang berada di paling depan
- front(), untuk melihat data di Queue yang berada di paling depan
- size(), untuk melihat jumlah data / ukuran dari sebuah Queue



2.2 PRINSIP OOP

2.2.1 Encapsulation

Prinsip enkapsulasi atau kapsulisasi dalam OOP dilakukan ketika setiap objek dalam pemrograman dapat mempertahankan keadaan privat dari kelas cetak biru. Dengan demikian, tidak ada objek lain yang dapat mengakses status objek ini secara langsung. Namun, objek mengelola statusnya sendiri melalui fungsi publik ini, sehingga objek lain masih dapat memanggil daftar fungsi publik.

Enkapsulasi adalah klasifikasi umum dari logika objek. Ini seperti menempatkan semua logika yang terkait dengan kelas B ke dalam variabel pribadi yang terkait dengan B. Misalnya kelas macan, logika yang terdapat di kelas ini adalah mengaum, belang, karnivora, dll, tergantung sifat B. Namun, mungkin ada variabel publik yang secara umum diterima oleh objek dari kelas lain, seperti makan. , Pelacakan, eksekusi, dll.

2.2.2 Abstraction

Abstraction adalah perpanjangan dari enkapsulasi. Abstraction adalah proses pemilihan data dari kumpulan yang lebih besar dan menunjukkan bahwa hanya detail yang relevan yang dapat menjadi objek. Misalnya, jika Anda membuat aplikasi kencan online, Anda mungkin diminta untuk mengumpulkan semua informasi tentang pengguna aplikasi tersebut. Ini termasuk identitas, nomor kontak, makanan favorit, hobi, dan banyak lagi. Tentu saja, jumlah data ini sangat besar sehingga tidak semuanya diperlukan untuk membuat aplikasi kencan online. Oleh karena itu, untuk memilih informasi yang terkait dengan aplikasi, proses mengambil dan memilih informasi pengguna dari kumpulan besar data pengguna dilakukan. Proses penyaringan ini disebut abstraksi. Keuntungan dari abstraksi adalah pengembang dapat menerapkan informasi yang sama dari satu aplikasi ke aplikasi lain, dengan atau tanpa perubahan.

2.2.3 Inheritance

Inheritance adalah kemampuan suatu objek untuk memiliki beberapa atau bahkan semua properti objek lain dalam pemrograman OOP. Misalnya, ini bisa menjadi anak yang mewarisi sifat dari orang tuanya. Warisan adalah prinsip OOP yang dapat digunakan kembali sebagai manfaat utama objek. Sebagai contoh, dalam bahasa pemrograman Java terdapat beberapa jenis pewarisan, seperti pewarisan tunggal, multi level, hierarkis, asosiatif, dan multi level. Misalnya, objek Grape diklasifikasikan sebagai Fruit. Anggur ini kemudian memperoleh karakteristik kelas Buah seperti mengandung biji, rasa berdaging, dll. Ini mirip dengan objek lain dari kelas Buah di mana subkelas atau objek Anggur memperoleh properti inti dari kelas Buah ditambah beberapa properti unik dari objek itu sendiri.

2.2.4 Polymorphism

Polimorfisme merupakan prinsip OOP yang memberi cara pada pengembang untuk menggunakan kelas atau klasifikasi objek persis seperti induknya sehingga tidak ada kebingungan dengan tipe-tipe campuran. Hal ini disebabkan karena sub-kelas atau objek menyimpan fungsi dan metodenya sendiri secara unik.

Jika pengembang memiliki sebuah kelas umum seperti Tumbuhan, maka di dalamnya ada beberapa metode seperti bentuk daun, batang, waktu tumbuh, dan lain-lain sebagai fungsi dan metode umumnya. Sedangkan di dalam kelas Tumbuhan ada sub kelas atau objek seperti Durian, Rumput Gajah, Pakis, dan lainnya yang masing-masing memiliki ciri khas sendiri namun tetap mempunyai fungsi atau metode dari induk kelasnya yaitu Tumbuhan dan meliputi bentuk daun, batang, dan waktu tumbuh. Hal ini yang lantas disebut sebagai prinsip polimorfisme dalam OOP.

Kelebihan OOP	Kekurangan OOP
Parallel development, Dalam pengembangan paralel, tim pemrograman dapat bekerja secara mandiri setelah menyelesaikan pemrograman kelas modular. Metode ini memungkinkan OOP untuk mengurangi waktu pengembangan untuk satu kelas (class) daripada membuatnya secara individual.	Tidak efisien, bisa dikatakan OOP memang kurang efisien karena lebih banyak menggunakan beban pada prosesor komputer. Hal ini membuat OOP menjadi pilihan yang kurang efisien ketika ada batasan teknis pada perangkat pengembang.
Reusable, setelah kelas modular dibuat, pengembang dapat menggunakannya lagi dalam aplikasi atau proyek pengembangan lainnya. Hal ini membuat OOP menjadi *reusable* dan fleksibel bahkan hanya memerlukan sedikit atau tidak ada modifikasi sama sekali.	Kontrol yang ketat umumnya dapat menyebabkan kelebihan kapasitas dan kondisi di luar kendali karena manfaatnya yang dapat diskalakan. Ini terjadi ketika pembuatan kode yang besar dan membengkak terjadi dan biaya pengembangan perangkat lunak meningkat.
Scalability OOP, memungkinkan pengembang untuk mengembangkan program hanya dengan menambahkan beberapa fungsi, kelas, dan objek. Prosedur pembuatan kode di OOP juga dapat dipertahankan dan dilindungi dengan meningkatkan keamanan pemrograman. Hal ini dikarenakan validasi yang tinggi saat mengakses kode prosedur OOP.	OOP dapat diakses dengan validitas tinggi, tetapi cenderung diduplikasi, tetapi mudah dirancang tetapi sulit diimplementasikan, sehingga cenderung terduplikasi. Ini juga karena kelas modul yang sangat fleksibel dalam aplikasi. Pengembang mungkin secara tidak sengaja menerapkannya dalam proyek baru dan menemukan duplikat.

BAB III

METODOLOGI PENELITIAN

3.1 Sumber Data

Adapun sumber data dalam penyusunan laporan ini yaitu data sekunder. Data Sekunder merupakan sumber data suatu penelitian yang diperoleh peneliti secara tidak langsung melalui media perantara (diperoleh atau dicatat oleh pihak lain). Data sekunder itu berupa bukti,catatan atau laporan historis yang telah tersusun dalam arsip atau data dokumenter. Data sekunder yang digunakan yaitu berupa permasalahan yang telah diberikan.

3.2 Langkah Analisis

Langkah analisis yang diambil dalam pembuatan modul beserta penyelesaian dalam *study case* yang ada, dimana langkah pertama yaitu memahami study case permasalahan apa yang harus diselesaikan dengan berpedoman pada studi literatur yang didapat selama proses penyelesaian *study case*. Setelah melakukan studi literatur langkah selanjutnya yaitu penyelesaian *study case* dengan metode-metode yang sesuai dengan permasalahan tehadapat *study case* tersebut. Sehingga penyelesaian pada *study case* yang ada dapat ditemukan dan diselesaikan.

3.3 Alur Penelitian

BAB IV

ANALISIS DAN PEMBAHASAN

Permasalahan 2

Kamu sedang magang disebuah cabang Bank Konvensional yang belum memiliki system antrian. Banyak nasabah yang baru masuk langsung dilayani, padahal masih antrian. Buatlah sebuah antrian di bank dengan ketentuan sebagai berikut:

1. Tambahkan Antrian, Merupakan menu yang digunakan untuk memasukkan antrian nasabah baru.
2. Panggilan Antrian, Merupakan menu yang digunakan untuk memanggil antrian nasabah
3. Lihat semua daftar, menu yang digunakan untuk melihat antrian customer dan yang sedang dilayani
4. Exit

Contoh tampilan antrian bank sebagai berikut:

```
==== PROGRAM ANTRIAN BANK =====
================================
|1. Tambahkan Antrian |
|2. Panggilkan Antrian |
|3. Lihat semua daftar |
|4. Hapus Semua daftar |
|5. Exit |
================================

Pilih : 1

No. Antrian : 1
Antian yang Menunggu : 0
```

Pembahasan

Program :

```
import pandas as pd
import numpy as np
```

Import library yang diperlukan untuk pembuatan program

```
class Nasabah :
    def __init__(self) :
        self.Nasabah = []
        self.nomor_antrian = []
        global m
        m = []
```

Pada program “Antrian Nasabah Bank” ini, kelompok kami menggunakan struktur program yaitu OOP (Object Oriented Programming). Pertama kami membuat suatu class bervariabel Nasabah, lalu membuat atribut di dalamnya yakni self.Nasabah dan self.nomor_antrian dan dilanjut dengan membuat global m yaitu berisi list kosong. Global ini sebagai suatu keyword untuk membuat variabel m dan isinya menjadi dapat diakses oleh seluruh codingan yang ada pada kernel.

```
def TambahNasabah(self):
    p = int(input("Tambah Berapa Nasabah: "))
    for i in range(p):
        Antrian = len(self.nomor_antrian)
        Antrian += 1
        Nama = input('Masukkan Nama Nasabah: ')
        self.Nasabah.append(Nama)
        m.append(Antrian)
        self.nomor_antrian.append(Antrian)
    print("")
    print("Nomor Antrian Anda : 1")
    print("berhasil menambah antrian anda")
    print("Antrian yang sedang menunggu :", Antrian - 1)
```

Setelah membuat attribute, dilanjut membuat suatu method pada class Nasabah, pertama adalah untuk menu ke satu yakni def TambahNasabah(self). Disini kami membuat inputan untuk user ingin menambahkan berapa banyak nasabah bank ke antrian nasabah, lalu dimasukkan ke perulangan yang telah ada perulangan yang tampil adalah inputan Nama dan Nomor Antrian.

```
  def PanggilkanAntrian(self):
      panggilNo = int(input("Masukan nomor antrian yang ingin di panggil: "))
      a = pd.Series(self.nomor_antrian)
      b = pd.Series(self.Nasabah)
      df = pd.DataFrame([a,b], index = ("No","Nama")).T
      print("")
      if panggilNo == 1:
          print (df.iloc[0,0:])
          print("")
          print("Nomor antrian yang anda panggil adalah", panggilNo)
      elif panggilNo == 2:
          print (df.iloc[1,0:])
          print("")
          print("Nomor antrian yang anda panggil adalah", panggilNo)
      elif panggilNo == 3:
          print (df.iloc[2,0:])
          print("")
          print("Nomor antrian yang anda panggil adalah", panggilNo)
      elif panggilNo == 4:
          print (df.iloc[3,0:])
          print("")
          print("Nomor antrian yang anda panggil adalah", panggilNo)
      else:
          print("Masukan sesuai dengan banyaknya antrian")
```

Setelahnya kami membuat PanggilkanAntrian(self): sebagai menu kedua. Disini kami membuat inputan panggilNo sebagai inputan kepada user ingin memanggil antrian nomer berapa. Variabel a adalah sebagai series dari self.nomor_antrian, dan variabel b adalah series dari self.Nasabah. Setelahnya kami gabung kedua series menjadi dataframe dengan nama variabel df. Lalu kami buat percabangan untuk menampilkan output yang diinginkan, yakni jika memilih antrian 1 maka akan slicing dari df yang ada dan menampilkan urutan nomor 1 yang muncul adalah nomor antrian dan nama nasabah. jika memasukkan tidak sesuai dengan banyaknya nasabah maka akan terprint peringatan untuk memasukkan nomor sesuai dengan banyaknya data pada dataframe df.

```
def SemuaDaftar(self):
        if len(self.Nasabah) == 1 :
            print("Nasabah saat ini adalah :", self.Nasabah[0])
            print("tidak ada Nasabah selanjutnya")
        elif len(self.Nasabah) == 2:
            print("Nasabah saat ini adalah :", self.Nasabah[0])
            print("Nasabah selanjutnya :", self.Nasabah[1])
        elif len(self.Nasabah) == 3:
            print("Nasabah saat ini adalah :", self.Nasabah[0])
            print("Nasabah Kedua :", self.Nasabah[1])
            print("Nasabah Ketiga :", self.Nasabah[2])
        elif len(self.Nasabah) == 4:
            print("Nasabah saat ini adalah :", self.Nasabah[0])
            print("Nasabah kedua :", self.Nasabah[1])
            print("Nasabah ketiga :", self.Nasabah[2])
            print("Nasabah ketiga :", self.Nasabah[3])
        elif len(self.Nasabah) > 4:
            print("Nasabah saat ini adalah :", self.Nasabah[0])
            print("Nasabah kedua :", self.Nasabah[1])
            print("Nasabah ketiga :", self.Nasabah[2])
            print("Nasabah ketiga :", self.Nasabah[3])
            print("Nasabah lainnya sudah ada di-antrian")
        else:
            print("Belum Ada Antrian Silahkan Memasukkan Antrian Terlebih dahulu")
```

Kami juga membuat method SemuaDaftar dengan output menampilkan daftar yang ada pada atribut Self.Nasabah. pecabangan di buat untuk melihat jika ada 1 nasabah maka akan dijalankan percabnagan yang pertama, jika ada 4 nasabah maka akan ter-output nasabah pertama, kedua, ketiga dan keempat. namun jika nasabah lebih dari 4 maka yang tampil adalah keempat nasabah dan selebihnya akan tersimpan tidak kami tampilkan.

```
def HapusNasabah(self):
        for i in range (len(self.Nasabah)) :
            self.Nasabah.pop(0)
        print("Antrian Nasabah Telah dihapus seluruhnya")
```

untuk method ini adalah menu keempat yang dimana digunakan untuk menghapus semua daftar nasabah yang ada. dengan menggunakan self.Nasabah.pop(0) yang diulang-ulang sebanyak nasabah yang ada.

```
Bank = Nasabah()
```

Disini adalah pembuatan object dari class nasabah yang telah dibuat. dengan object yang bervariabel Bank

```
kesempatan = 2
p = 0
while p == 0:
    user = input('ID: ')
    pwd = input('Password: ')
    if (user == 'Admin') & (pwd == '12062100'):
        print("")
        print(" --- ANDA TELAH BERHASIL LOGIN ---")
        print(" --- Selamat Datang ---")
        print("")
        kondisi = 0
        while kondisi == 0:
            print("")
            print("=="*5,"PROGRAM ANTRIAN BANK", "=="*5)
            print("==="*13)
            print("Menu :\n|1. Tambahakan Antrian\t| \n|2. Panggilkan Antrian\t| \n|3. Lihat Semua
Daftar\t|\n|4. Hapus Semua Daftar\t|\n|5. Exit\t\t|")
            print("==="*13)
            pick = int(input("Masukkan pilihan (1/2/3/4/5): "))
            print("")
            if pick == 1:
                Bank.TambahNasabah()
            elif pick == 2:
                Bank.PanggilkanAntrian()
            elif pick == 3:
                Bank.SemuaDaftar()
            elif pick == 4:
                Bank.HapusNasabah()
            elif pick == 5:
                print("")
                print("PROGRAM BERHENTI")
                print("Terimakasih")
                break
            else :
                print("!!! MASUKAN NOMOR SESUAI YANG ADA PADA MENU !!!")
                continue
        break

    elif (user != 'Admin') & (pwd != '1206210011'):
        kesempatan -= 1
        if kesempatan > 0:
            print("ID atau Password salah, silahkan coba lagi")
            print('kesempatan login sisa', kesempatan,'kali\n')
        elif kesempatan == 0:
            print("ID atau Password salah")
            print('=======================COBA LAGI NANTI==============================')
            break
```

Lalu disini kelompok kami membuat program untuk memunculkan kelima menu yang ada, lalu juga menambahkan password dan user yang dimana untuk kesempatan kami beri 2 kali, jika gagal maka akan terhenti, jika benar maka akan masuk kedalam menu.

lalu ada juga kelima menu yang akan tampil ketika password dan user benar. program untuk kelima menu tersebut kami beri while agar programnya terus berulang tanpa henti, namun ketika user memilih untuk berhenti dapat dilakukan dengan cara memilih menu exit yakni nomor 5. jika user memilih nomor yang tidak ada pada menu, maka akan muncul peringatan, jika user memilih nomor sesuai menu maka akan muncul output yang diinginkan.

Output :

```
ID: Admin
Password: 12062100

 --- ANDA TELAH BERHASIL LOGIN ---
 --- Selamat Datang ---


========== PROGRAM ANTRIAN BANK ==========
=======================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=======================================
Masukkan pilihan (1/2/3/4/5): 1

Tambah Berapa Nasabah: 4
Masukkan Nama Nasabah: Rendika
Masukkan Nama Nasabah: Farhan
Masukkan Nama Nasabah: Ryanta
Masukkan Nama Nasabah: Ozam

Nomor Antrian Anda : 1
berhasil menambah antrian anda
Antrian yang sedang menunggu : 3
```

Output yang muncul jika password dan user benar, lalu kita memilih menu nomor 1 yakni Tambahan Antrian Nasabah.

```
========== PROGRAM ANTRIAN BANK ==========
=======================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=======================================
Masukkan pilihan (1/2/3/4/5): 2

Masukan nomor antrian yang ingin di panggil: 1

No           1
Nama    Rendika
Name: 0, dtype: object

Nomor antrian yang anda panggil adalah 1

========== PROGRAM ANTRIAN BANK ==========
=======================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=======================================
Masukkan pilihan (1/2/3/4/5): 2

Masukan nomor antrian yang ingin di panggil: 3

No           3
Nama    Ryanta
Name: 2, dtype: object

Nomor antrian yang anda panggil adalah 3
```

Output yang muncul jika kita memilih menu nomor 2 yakni Panggilkan Antrian.

```
========== PROGRAM ANTRIAN BANK ==========
=======================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=======================================
Masukkan pilihan (1/2/3/4/5): 3

Nasabah saat ini adalah : Rendika
Nasabah kedua : Farhan
Nasabah ketiga : Ryanta
Nasabah ketiga : Ozam
```

Output yang muncul jika kita memilih menu nomor 3 yakni lihat semua daftar antrian.

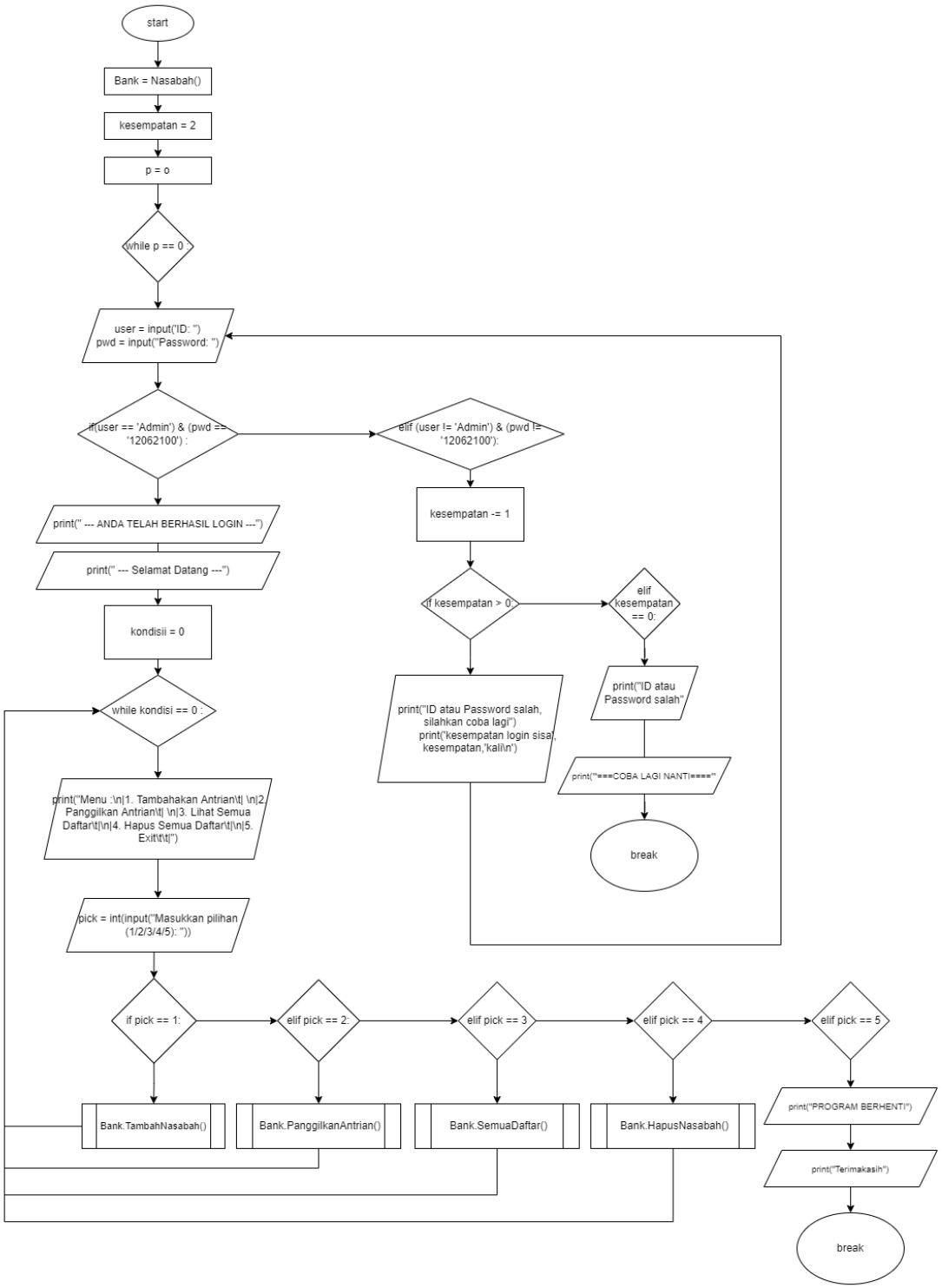
```
========== PROGRAM ANTRIAN BANK ==========
=====================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=====================================
Masukkan pilihan (1/2/3/4/5): 4

Antrian Nasabah Telah dihapus seluruhnya
```

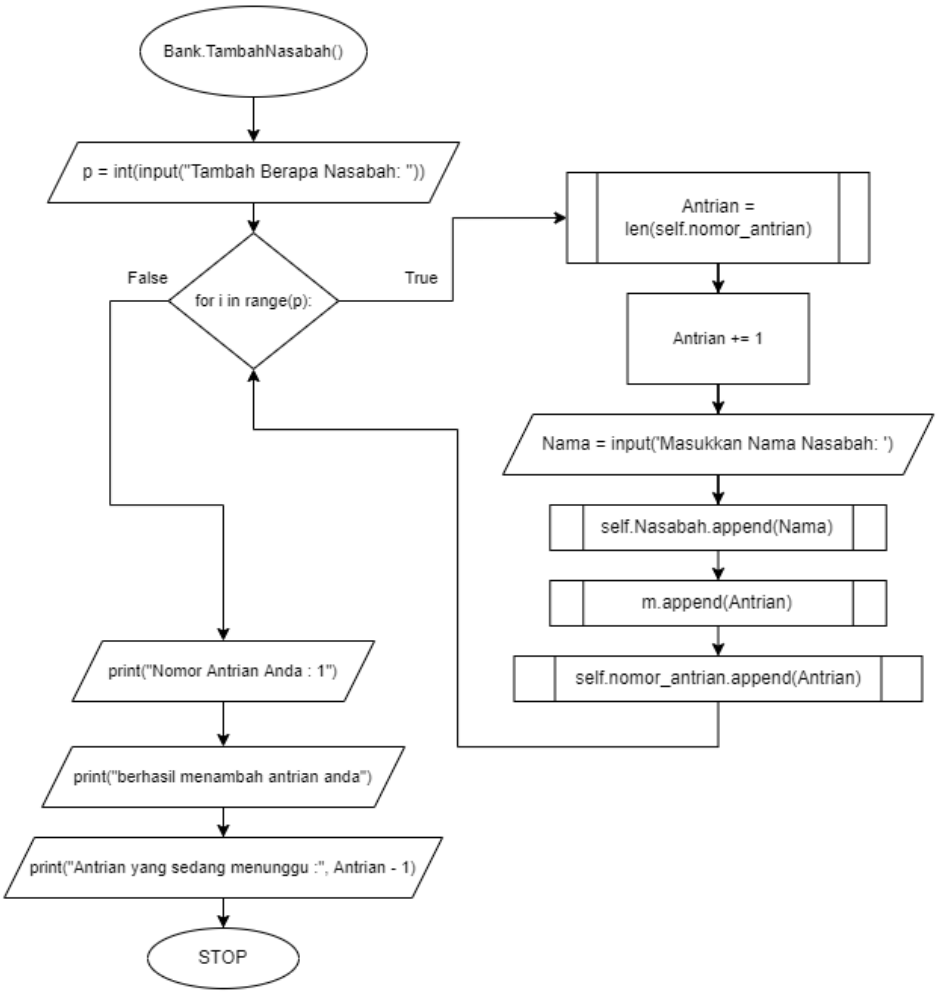
Output yang muncul jika kita memilih menu nomor 4 yakni Hapus Semua Daftar..

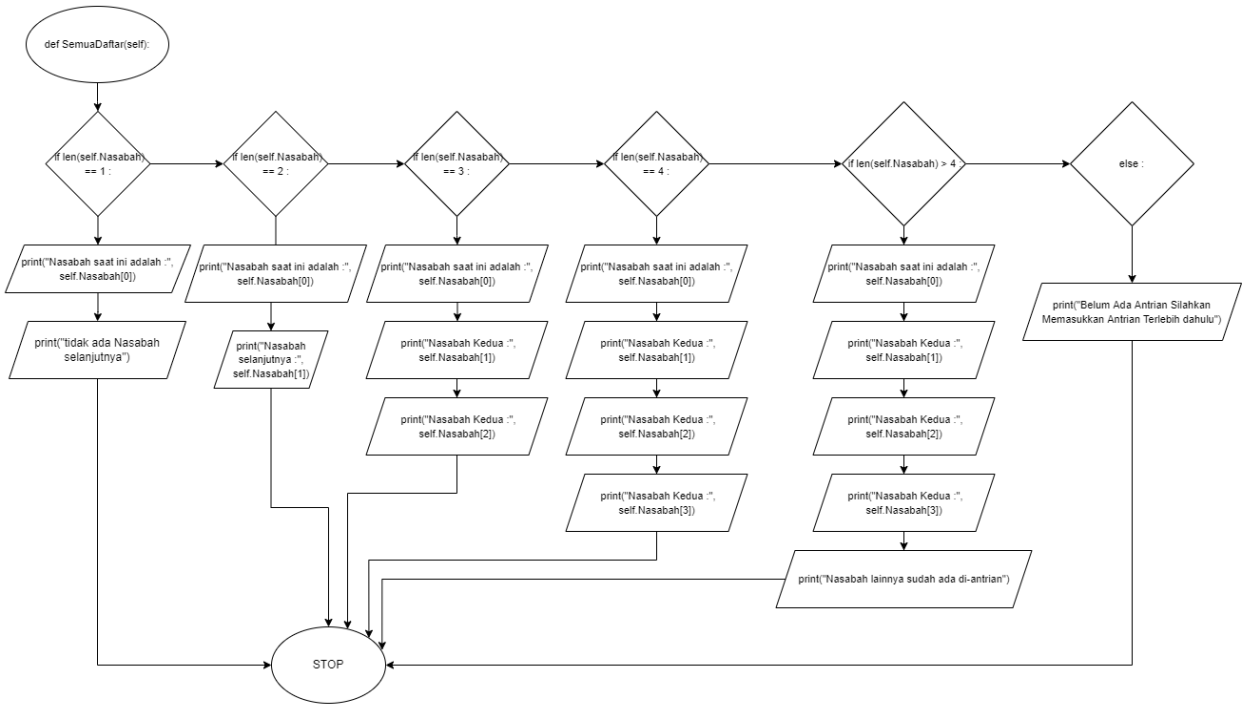
```
========== PROGRAM ANTRIAN BANK ==========
=====================================
Menu :
|1. Tambahakan Antrian  |
|2. Panggilkan Antrian  |
|3. Lihat Semua Daftar  |
|4. Hapus Semua Daftar  |
|5. Exit                |
=====================================
Masukkan pilihan (1/2/3/4/5): 5


PROGRAM BERHENTI
Terimakasih
```

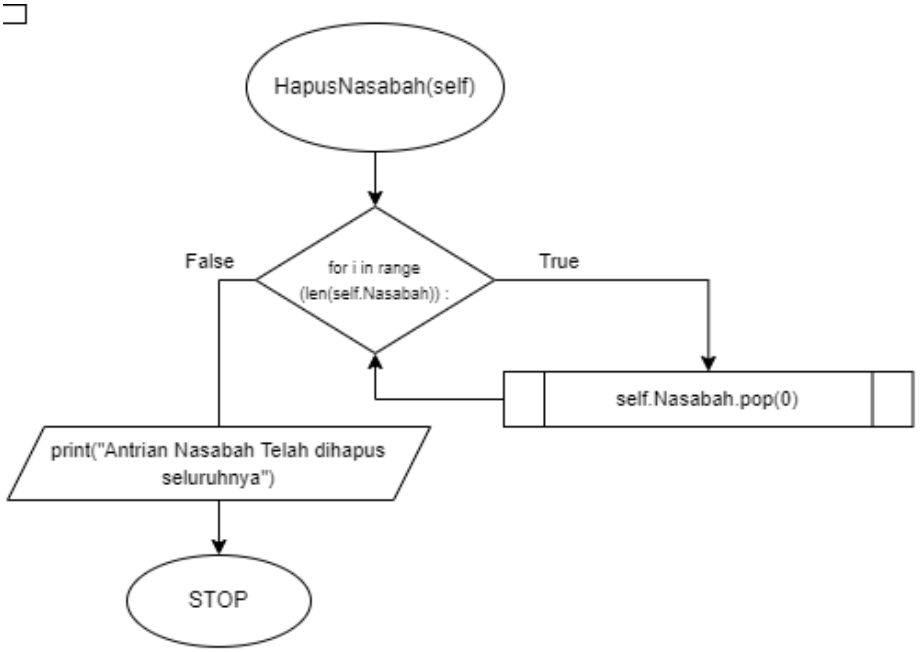
Output yang muncul jika kita memilih menu nomor 5 yakni Exit/

FLOWCHART


start
Bank = Nasabah()
kesempatan = 2
p = o
while p == 0
user = input('ID: ')
pwd = input("Password: ")
if(user == 'Admin') & (pwd == '12062100') :
elif (user != 'Admin') & (pwd != '12062100'):
print(" --- ANDA TELAH BERHASIL LOGIN ---")
print(" --- Selamat Datang ---")
kondisii = 0
kesempatan -= 1
if kesempatan > 0:
elif kesempatan == 0:
print("ID atau Password salah, silahkan coba lagi")
print('kesempatan login sisa', kesempatan,'kali\n')
print("ID atau Password salah"
print("===COBA LAGI NANTI====")
break
while kondisi == 0 :
print("Menu :\n|1. Tambahakan Antrian\t| \n|2. Panggilkan Antrian\t| \n|3. Lihat Semua Daftar\t|\n|4. Hapus Semua Daftar\t|\n|5. Exit\t\t|")
pick = int(input("Masukkan pilihan (1/2/3/4/5): "))
if pick == 1:
elif pick == 2:
elif pick == 3
elif pick == 4
elif pick == 5
Bank.TambahNasabah()
Bank.PanggilkanAntrian()
Bank.SemuaDaftar()
Bank.HapusNasabah()
print("PROGRAM BERHENTI")
print("Terimakasih")
break


Bank.TambahNasabah()
p = int(input("Tambah Berapa Nasabah: "))
for i in range(p):
False
True
Antrian = len(self.nomor_antrian)
Antrian += 1
Nama = input('Masukkan Nama Nasabah: ')
self.Nasabah.append(Nama)
m.append(Antrian)
self.nomor_antrian.append(Antrian)
print("Nomor Antrian Anda : 1")
print("berhasil menambah antrian anda")
print("Antrian yang sedang menunggu :", Antrian - 1)
STOP


def SemuaDaftar(self):
if len(self.Nasabah) == 1 :
if len(self.Nasabah) == 2 :
if len(self.Nasabah) == 3 :
if len(self.Nasabah) == 4 :
if len(self.Nasabah) > 4
else :
print("Nasabah saat ini adalah :", self.Nasabah[0])
print("tidak ada Nasabah selanjutnya")
print("Nasabah selanjutnya :", self.Nasabah[1])
print("Nasabah Kedua :", self.Nasabah[1])
print("Nasabah Kedua :", self.Nasabah[2])
print("Nasabah Kedua :", self.Nasabah[3])
print("Nasabah lainnya sudah ada di-antrian")
print("Belum Ada Antrian Silahkan Memasukkan Antrian Terlebih dahulu")
STOP


HapusNasabah(self)
for i in range
(len(self.Nasabah)) :
False
True
self.Nasabah.pop(0)
print("Antrian Nasabah Telah dihapus
seluruhnya")
STOP

BAB V

PENUTUP

5.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan penelitian yang dilakukan oleh peneliti diatas dapat disimpulkan bahwa :

1. Pemrograman OOP adalah suatu cara mengorganisasi program dengan memodelkan objek-objek seperti benda, sifat, sistem dan lainnya, ke dalam sebuah bahasa pemrograman.
2. Beberapa konsep OOP dasar, antara lain Encapsulation (Class and Object), Inheritance (Penurunan sifat), Polymorphisme, Access Modify, Constructor, Destructor, Static Properties, Instance Variable dan Abstract Class.
3. Istilah-istilah yang digunakan pada OOP adalah Class, Class Variable, Data Member, Function Overloading, Instance Variable, Inheritance, instance, Instantiation, Method, Object dan Operator Overloading.
4. Queue adalah struktur data yang memiliki properti FIFO (first in, first out). Operasi yang dapat dilakukan adalah enqueue dan dequeue.

5.2 Saran

Penulis menyarankan agar peneliti selanjutnya meneliti lebih banyak mengenai penggunaan dan pengimplementasian dari OOP dan Queue. Penulis juga menyarankan agar peneliti selanjutnya dapat mengupas lebih dalam proses pemrograman dari permasalahan kasus 2 yang telah diberikan. Sehingga output yang diberikan lebih banyak lagi dan lebih baik lagi.

BAB V1

DAFTAR PUSTAKA

Alexander S, Gills. 2021. ***object-oriented programming (OOP).Technical Writer and Editor. Sarah Lewis.*** **Tech-Target.**

generatedlink:https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP#:~:text=Object%2Doriented%20programming%20(OOP)%20is%20a%20computer%20programming%20model,has%20unique%20attributes%20and%20behavior

Patrick, Wibowo. 2021. ***Apa Itu Bahasa Pemrograman Python?.Digital Ekonomi. Puri, Mei.*** **Warta Ekonomi.**

generatedlink:https://www.wartaekonomi.co.id/read366664/apa-itu-bahasa-pemrograman-python

Rapeti, Charan. 2020. ***Stack-Queues-Linkedlist. UNKNOWN.*** **Stack Overflow**

generatedlink:https://stackoverflow.com/questions/32151392/stacks-queues-and-linked-lists