

Kelas : 03

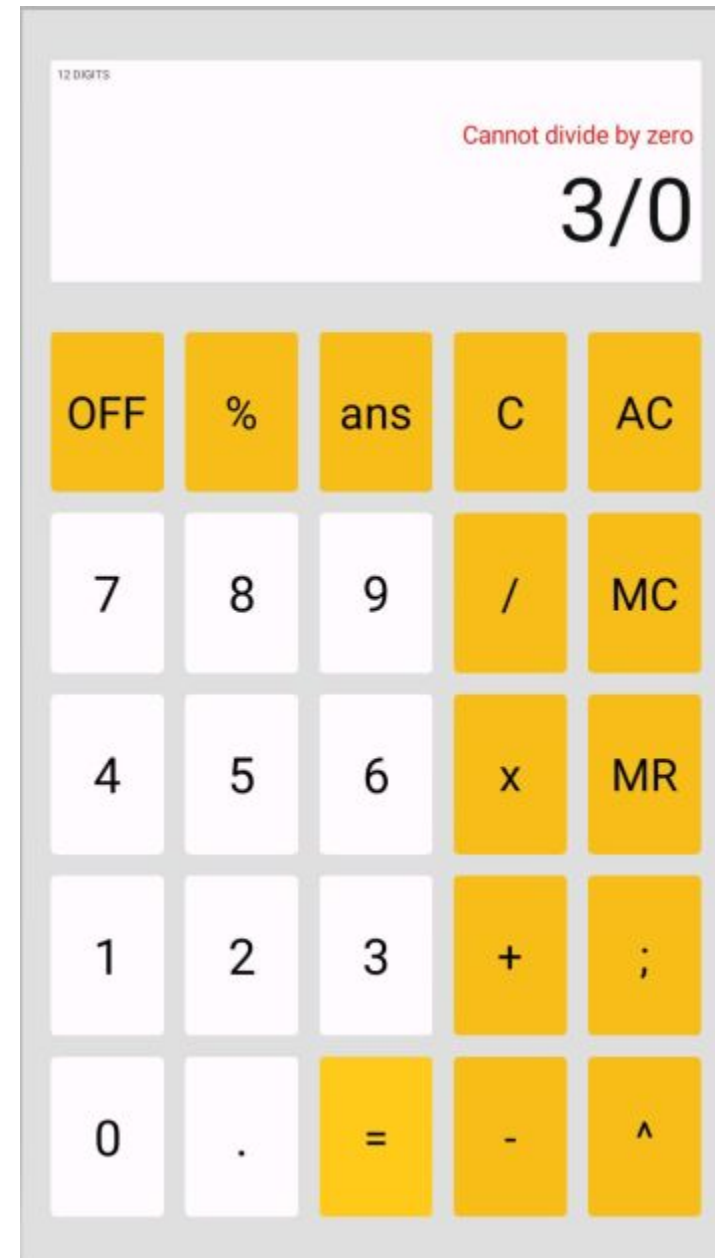
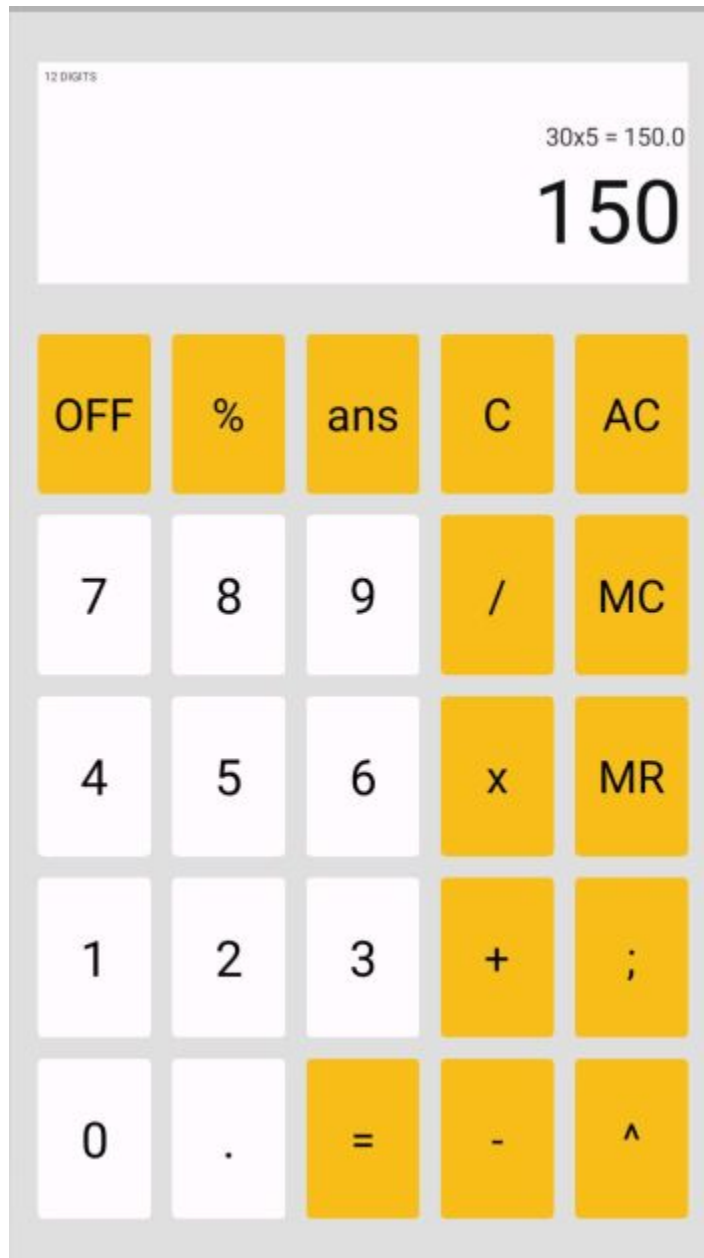
Kelompok : 07

1. 13518024 / Jovan Karuna Cahyadi
  2. 13518066 / Byan Sakura Kireyna Aji
  3. 13518075 / Daniel Riyanto
  4. 13518084 / Jonathan Yudi Gunawan
- Asisten Pembimbing : Fahmi

## 1. Deskripsi Umum Aplikasi

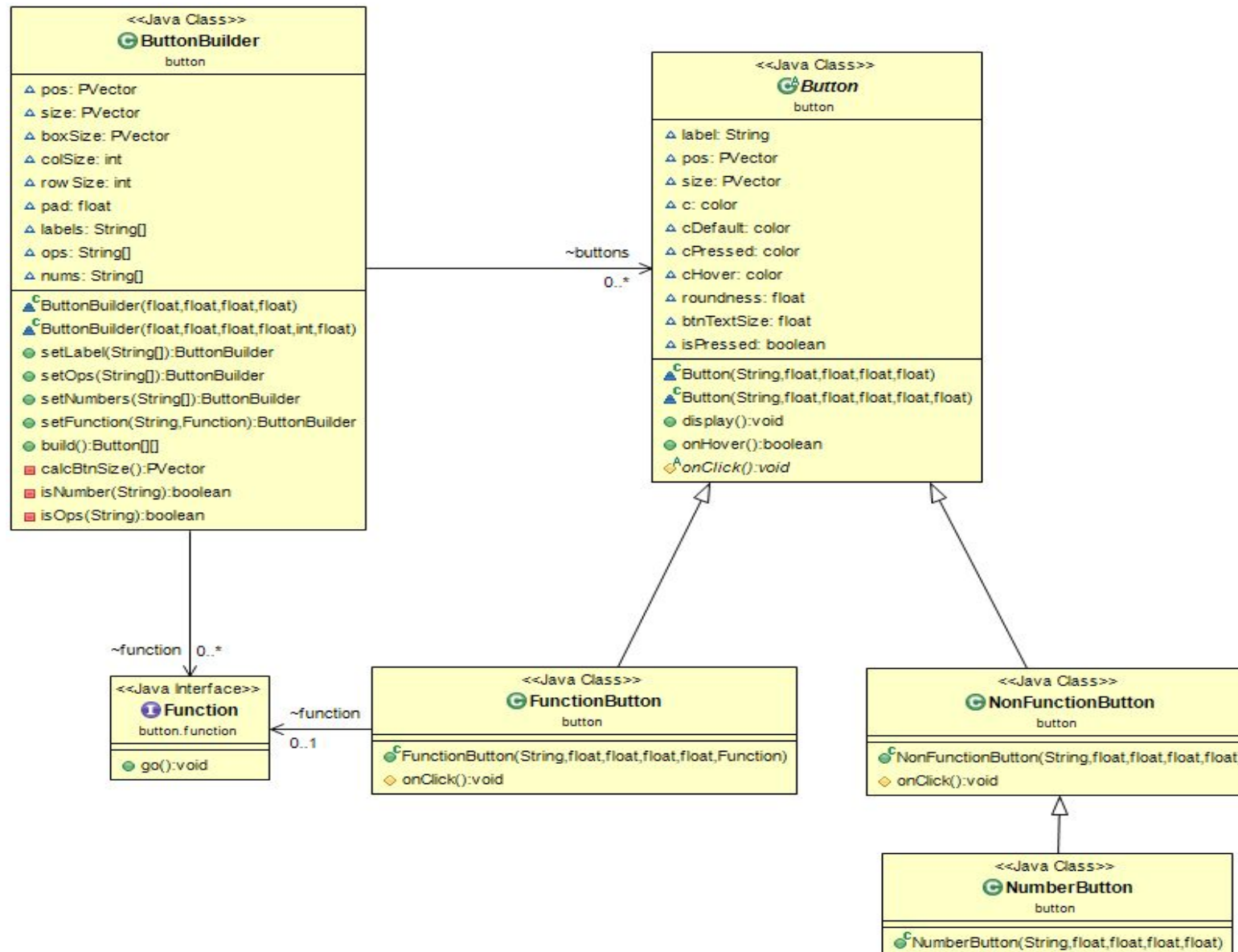
Pada tugas besar kali ini kami merancang suatu kalkulator berbasis GUI yang dilengkapi dengan tombol 0-9 dan titik, beberapa operator matematika, perhitungan, ans, MC dan MR, serta clear. Aplikasi kalkulator ini kami rancang dalam bahasa pemrograman yang berbasis pada paradigma objek. Pada pengerjaan program juga diterapkan beberapa design pattern seperti Builder pattern dan Singleton Pattern. Kami juga mengerjakan bonus berupa operasi pemangkatan dan modulo, serta dapat menerima ekspresi lebih dari 2 sekaligus, misal  $5 + 2 * 3$  yang juga memperhatikan prioritas pengerjaan.

Seperti kalkulator pada umumnya, aplikasi kami mampu melakukan perhitungan sesuai dengan input yang dimasukkan user melalui tombol kalkulator dan akan menampilkan hasil perhitungan pada layar textbox yang disediakan. Kalkulator kami menerima berbagai ekspresi namun tidak semua masukan akan dapat dijalankan karena beberapa string yang salah akan mengembalikan pesan error pada textbox. Berikut tampilan aplikasi ketika dijalankan (kiri: normal, kanan: error):



## 2. Diagram Kelas

### 2.1 Diagram Kelas Package Button

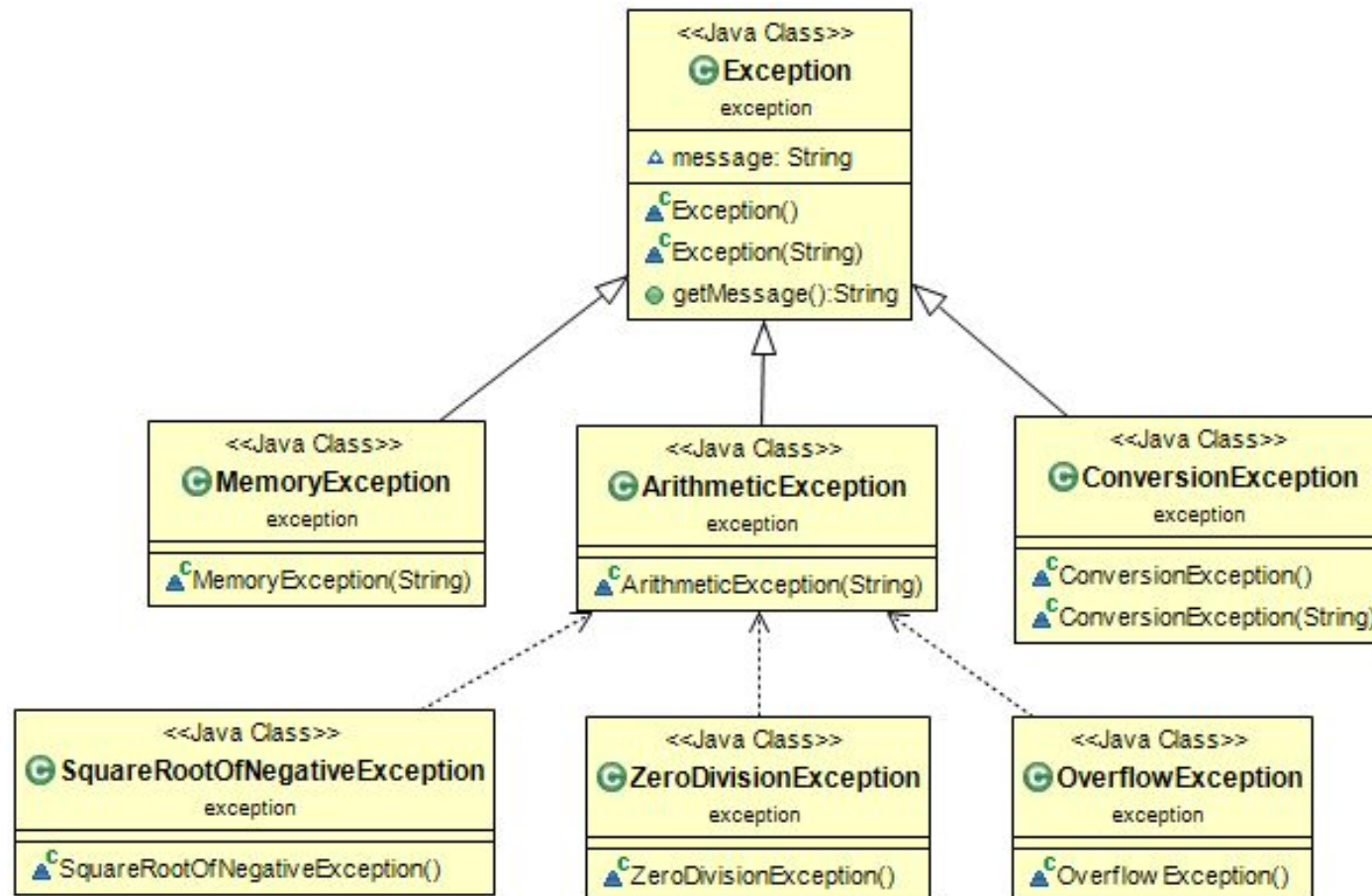


Package button berisi 5 class. Class Button merupakan class utama yang menjadi parent bagi 3 class. Button menyimpan posisi, ukuran, warna, teks pada button, dan state apakah dia sedang ditekan. Button memiliki method display, onClick, dan onHover.

Button dibagi menjadi 2 tipe yaitu functional button dan non-functional button. Functional button akan melakukan suatu perhitungan ketika ditekan sehingga memiliki 1 atribut tambahan yaitu function. Function sendiri merupakan abstract class dari berbagai jenis fungsi yang dapat dilakukan oleh kalkulator kami. Non functional button tidak melakukan perhitungan sama sekali ketika ditekan, namun hanya menampilkan keluar isi teks pada button ke layar kalkulator. Kami memiliki 1 class khusus untuk number button karena memiliki warna berbeda, ukuran berbeda, dan sedikit behavior yang berbeda dibanding dengan operator seperti + atau -.

Class terakhir adalah ButtonBuilder yang menerapkan design pattern builder pattern dengan atributnya yaitu posisi (posisi button paling kiri atas), ukuran (ukuran total container), jumlah button dalam 1 row, padding antar button, dan array of label, operator, dan hashmap string, function (akan dijelaskan di bawah). Button memiliki method seperti setOpr, setLabel, dan setFunction yang akan digunakan nanti ketika dipanggil method build. Method build akan mereturn array 2D of Button yang dapat ditampilkan nantinya oleh class View.

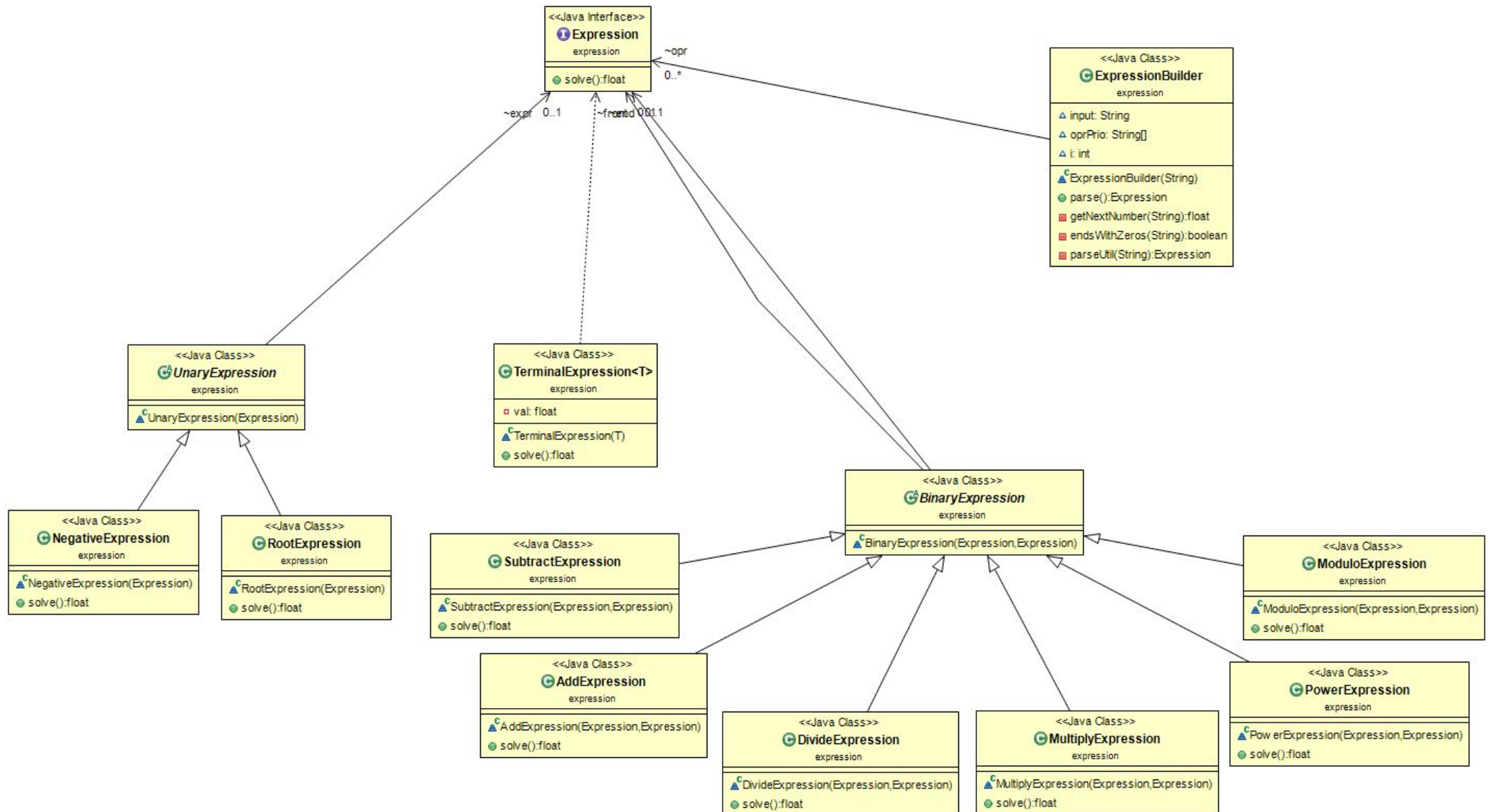
## 2.2 Diagram Kelas Package Exception



Package exception terdiri 7 kelas dengan 1 kelas utama yaitu Exception, 6 kelas lainnya merupakan turunan dari class exception. Exception memiliki atribut message bertipe string dan merupakan turunan dari class Throwable sehingga dapat di throw. Exception dibagi menjadi 3 yaitu memory (memori penuh atau memori kosong), arithmetic (galat pada perhitungan), dan conversion (masukan tidak dapat diparsing menjadi expression yang benar).

Arithmetic Exception dibagi lagi menjadi 3 yaitu akar negatif, dibagi dengan nol, dan overflow (layar textbox tidak muat).

## 2.3 Diagram Kelas Package Expression



Package Expression memiliki class utama yaitu Expression yang memiliki abstract method solve. Class Expression tersebut terbagi menjadi 3 subclass yaitu binary, unary, dan terminal.

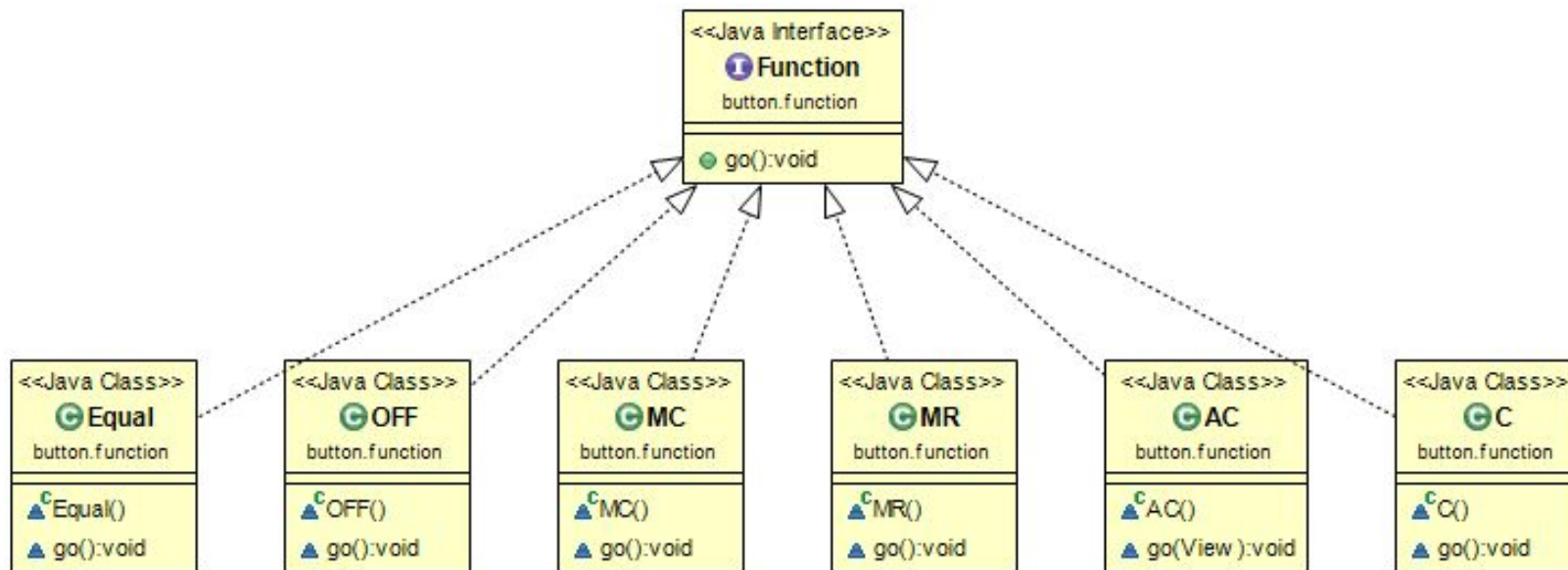
Binary Expression terbagi menjadi 6 yaitu Subtract, Add, Divide, Multiply, Modulo, dan Power. Binary expression memiliki 2 atribut Expression dan memiliki method solve.

Unary Expression terbagi menjadi 2 yaitu Negative Expression dan Root Expression dan hanya memiliki 1 atribut Expression.

Ada satu class lagi yaitu ExpressionBuilder, di dalamnya terdapat parser yang mengubah dari string inputan user menjadi sebuah expression yang valid atau meng-throw exception bila gagal. Sebelumnya pada expression ingin digunakan Factory Pattern tapi karena masing-masing operator memiliki behavior throw berbeda (dan diharuskan memiliki throw message berbeda) sehingga kami putuskan untuk mengimplementasikan satu-satu dalam method parse. Di dalam class ini juga yang menentukan precedence dari operator, misal operator perkalian dikerjakan terlebih dahulu daripada operator penjumlahan, atau operator negatif dikerjakan paling pertama dari operator manapun.



## 2.4 Diagram Kelas Package Function



Package function terdiri dari 1 interface function dan 6 fungsi yang mengimplementasikan interface tersebut. Fungsi-fungsi tersebut berupa equal (=), OFF, MC, MR, AC, dan C.

Equal digunakan untuk mengevaluasi string inputan user dan menampilkan kembali hasilnya ke layar.

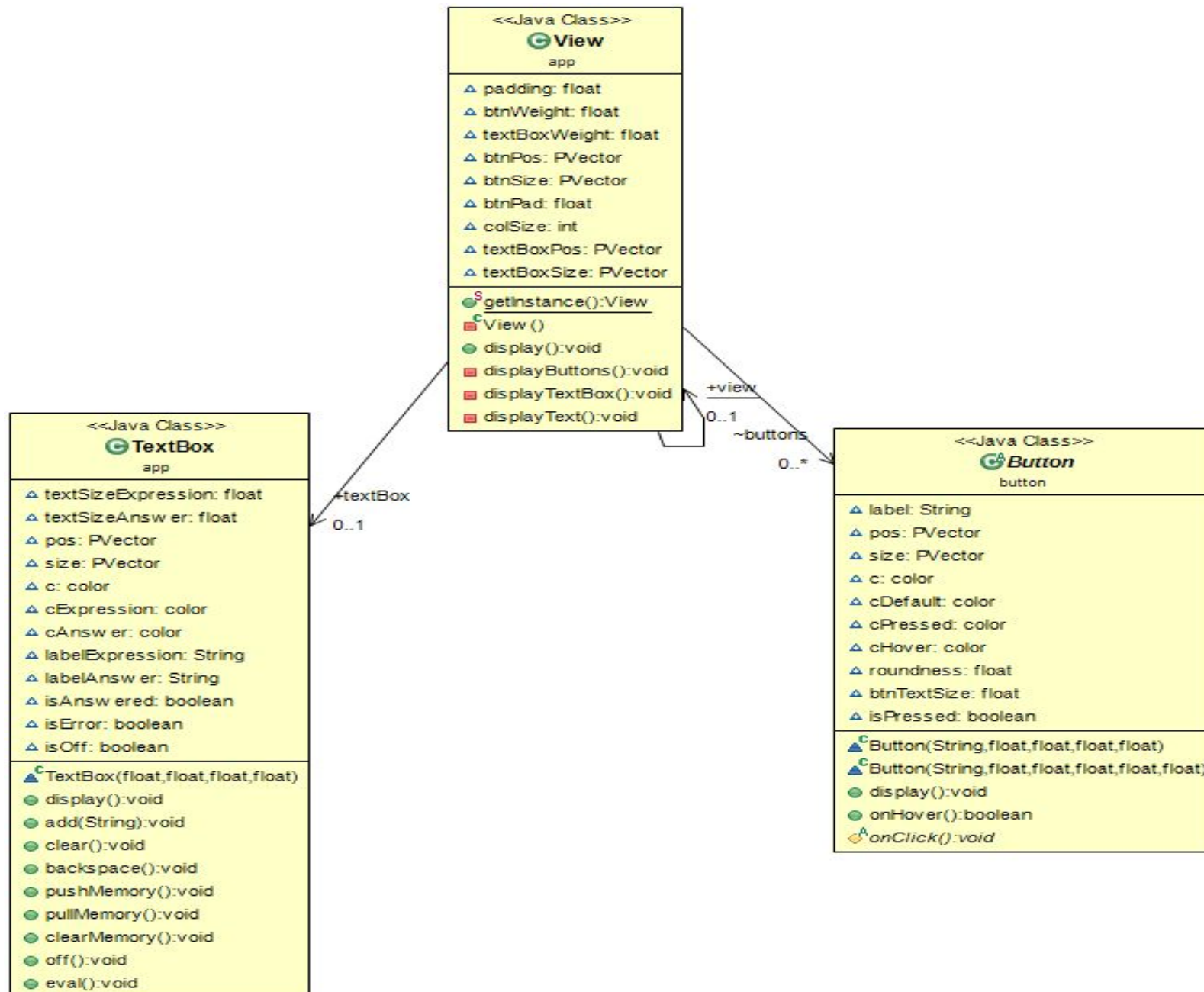
OFF digunakan untuk “mematikan” kalkulator sehingga tombol-tombolnya tidak bisa dipencet.

MC dan MR digunakan untuk menyimpan dan mengeluarkan isi memori kalkulator.

AC digunakan untuk membersihkan layar dan membersihkan memori sekaligus.

C digunakan untuk menghapus satu karakter dari layar.

## 2.5 Diagram Kelas Package App



Package app terdiri dari View dan textBox. Keduanya berfungsi untuk menjadi jembatan antara user dengan perhitungan dalam program. Kedua class memanfaatkan library processing untuk menggambar persegi maupun menampilkan teks, serta menerima inputan mouse user.

Class View memiliki sebuah textBox dan sejumlah Button (komposisi). Button dihasilkan dengan memanfaatkan class ButtonBuilder. Class View berfungsi untuk mengatur posisi kedua objek ini dan menjadi jembatan komunikasi antara keduanya. Misal, button dapat mengakses textBox melalui View.

Pada textBox dan View dapat digunakan Singleton Pattern dengan menjadikan konstruktornya private dan memiliki 1 atribut static view, namun sayangnya pada processing tidak memungkinkan untuk memiliki atribut statik karena semua class di dalam processing merupakan inner class dari sebuah class yang lebih besar.

### **3. Bahasa Pemrograman**

Untuk aplikasi ini kami menggunakan bahasa pemrograman Java yang mendukung pemrograman berparadigma objek dan GUI. Kami juga menggunakan library Processing seperti `processing.core.PApplet` untuk mempermudah pemrograman GUI. Pada Java juga terdapat kelas yang merupakan objek dimana informasi atau tugas terkandung. Keberadaan kelas ini dapat mempermudah penentuan struktur desain pemrograman yang berorientasi kepada objek.

### 3.1. Inheritance

Contoh penggunaan di bahasa:

```
class MountainBike extends Bicycle {  
    public int seatHeight;  
    public MountainBike(int gear,int speed, int startHeight) {  
        super(gear, speed);  
        seatHeight = startHeight;  
    }  
}
```

Contoh penggunaan di aplikasi:

```
class AddExpression extends BinaryExpression{ // class BinaryExpression ada di 3.2  
    AddExpression(Expression front, Expression end){  
        super(front, end);  
    }  
  
    public float solve() throws Exception {  
        try {  
            ...  
        } catch (Exception e) {  
            throw e;  
        }  
    }  
}
```

Kode di atas adalah salah satu contoh penerapan inheritance dari banyak class lainnya. Dengan menerapkan inheritance dapat diterapkan juga konsep lain seperti polymorphism, dll.

### 3.2. Interface / Abstract Base Class

Contoh penggunaan di bahasa:

```
interface Animal {  
    public void animalSound();  
    public void run();  
}
```

Contoh penggunaan di aplikasi:

```
interface Expression {  
    abstract float solve() throws Exception;  
}  
  
abstract class BinaryExpression implements Expression{  
    Expression front, end;  
    BinaryExpression(Expression front, Expression end){  
        this.front = front;  
        this.end = end;  
    }  
}
```

Expression menerapkan interface dan juga abstract base class. Pada Binary Expression dan Unary Expression digunakan abstract base class karena diperlukan atribut dan konstruktor yang sama, sedangkan pada Expression masih terlalu general untuk mengimplementasikan kedua hal tersebut.

### 3.3. Composition / Aggregation

Contoh penggunaan di bahasa:

```
class Association
{
    public static void main (String[] args)
    {
        Bank bank = new Bank("Axis");
        Employee emp = new Employee("Neha");

        System.out.println(emp.getEmployeeName() +
            " is employee of " + bank.getBankName());
    }
}
```

Contoh penggunaan di aplikasi:

```
abstract class BinaryExpression implements Expression{
    Expression front, end;
    BinaryExpression(Expression front, Expression end){
        this.front = front;
        this.end = end;
    }
}
```

BinaryExpression merupakan komposisi dari 2 Expression(front dan end).



### 3.4. Method / Operator Overloading

Contoh penggunaan di bahasa:

```
public int sum(int x, int y){  
    return (x + y);  
}
```

Contoh penggunaan di aplikasi:

```
abstract class Button {  
    ...  
    // constructor  
    Button(String labelB, float xpos, float ypos, float widthB, float heightB) {  
        this(labelB, xpos, ypos, widthB, heightB, 10);  
    }  
  
    // constructor (fully customize)  
    Button(String labelB, float xpos, float ypos, float widthB, float heightB, float roundnessB) {  
        this.label = labelB;  
        this.pos = new PVector(xpos, ypos);  
        this.size = new PVector(widthB, heightB);  
        this.roundness = roundnessB;  
        this.isPressed = false;  
        this.cDefault = BTN_DEFAULT;  
        this.cPressed = BTN_PRESSED;  
        this.cHover = BTN_HOVER;  
    }  
}
```

```
    this.btnTextSize = min(this.size.x, this.size.y)/2.5;
  }
  ...
}
```

Constructor class Button dapat menerima 2 jenis input berbeda, dengan jumlah parameter berbeda.

### 3.5. Polymorphism

Contoh penggunaan di bahasa:

```
class Animal {  
    public void animalSound() {  
        System.out.println("The animal makes a sound");  
    }  
}  
  
class Pig extends Animal {  
    public void animalSound() {  
        System.out.println("The pig says: wee wee");  
    }  
}
```

Contoh penggunaan di aplikasi:

```
class Exception extends Throwable{  
    ...  
}  
  
class ArithmeticException extends Exception {  
    ...  
}  
  
class ConversionException extends Exception {  
    ...  
}
```

```
}  
  
class View {  
    ...  
    public void eval() {  
        ...  
        try {  
  
        } catch (Exception e) {  
            this.labelExpression = e.getMessage();  
            ...  
        }  
    }  
}  
}
```

Class Exception ini disebut memiliki sifat *polymorphism* karena memiliki banyak bentuk untuk error-error seperti *error* pembagian, akar, konversi dan lain-lain. Pada saat di-*catch*, program belum mengetahui Exception yang akan dicatch secara pasti, namun karena semua Exception merupakan turunan dari Exception maka semua exception dapat di-treat sebagai Exception.

### 3.6. Generic

Contoh penggunaan di bahasa:

```
class Test<T> {  
    // An object of type T is declared  
    T obj;  
    Test(T obj) { this.obj = obj; } // constructor  
    public T getObject() { return this.obj; }  
}
```

Contoh penggunaan di aplikasi:

```
class TerminalExpression<T extends Number> implements Expression {  
    private float val;  
  
    TerminalExpression(T val){  
        this.val = val.floatValue();  
    }  
    public float solve(){  
        return this.val;  
    }  
}
```

Constructor dari class TerminalExpression dapat menerima berbagai jenis class T selama masih anggota dari Number (Number merupakan bawaan dari java yaitu class yang merupakan sebuah angka).

### 3.7. Exception

Contoh penggunaan di bahasa:

```
package exception;
class ConversionException extends Exception {
    ConversionException(){
        this("");
    }
    ConversionException(String msg){
        this.message = msg;
    }
}
```

Contoh penggunaan di aplikasi:

```
class Exception extends Throwable{
    String message;
    Exception(){
        this("");
    }
    Exception(String msg){
        this.message = msg;
    }
    public String getMessage(){
        return this.message;
    }
}
```

```
class ArithmeticException extends Exception {
    ArithmeticException(String msg){
        super(msg);
    }
}
class OverflowException extends ArithmeticException {
    OverflowException(){
        super("Number is too big");
    }
}
class PowerExpression extends BinaryExpression {
    ...
    public float solve() throws Exception{
        try{
            float ans = pow(this.front.solve(),this.end.solve());
            if(ans > pow(10,12)-1 || ans < -pow(10,11)+1){
                throw new OverflowException();
            } else {
                return ans;
            }
        } catch (Exception e){
            throw e;
        }
    }
}
```

## 3.8. Library

### 3.8.1 HashMap

Digunakan pada ButtonBuilder untuk me-mapping operator dengan fungsinya, sehingga untuk menambahkan FunctionButton baru cukup dengan method setLabel, setOps, dan setFunction. **ButtonBuilder** menerapkan paradigma objek **Open Closed Principle** karena untuk menambah button baru tidak perlu mengubah kode di dalam ButtonBuilder. Berikut cuplikan kode pada class ButtonBuilder yang menunjukkan penggunaan HashMap:

```
import java.util.HashMap;

class ButtonBuilder{
    ...
    HashMap<String, Function> function;

    ButtonBuilder(float xpos, float ypos, float widthB, float heightB, int _colSize, float pad){
        ...
        this.function = new HashMap<String, Function>();
    }

    ...

    public ButtonBuilder setFunction(String fname, Function func){
        this.function.put(fname, func);
        return this;
    }
}
```



```
public Button[][] build(){
    this.buttons = new Button[this.rowSize][this.colSize];
    int i = 0;
    int j = 0;
    for(String s: this.labels){
        if(this.isNumber(s)){
            ...
        } else if(this.isOps(s)){
            ...
        } else {
            this.buttons[i][j] = new FunctionButton(s, this.pos.x + this.size.x*j + this.pad*j, this.pos.y + this.size.y*i +
this.pad*i, this.size.x, this.size.y, this.function.get(s));
        }
        ...
    }
    return this.buttons;
}
}
```

### 3.8.2 Queue dan LinkedList

Queue digunakan pada TextBox untuk menyimpan/push hasil saat ini dengan penggunaan MC dengan menggunakan container berupa LinkedList. Hasil tersebut disimpan dalam aturan queue/antrian yang akan dipanggil/pull dengan menggunakan MR. Hasil penyimpanan akan dihapus semuanya jika menggunakan AC(clear) pada kalkulator. Berikut cuplikan kode pada class TextBox yang menunjukkan penggunaan Queue dan LinkedList:

```
import java.util.LinkedList;
import java.util.Queue;
...
Queue<Float> memory;

void setup(){
    memory = new LinkedList<Float>();
}

...
class TextBox{
    ...
    public void pushMemory(){
        if(!this.isOff){
            try {
                if (this.isAnswered) {
                    memory.add(Float.parseFloat(this.labelAnswer));
                    this.labelAnswer = "0";
                    this.labelExpression = "";
                } else /* !this.isAnswered */ {
```

```
        throw new MemoryException("Not a Number");
    }
} catch (Exception e) {
    this.labelExpression = e.getMessage();
    this.isError = true;
}
}
}

public void pullMemory(){
    if(!this.isOff){
        try{
            if (memory.size() != 0) {
                float f = memory.remove();
                if(f == int(f)){
                    this.add(str(int(f)));
                } else {
                    this.add(str(f));
                }
            } else {
                throw new MemoryException("Memory is empty");
            }
        } catch (MemoryException e){
            this.labelExpression = e.getMessage();
            this.isError = true;
        }
    }
}
```

```
    }  
  }  
}  
public void clearMemory() {  
    while (memory.size() > 0) {  
        memory.remove();  
    }  
}  
...  
}
```

## 4. Test

Kami membagi testing menjadi beberapa kelompok berdasar jenis testing yang akan digunakan (misal testing untuk parsing string, testing button non fungsional, testing button fungsional, testing queue (MC dan MR), dll). Dari tes yang dilakukan diharapkan dapat meminimalisir bug yang ada di program dan dapat dengan cepat mengidentifikasi letak kesalahan pada program.

Langkah yang dilakukan adalah dengan mendaftar skenario pengecekan mulai dari yang simpel hingga yang rumit. Skenario meliputi seluruh user stories mulai dari awal (kalkulator baru dinyalakan) hingga kemungkinan terburuk seperti kombinasi operator yang sangat rumit. Skenario juga mencakup seluruh tombol yang tersedia di kalkulator kami.

Berikut hasil dari beberapa testing yang kami lakukan:

### 4.1 Test Urutan Perkalian dan Penjumlahan

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "5", "+", "2", "x", "3".  2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan "5+2x3"  2.) Bagian atas TextBox menampilkan "5+2x3 = 11.0" dan bagian bawah TextBox menampilkan "11"	Sama dengan yang diharapkan

**4.2 Test Urutan Operator Negatif dengan Operator Binary (contoh :Perkalian)**

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "5", "x", "-", "3".  2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan "5x-3"  2.) Bagian atas TextBox menampilkan "5x-3 = -15.0" dan bagian bawah TextBox menampilkan "-15"	Sama dengan yang diharapkan

**4.3 Test Urutan Operator Negatif dengan Akar**

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "-", "√", "4".  2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan "-√4"  2.) Bagian atas TextBox menampilkan "-√4 = -2.0" dan bagian bawah TextBox menampilkan "-2"	Sama dengan yang diharapkan

#### 4.4 Test Exception Akar Negatif

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu " $\sqrt{\phantom{x}}$ ", "-", "4".  2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan " $\sqrt{-4}$ "  2.) Bagian atas TextBox menampilkan "Cannot have negative root" dan bagian bawah TextBox menampilkan " $\sqrt{-4}$ "	Sama dengan yang diharapkan

#### 4.5 Test Overflow

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "10", "^", "13".  2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan " $10^{13}$ "  2.) Bagian atas TextBox menampilkan "Number is too big" dan bagian bawah TextBox menampilkan " $10^{13}$ "	Sama dengan yang diharapkan

#### 4.6 Test Tombol Ans

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "1", "+", "1". 2.) Menekan tombol "=" 3.) Menekan tombol "+", "ans" 4.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan "1+1" 2.) Bagian atas TextBox menampilkan "1+1=2.0" dan bagian bawah TextBox menampilkan "2" 3.) Sebelum menekan tombol "=", TextBox menampilkan "2+ans" 4.) Bagian atas TextBox menampilkan "2+ans=4.0" dan bagian bawah TextBox menampilkan "4"	Sama dengan yang diharapkan

#### 4.7 Test Tombol Ans "Salah" dan C

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "1", "+", "1". 2.) Menekan tombol "=" 3.) Menekan tombol "C" 4.) Menekan tombol "ans", "ans"	1.) Sebelum menekan tombol "=", TextBox menampilkan "1+1" 2.) Bagian atas TextBox menampilkan "1+1=2.0" dan bagian bawah TextBox menampilkan "2" 3.) Bagian atas TextBox menampilkan "1+1=2.0" dan bagian bawah TextBox menampilkan "0"	Sama dengan yang diharapkan



5.) Menekan tombol “=”	<p>4.) Bagian atas TextBox menampilkan “1+1=2.0” dan bagian bawah TextBox menampilkan “ansans”</p> <p>5.) Bagian atas TextBox menampilkan “<b>Expression incomplete</b>” dan bagian bawah TextBox menampilkan “ansans”</p>	
------------------------	--	--

#### 4.8 Test Mengambil dari Memory Kosong dan test tombol AC

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
<p>1.) Pada saat kondisi tutup, menekan tombol “AC”</p> <p>2.) Menekan tombol-tombol secara berurutan, yaitu “1”, “+”, “1”.</p> <p>3.) Menekan tombol “AC”</p> <p>4.) Menekan tombol “MR”</p>	<p>1.) Kalkulator menampilkan layar terang dan TextBox menampilkan “0”</p> <p>2.) TextBox menampilkan “1+1”</p> <p>3.) TextBox menampilkan “0”</p> <p>4.) Bagian atas TextBox menampilkan “<b>Memory is empty</b>” dan bagian bawah TextBox menampilkan “0”</p>	Sama dengan yang diharapkan

**4.9 Test Menyimpan Ekspresi tidak valid**

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol “1”, “+” 2.) Menekan tombol “MC”	1.) Sebelum menekan tombol “MC”, TextBox menampilkan “1+” 2.) Bagian atas TextBox menampilkan “Not a Number” dan bagian bawah TextBox menampilkan “1+”	Sama dengan yang diharapkan

**4.10 Test tombol OFF**

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol “OFF” 2.) Menekan tombol “1 + 1” 3.) Menekan tombol “=”	1.) TextBox menampilkan layar gelap 2.) TextBox tidak menampilkan apa-apa 3.) TextBox tidak menampilkan apa-apa	Sama dengan yang diharapkan

**4.11 Test tombol MC dan MR**

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu “1”, “+”, “1”.	1.) Sebelum menekan tombol “=”, TextBox menampilkan “1+1”	Sama dengan yang diharapkan

2.) Menekan tombol "=" 3.) Menekan tombol "MC" 4.) Menekan tombol "4", "-", "MR" 5.) Menekan tombol "="	2.) Bagian atas TextBox menampilkan "1+1=2.0" dan bagian bawah TextBox menampilkan "2" 3.) TextBox menampilkan "0" 4.) TextBox menampilkan "4-2" 5.) Bagian atas TextBox menampilkan "4-2=2.0" dan bagian bawah TextBox menampilkan "2"	
--	--	--

#### 4.11 Test ZeroDivisionException

Yang Dilakukan	Yang Diharapkan	Hasil Keluaran
1.) Menekan tombol-tombol secara berurutan, yaitu "3", "/", "0". 2.) Menekan tombol "="	1.) Sebelum menekan tombol "=", TextBox menampilkan "3/0" 2.) Bagian atas TextBox menampilkan "Cannot divide by zero" dan bagian bawah TextBox menampilkan "3/0"	Sama dengan yang diharapkan

## 5. Pembagian Tugas

Package / modul	Class	Implementasi	Tester
button	Button	13518066 13518084	13518024 13518066 13518075 13518084
button	FunctionButton	13518084	13518024 13518066 13518075 13518084
button	NonFunctionButton	13518084	13518024 13518066 13518075 13518084
button	NumberButton	13518066	13518024 13518066 13518075 13518084
button	ButtonBuilder	13518084	13518024 13518066 13518075 13518084
button.function	Function	13518084	13518024 13518066 13518075 13518084

button.function	AC	13518024	13518024 13518066 13518075 13518084
button.function	C	13518024	13518024 13518066 13518075 13518084
button.function	MC	13518075	13518024 13518066 13518075 13518084
button.function	MR	13518075	13518024 13518066 13518075 13518084
button.function	OFF	13518024	13518024 13518075 13518084
button.function	Equal	13518084 13518024	13518024 13518066 13518075 13518084
exception	Exception	13518084	13518024 13518066 13518075 13518084

exception	MemoryException	13518075	13518024 13518075 13518084
exception	ArithmeticException	13518084	13518075 13518084
exception	ConversionException	13518084	13518024 13518075 13518084
exception	SquareRootOfNegativeException	13518084	13518024 13518066 13518075 13518084
exception	ZeroDivisionException	13518024 13518084	13518024 13518084
exception	OverflowException	13518024 13518084	13518075 13518084
expression	Expression	13518075 13518066 13518084	13518024 13518066 13518075 13518084
expression	UnaryExpression	13518075	13518024 13518066 13518075 13518084
expression	BinaryExpression	13518075	13518024

		13518066	13518066 13518075 13518084
expression	TerminalExpression	13518075 13518066	13518024 13518066 13518075 13518084
expression	NegativeExpression	13518075 13518066	13518024 13518066 13518075 13518084
expression	RootExpression	13518024 13518075	13518024 13518066 13518075
expression	SubtractExpression	13518024 13518075	13518024 13518066 13518075 13518084
expression	AddExpression	13518024 13518075 13518066	13518024 13518066 13518075 13518084
expression	DivideExpression	13518024 13518075 13518066	13518024 13518066 13518075 13518084





expression	MultiplyExpression	13518024 13518075 13518066	13518024 13518066 13518075 13518084
expression	PowerExpression	13518024	13518024 13518066
expression	ModuloExpression	13518024	13518024 13518075
expression	ExpressionBuilder	13518024 13518066 13518075 13518084	13518024 13518066 13518075 13518084
app	View	13518024 13518084	13518024 13518066 13518075 13518084
app	TextBox	13518024 13518075 13518084	13518024 13518066 13518075 13518084
-	Main	13518024 13518066 13518075 13518084	13518024 13518066 13518075 13518084



## 6. LAMPIRAN

Form Asistensi Tugas Besar  
IF2210/Pemrograman Berorientasi Objek Sem. 2 2019/2020

No. Kelompok/Kelas : G07 / K03  
Anggota Kelompok (NIM>Nama/TTD) :

1	135100024	Juan Kaura Cahyadi	
2	135100666	Ryhan Salura	
3	135100075	Daniel Rianto	
4	135100044	Jonathan Yudi G	
5			

Asisten Pembimbing : A. Fahmi  
No / Tanggal Asistensi : 0 / 12 Maret 2020

## Catatan Asistensi:

- 1) Khas reset "=" baru di evaluasi
- 2) maksimum inputan bebas
- 3) Try catch dilakukan untuk evaluasi (puncak = 1)

Tanda Tangan Asisten:

