LAPORAN

Struktur Data Linear
Praktikum 9 : Stack dan Queue

NAMA : Johanes Yogtan Wicaksono Raharja

NIM : 215314105

Program Studi INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS SANATA DHARMA

1. **Stack**
   - Penjelasan Program Stack
     Stack adalah salah satu struktur data yang digunakan untuk menyimpan sekumpulan objek ataupun variabel. Karakteristik dari stack sendiri adalah LIFO (last in first out), yang artinya data terakhir yang masuk adalah data yang akan keluar terlebih dahulu. Oleh karena itu, seperti namanya, objek yang terkumpul terlihat seperti tumpukan.

   - Screenshot Listing Program

```
1  package Implementasi;
2  import java.util.Arrays;
3  import java.util.EmptyStackException;
4  import java.util.NoSuchElementException;
5
6  public class MyArrayStack<E> implements Stack<E> {
7
8      private E[] elemen;
9      private int top;
10     private static final int DEFAULT_CAPACITY = 5;
11
12     public MyArrayStack() {
13         elemen = (E[]) new Object[DEFAULT_CAPACITY];
14         top = -1;
15     }
16     public boolean isEmpty() {
17         return (elemen == null || top <= -1);
18     }
19     public void makeEmpty() {
20         top = -1;
21         for (int i = 0; i < elemen.length; i++) {
22             elemen[i] = null;
23         }
24     }
25     public E top() {
26         if (isEmpty()) {
27             return (E) "Tumpukan Sedang Kosong";
28         } else {
29             return elemen[top];
30         }
31     }
32     public void pop() {
33         if (isEmpty()) {
34             System.out.println("Tumpukan Sedang Kosong");
35         } else {
36             elemen[top] = null;
37         }
38         top--;
39     }
40     public E topAndPop() {
41         if (isEmpty()) {
42             return (E) "Tumpukan Sedang Kosong";
43         } else {
44             E free = elemen[top];
45             pop();
46             return free;
47         }
48
49     }
```

```
    public void push(E x) {
        top++;
        if (isEmpty()) {
            System.out.println("Array Sedang Kosong");
        } else {
            if (top == elemen.length - 2) {
                doubleArray();
            } else {
                elemen[top] = x;
            }
        }
    }
    private void doubleArray() {
        if (isEmpty()) {
            System.out.println("Tumpukan Sedang Kosong");
        } else {
            elemen = Arrays.copyOf(elemen, elemen.length * 2);
        }
    }
    public void cetak() {
        for (int i = 0; i < elemen.length; i++) {
            if (elemen[i] != null) {
                System.out.println(elemen[i] + " ");
            }

        }
        if (isEmpty()) {
            System.out.println("Tumpukan Sedang Kosong");
        }
    }
    public static void main(String[] args) {
        MyArrayStack<Integer> a = new MyArrayStack<>();
        a.push(9);
        a.push(10);
        a.cetak();
        a.doubleArray();
        a.cetak();
        a.makeEmpty();
        a.cetak();

    }
}
```

- Penjelasan tiap Method

```
public MyArrayStack() {
    elemen = (E[]) new Object[DEFAULT_CAPACITY];
    top = -1;
}
public boolean isEmpty() {
    return (elemen == null || top <= -1);
}
```

```
public boolean isEmpty() {
    return (elemen == null || top <= -1);
}
public void makeEmpty() {
    top = -1;
    for (int i = 0; i < elemen.length; i++) {
        elemen[i] = null;
    }
}
```

Konstruktor kelas tanpa parameter,berfungsi untuk memberi nilai default apabila tidak diisi. Dilanjutkan dengan metode isEmpty() untuk mengembalikan data apabila elemen ksoong maupun kurang dari -1. Dilanjutkan dengan makeEmpty dengan melakukan perulangan yang bersifat dinamis yang dimana disitu setiap perulangan array diisikan dengan kosong.

```
public E top() {
    if (isEmpty()) {
        return (E) "Tumpukan Sedang Kosong";
    } else {
        return elemen[top];
    }
}
```

Sebuah meotde to() yang digunakan untuk menambahkan sebuah data. Disini dilakukan sebuah pengecekan jika metode yang dipanggil yaitu kosong maka akan mengambalikan kalimat kosong, sedangkan tidak maka akan melakukan pertambahan data di array.

```
public void pop() {
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    } else {
        elemen[top] = null;
    }
    top--;
}
```

Metode pop() yang digunakan untuk mengeluarkan sebuah data. dilakukan sebuah pengecekan jika metode yang dipanggil yaitu kosong maka akan mengambalikan kalimat kosong, sedang tidak maka akan melakukan pengosongan di array dan melakukan penguarangan indek

```
public E topAndPop() {
    if (isEmpty()) {
        return (E) "Tumpukan Sedang Kosong";
    } else {
        E free = elemen[top];
        pop();
        return free;
    }

}

public void push(E x) {
    top++;
    if (isEmpty()) {
        System.out.println("Array Sedang Kosong")
    } else {
        if (top == elemen.length - 2) {
            doubleArray();
        } else {
            elemen[top] = x;
        }
    }
}
```

Metode bertipa void push() digunakan untuk menambahkan sebuah data ke dalam stack (array stack). Pertama-tama metod ini akan melakukan pertambahan indek, selanjutnya jika pemanggilan data kosong benar akan diucetak kalimat kosong, jika tidak dilakukan

percabangan lagi engan jika top sama dengan panjang indek dikurangi 2, maka akan memanggil metode, jiak tidak lagi maka variabel x akan disimpan ke array elemen.

```
private void doubleArray() {
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    } else {
        elemen = Arrays.copyOf(elemen, elemen.length * 2);
    }
}
```

```
public void cetak() {
    for (int i = 0; i < elemen.length; i++) {
        if (elemen[i] != null) {
            System.out.println(elemen[i] + " ");
        }

    }
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    }
}
```

Sebuah method cetak() yang digunakan untuk mencetak semua data yang terdapat pada antrian. menjalankan perulangan dengan i yang bersifat dinamis, melakukan percabangan untuk mengecek jika data array tidak kosong akan mencetak data array tersebut, dilanjutkan jika kosong dengan memanggil metode isEmpty maka mencetak kalimat tumpukan sedang kosong.

2. Queue

- Penjelasan Program Queue
  Queue bisa disebut juga antrian pada struktur data. Pengertian queue adalah sekumpulan data yang mana penambahan elemen hanya bisa dilakukan pada suatu ujung yang disebut sisi belakang (rear), dan penghapusan (pengambilan elemen) dilakukan lewat ujung lain. Contoh queue paling sederhana dapat dilihat pada antrian. Prinsip kerja dari queue adalah prinsip “First In First Out” (FIFO) atau “masuk pertama keluar pertama”.

- Screenshot Listing Program

```
1   package Implementasi;
2
3   import static java.awt.PageAttributes.MediaType.A;
4   import java.util.Arrays;
5   import java.util.NoSuchElementException;
6   import java.util.*;
7   import java.util.logging.Level;
8   import java.util.logging.Logger;
9
10  public class MyArrayQueue<E> implements Queue<E> {
11      private E[] elemen;
12      private int front;
13      private int rear;
14      private static final int DEFAULT_CAPACITY = 5;
15      private int one, two;
16
17      public MyArrayQueue() {
18          one = DEFAULT_CAPACITY;
19          elemen = (E[]) new Object[DEFAULT_CAPACITY];
20          front = 0;
21          rear = -1;
22
23      }
24      public boolean isEmpty() {
25          return (elemen == null || two == -1);
26
27      }
28      @Override
29      public void makeEmpty() {
30          rear = -1;
31          for (int i = 0; i < elemen.length; i++) {
32              elemen[i] = null;
33          }
34      }
35      @Override
36      public E getFront() {
37          if (isEmpty()) {
38              //   throw new QueueException();
39              return (E) "Tumpukan Sedang Kosong";
40          } else {
41              return elemen[front % one];
42          }
43      }
44      @Override
45      public void dequeue() {
46          if (isEmpty()) {
47              System.out.println("Tumpukan Sedang Kosong");
48          } else {
49              elemen[front % one] = null;
50              front++;
51          }
52          two--;
53      }
54      @Override
```

```
    public E getFrontAndDequeue() {
        if (isEmpty()) {
            return (E) "Tumpukan Sedang Kosong";
        } else {
            E free = elemen[front];
            dequeue();
            return free;
        }
    }
    @Override
    public void enqueue(E x) {
        if (isEmpty()) {
            System.out.println("Tumpukan Sedang Kosong");
        } else {
            doubleSize();
            rear++;
            elemen[rear % one] = x;
            two++;
        }

    }
    public void cetak() {
        for (int i = 0; i < elemen.length; i++) {
            if (elemen[i] != null) {
                System.out.println(elemen[i] + " ");
            }

        }
        if (isEmpty()) {
            System.out.println("Tumpukan Sedang Kosong");
        }
    }
    public void doubleSize() {
        E[] newArray = (E[]) new Object[2 * one];
        for (int i = front; i <= rear; i++) {
            newArray[i - front] = elemen[i % one];
        }
        elemen = newArray;
        front = 0;
        rear = two - 1;
        one *= 2;
    }
    public static void main(String[] args) {
        MyArrayQueue<Integer> a = new MyArrayQueue<>();
        a.enqueue(10);
        a.enqueue(20);
        a.enqueue(12);
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();
        a.dequeue();
        a.cetak();

    }
}
```

- Penjelasan tiap Method

```
public class MyArrayQueue<E> implements Queue<E> {
    private E[] elemen;
    private int front;
    private int rear;
    private static final int DEFAULT_CAPACITY = 5;
    private int one, two;

    public MyArrayQueue() {
        one = DEFAULT_CAPACITY;
        elemen = (E[]) new Object[DEFAULT_CAPACITY];
        front = 0;
        rear = -1;

    }
```

Konstruktor kelas tanpa parameter,berfungsi untuk memberi nilai default apabila tidak diisi.

```
public boolean isEmpty() {
    return (elemen == null || two == -1);

}
@Override
public void makeEmpty() {
    rear = -1;
    for (int i = 0; i < elemen.length; i++) {
        elemen[i] = null;
    }
}
```

Dilanjutkan dengan metode isEmpty() untuk mengembalikan data apabila elemen ksoong maupun kurang dari -1. Dilanjutkan dengan makeEmpty dengan melakukan perulangan yang bersifat dinamis yang dimana disitu setiap
perulangan array diisikan dengan kosong.

```
public E getFront() {
    if (isEmpty()) {
        //   throw new QueueException();
        return (E) "Tumpukan Sedang Kosong";
    } else {
        return elemen[front % one];
    }
}

public void dequeue() {
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    } else {
        elemen[front % one] = null;
        front++;
    }
    two--;
}
```

Metode dequeue() ini digunakjan untuk mengeluarkan data dari sebuah array, Method in pertama-tama akan melakukan percabangan, jika data array kosong dengan melajkukan pemanggilan metode maka akan mencetak kalimat kosong, jika tidak maka akan memroses array elemen untuk diisikan kosong dan melakukan pertambahan indek front dan pengurang rear.

```
public E getFrontAndDequeue() {
    if (isEmpty()) {
        return (E) "Tumpukan Sedang Kosong";
    } else {
        E free = elemen[front];
        dequeue();
        return free;
    }
}

public void enqueue(E x) {
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    } else {
        doubleSize();
        rear++;
        elemen[rear % one] = x;
        two++;
    }

}
```

Sebuah metode (enqueue) yang digunakan untuk menambahkan data dalam sebuah array. Method in pertama-tama akan melakukan percabangan, jika data array kosong dengan melajkukan pemanggilan metode maka akan mencetak kalimat kosong, jika tidak maka akan memanggil meotde lalu melakukan pertamabahan indek rear, hingga pertambahan di array elemen dan penambah variaebel two

```
public void cetak() {
    for (int i = 0; i < elemen.length; i++) {
        if (elemen[i] != null) {
            System.out.println(elemen[i] + " ");
        }

    }
    if (isEmpty()) {
        System.out.println("Tumpukan Sedang Kosong");
    }
}
```

Sebuah method yang digunakan untuk mencetak semua data yang terdapat pada antrian. menjalankan perulangan dengan i yang bersifat dinamis, melakukan percabangan untuk mengecek jika data array tidak kosong akan mencetak data array tersebut, dilanjutkan jika kosong dengan memanggil metode isEmpty maka mencetak kalimat tumpukan sedang kosong.

```
public void doubleSize() {
    E[] newArray = (E[]) new Object[2 * one];
    for (int i = front; i <= rear; i++) {
        newArray[i - front] = elemen[i % one];
    }
    elemen = newArray;
    front = 0;
    rear = two - 1;
    one *= 2;
}
```