LAPORAN

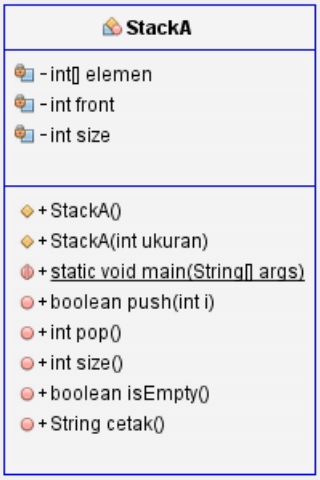
Struktur Data Linear
Praktikum 7 : STACK (TUMPUKAN)

NAMA : Johanes Yogtan WR

NIM : 215314105

Program Studi INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS SANATA DHARMA

A. DIAGRAM UML



B. METODE

1. Stack()

```
public StackA() {
    elemen = new int[10];
    size = 0;
    front = -1;
}
```

Penjelasan : Konstruktor kelas tanpa parameter,berfungsi untuk memberi nilai default apabila tidak diisi. Disitu dideklarasikan array elemen stack yang berisi 1o, size berisi 0, dan front berisi -1;

2. Stack(int ukuran)

```
public StackA(int ukuran) {
    elemen = new int[ukuran];
    front = -1;
    size = 0;
}
```

Penjelasan : Konstruktor kelas yang ada parameter, parameter ukuran disitu berfungsi untuk. Atribut array stack akan diinisiasi dengan ukuran yang telah diisikan di dalam parameter, sedangkan untuk atribut front dan size memiliki nilai masing – masing -1 dan 0.

3. push(int)

```
public boolean push(int i) {
    if (size < elemen.length) {
        front++;
        elemen[front] = i;
        size++;
        return true;
    }
    return false;
}
```

Penjelasan : Metode bertipa void digunakan untuk menambahkan sebuah data ke dalam stack (array stack). Method ini pertama – tama akan menjalankan percabangan untuk menambahkan data di array, dengan variabel size yang dinamis sesuai dengan inputan user itu juga yang dijalankan, dan selama percabangan berlangusng parameter I akan masuk ke array elemen dengan pertambahan sizenya, jika benar maka akan dijalankan jika tidak tidak dijalankan

4. pop()

```
public int pop() {
    if (size > 0) {
        size--;
        return elemen[front--];
    } else {

    }
    return 0;
}
```

Penjelasan : Methode yang bertipe int digunakan untuk mengeluarkan sebuah data dari dalam stack (array stack). Method ini pertama – tama akan menjalankan percabangan untuk mengeluarkan data di array, dengan variabel size yang dinamis sesuai dengan inputan user itu juga yang dijalankan, jiak benar maka size akan dikurangi begitu juga dengan data di array menggunakan variabel front--, jika tidak maka tidak dijalankan.

5. size()

```
public int size() {
    return size;

}
```

Penjelasan : Metode yang digunakan untuk mengembalikan nilai atribut size, variabel ini untuk melihat jumlah data arary secara visualisasi tidak seperti indek

6. isEmpty

```
public boolean isEmpty() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

Penjelasan : Metode yang digunakan untuk mengecek apakah stack tersebut kosong atau masih memiliki isi. Jika kosong maka dijalankan atau true, jika tidak akan bernilai false atau tidak dijalankan

7. cetak

```
public String cetak() {
    System.out.println();
    for (int i = front; i >= 0; i--) {

        System.out.print("Front " + i + " =>");
        System.out.println(" " + elemen[i] + " ");

    }
    String y = "Size";

    //return blank;
    return y;
```

Penjelasan : Sebuah method yang digunakan untuk mencetak semua data yang terdapat pada stack. Disini saya membuat ouputnya seperti stack beneren jadi front indek yang besar akan dijalankan terlebuih dahulu jadi ibarat indek yang terakhir masuk akan dicetak terlebih dahulu agar visualnya nanti yang bagian bawah akan mencetak yang pertama dulu masuk, saya juga disini membaut kalimat front … untuk memperjelas bagian indek dalam output, dan juga dengan nilai balikan size untuk memperjelas bagian sizenya.

C. IMPLEMENTASI

Screenshot Kelas Main :

```
package Modul7_Stack;
public class StackA {
    public static void main(String[] args) {
        StackA s = new StackA(3); //1
        s.push(23); //2
        s.push(45); //3
        s.push(56); //4

        System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        if ((s.pop() == -1)) {//6
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        s.pop(); //7
        if ((s.isEmpty())) {
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()+" ");//5

        }

        s.push(56);  //8
        if ((s.pop() == -1)) {
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        if ((s.pop() == -1)) //9
        {
            System.out.println("data sudah habis");
        } else {

            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        if ((s.isEmpty()))
        {
            System.out.println("data sudah habis");
        }
    }
    private int[] elemen;
    private int front;
    private int size;

    public StackA() {
        elemen = new int[10];
        size = 0;
        front = -1;
    }
```

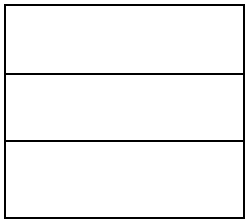
```
    public StackA(int ukuran) {
        elemen = new int[ukuran];
        front = -1;
        size = 0;
    }
    public boolean push(int i) {
        if (size < elemen.length) {
            front++;
            elemen[front] = i;
            size++;
            return true;
        }
        return false;
    }
    public int pop() {
        if (size > 0) {
            size--;
            return elemen[front--];
        } else {

        }
        return 0;
    }
    public int size() {
        return size;

    }
    public boolean isEmpty() {
        if (size == 0) {
            return true;
        } else {
            return false;
        }
    }
     public String cetak() {
        System.out.println();
        for (int i = front; i >= 0; i--) {

            System.out.print("Front " + i + " =>");
            System.out.println(" " + elemen[i] + " ");

        }
        String y = "Size";

        //return blank;
        return y;
    }
}
```

Screenshot Output :

```
Front 2 => 56
Front 1 => 45
Front 0 => 23
Ukuran : 3 Size

Front 1 => 45
Front 0 => 23
Ukuran : 2 Size

Front 0 => 23
Ukuran : 1 Size

Front 0 => 23
Ukuran : 1 Size

Ukuran : 0 Size
data sudah habis
```

Ilustrasi Stack :

- **StackA elemen = new StackA(3);**
  Front = -1
  Element.length = 3
  Size = 0



- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**

  Front = 2
  Element.length = 3
  Size = 3

56
45
23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **(s.pop() == -1)**

Front = 1
Element.length = 3
Size = 2

45
23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**

Front = 0
Element.length = 3
Size = 1

23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**

- **s.isEmpty() False**

  Front = 0
  Element.length = 3
  Size = 1

23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**
- **s.isEmpty() False**
- **s.push(56)**

  Front = 1
  Element.length = 3
  Size = 2

56
23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**
- **s.isEmpty() False**
- **s.push(56)**
- **s.pop();**

  Front = 0
  Element.length = 3
  Size = 1

23

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**
- **s.isEmpty() False**
- **s.push(56)**
- **s.pop();**
- **s.pop();**

Front = -1
Element.length = 3
Size = 0

- **StackA elemen = new StackA(3);**
- **s.push(23);**
- **s.push(45);**
- **s.push(56);**
- **s.pop();**
- **s.pop();**
- **s.isEmpty() False**
- **s.push(56)**
- **s.pop();**
- **s.pop();**
- **s.isEmpty() True (Data sudah habis)**

Front = -1
Element.length = 3
Size = 0

Penjelasan Ilustrasi :

```
package Modul7_Stack;
public class StackA {
    public static void main(String[] args) {
        StackA s = new StackA(3); //1
        s.push(23); //2
        s.push(45); //3
        s.push(56); //4

        System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        if ((s.pop() == -1)) {//6
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        s.pop(); //7
        if ((s.isEmpty())) {
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()+" ");//5

        }

        s.push(56);  //8
        if ((s.pop() == -1)) {
            System.out.println("data sudah habis");
        } else {
            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        if ((s.pop() == -1)) //9
        {
            System.out.println("data sudah habis");
        } else {

            System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5
        }

        if ((s.isEmpty()))
        {
            System.out.println("data sudah habis");
        }
    }
```

Nomor Komentar	Penjelasan
//1	Membuat object dari class StackD_Object dengan nama s dan objeknya dibuat dengan memanggil kosntruktor yang ada parameternya
//2	Memangil metode push untuk memasukkan datanya

//3	Memangil metode push untuk memasukkan datanya
//4	Memangil metode push untuk memasukkan datanya
//5	Mencetak ukuran stack dan isi stack
//6	Percabangan yangsyaratnya adalah hasil return method pop sama dengan -1. Jika true maka akan mencetak kalimat 'Data sudah habis' jika false maka akan mencetak ukuran dan isi stack yang sudah di pop
//7	Memangil metode pop untuk mengeluarkan data
//8	Memangil metode push untuk memasukkan datanya
//9	Percabangan yangsyaratnya adalah hasil return method pop sama dengan -1. Jika true maka akan mencetak kalimat 'Data sudah habis' jika false maka akan mencetak ukuran dan isi stack yang sudah di pop

```
Front 2 => 56
Front 1 => 45
Front 0 => 23
Ukuran : 3 Size

Front 1 => 45
Front 0 => 23
Ukuran : 2 Size

Front 0 => 23
Ukuran : 1 Size

Front 0 => 23
Ukuran : 1 Size

Ukuran : 0 Size
data sudah habis
```

Outputnya	Codingannya

`s.push(23); //2`
`s.push(45); //3`
`s.push(56); //4`
 	
`System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5`
	`Front 2 => 56`
	`Front 1 => 45`
	`Front 0 => 23`
	`Ukuran : 3 Size`

`if ((s.pop() == -1)) {//6`
`  System.out.println("data sudah habis");`
`} else {`
`  System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5`
`}`
	`Front 1 => 45`
	`Front 0 => 23`
	`Ukuran : 2 Size`

`s.pop(); //7`
`if ((s.isEmpty())) {`
`System.out.println("data sudah habis");`
`} else {`
`System.out.println("Ukuran : " + s.size() + " " + s.cetak()+" ");//5`
`}`
	`Front 0 => 23`
	`Ukuran : 1 Size`

`s.push(56); //8`
`if ((s.pop() == -1)) {`
`System.out.println("data sudah habis");`
`} else {`
`System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5`
`}`
	`Front 0 => 23`
	`Ukuran : 1 Size`

`if ((s.pop() == -1)) //9`
`{`
`System.out.println("data sudah habis");`
`} else {`
`System.out.println("Ukuran : " + s.size() + " " + s.cetak()); //5`
`}`
	`Ukuran : 0 Size`

`if ((s.isEmpty()))`
`{`
`System.out.println("data sudah habis");`
`}`
	`data sudah habis`

Ketika program dari penjelasan nomo 2 dijalankan akan membetuk output seperti ini. Saya membuat output seperti ini suapaya dapat memperjelas front dan sizenya, seperti yang dilihat pada spasi pertama sesuai dengan penjelasan nomr 2, output spasi pertama berasal dari 3 push yang sudah dijelaskan dan dicetak outputnya beserta sizenya, selanjutnya di spasi dua, perintah pop dijalankan, namun masuk ke dalam else sama halnya seperti spasi 3 sehingga dapat terbentuk seperti itu, namun bedanya dengan spasi 4 yang menjadi pertanyaan mengapa output 3 dan 4 spasi sama, karena di spasi 4 terlebih dahulu melakukan pertambahan data, namun di keluarkan lagi, oleh karena itu jadinya sama. Terakhir kelima percabangan isempty akhirnya bernilai true makannya dicetak menajkdi data sudah habis.