NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

2. 1 Percobaan 1

```java
package P9;

public class Barang24 {
    int kode;
    String nama, kategori;

    Barang24(int kode, String nama, String kategori) {
        this.kode = kode;
        this.nama = nama;
        this.kategori = kategori;
    }
}
```

```java
package P9;

public class Gudang24 {
    Barang24[] tumpukan;
    int size;
    int top;

    Gudang24(int kapasitas) {
        size = kapasitas;
        tumpukan = new Barang24[size];
        top = -1;
    }

    boolean cekKosong() {
        return top == -1;
    }

    boolean cekPenuh() {
        return top == size - 1;
    }

    void tambahBarang(Barang24 brg) {
        if (!cekPenuh()) {
            top++;
            tumpukan[top] = brg;
            System.out.println("Barang " + brg.nama + " berhasil ditambahkan ke Gudang");
        } else {
            System.out.println(x:"Gagal! Tumpukan barang di Gudang sudah penuh");
        }
    }

    Barang24 ambilBarang24() {
        if (!cekKosong()) {
            Barang24 delete = tumpukan[top];
            top--;
            System.out.println("Barang " + delete.nama + " diambil dari Gudang");
            return delete;
        } else {
            System.out.println(x:"Tumpukan barang kosong");
            return null;
        }
    }
```

```java
Barang24 lihatBarangTeratas() {
    if (!cekKosong()) {
        Barang24 barangTeratas = tumpukan[top];
        System.out.println("Barang teratas : " + barangTeratas.nama);
        return barangTeratas;
    } else {
        System.out.println(x:"Tumpukan barang kosong.");
        return null;
    }
}

void tampilkanBarang() {
    if (!cekKosong()) {
        System.out.println(x:"Rincian tumpukan barang di gudang:");

        for (int i = top; i >= 0; i--) {
            System.out.printf(format:"Kode %d: %s (Kategori %s)\n", tumpukan[i].kode, tumpukan[i].nama,
                    tumpukan[i].kategori);
        }
    } else {
        System.out.println(x:"Tumpukan barang kosong.");
    }
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 21
Masukkan nama barang: Majalah
Masukkan nama kategori: Buku
Barang Majalah berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 26
Masukkan nama barang: Jaket
Masukkan nama kategori: Pakaian
Barang Jaket berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 2
Barang Jaket diambil dari Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 33
Masukkan nama barang: Pizza
Masukkan nama kategori: Makanan
Barang Pizza berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 3
Rincian tumpukan barang di gudang:
Kode 33: Pizza (Kategori Makanan)
Kode 21: Majalah (Kategori Buku)
```

Pertanyaan :

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

1. *Lakukan perbaikan pada kode program, sehingga keluaran yang dihasilkan sama dengan verifikasi hasil percobaan! Bagian mana saja yang perlu diperbaiki?*

   Jawab :
   ```
   boolean menu = true;
   ```
   ```
   while (menu) {
   ```
   ```
   case 4:
       menu = false;
       break;
   ```
   Tambahkan variable boolean untuk menu nomor 4 yaitu keluar, sehingga program akan keluar dari while ketika variable tersebut bervalue false
   ```
   Barang24 lihatBarangTeratas() {
       if (!cekKosong()) {
   ```
   Ganti !isEmpty ke !cekKosong, karena method yang kita buat berma,a cekKosong

2. *Berapa banyak data barang yang dapat ditampung di dalam tumpukan? Tunjukkan potongan kode programnya!*

   Jawab : 7 Data barang
   ```
   Gudang24 gudang = new Gudang24(kapasitas:7);
   ```
   ```
   Gudang24(int kapasitas) {
       size = kapasitas;
       tumpukan = new Barang24[size];
       top = -1;
   }
   ```

3. *Mengapa perlu pengecekan kondisi !cekKosong() pada method tampilkanBarang? Kalau kondisi tersebut dihapus, apa dampaknya?*

   Jawab : Untuk memastikan bahwa stack tidak dalam keadaan kosong. Jika kondisi dihapus, maka akan menghasilkan error ketika method dipanggil, karena top bernilai -1 yang tidak menunjuk alamat apapun pada array.

4. *Modifikasi kode program pada class Utama sehingga pengguna juga dapat memilih operasi lihat barang teratas, serta dapat secara bebas menentukan kapasitas gudang!*

   Jawab :
   ```
   case 4:      Barang24 P9.Gudang24.
       gudang.lihatBarangTeratas();
       break;
   ```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 21
Masukkan nama barang: Majalah
Masukkan nama kategori: Buku
Barang Majalah berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 26
Masukkan nama barang: Jaket
Masukkan nama kategori: Pakaian
Barang Jaket berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 4
Barang teratas : Jaket
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 2
Barang Jaket diambil dari Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 4
Barang teratas : Majalah
```

5. *Commit dan push kode program ke Github*

2.2 Percobaan 2

```java
package P9;

public class StackKonversi24 {
    int size, top;
    int[] tumpukanBiner;

    StackKonversi24() {
        this.size = 32;
        tumpukanBiner = new int[size];
        top = -1;
    }

    boolean isEmpty() {
        return top == -1;
    }

    boolean isFull() {
        return top == size - 1;
    }

    void push(int data) {
        if (isFull()) {
            System.out.println(x:"Stack penuh.");
        } else {
            top++;
            tumpukanBiner[top] = data;
        }
    }

    int pop() {
        if (isEmpty()) {
            System.out.println(x:"Stack kosong.");
            return -1;
        } else {
            int data = tumpukanBiner[top];
            top--;
            return data;
        }
    }
}
```

```java
String konversiDesimalkeBiner(int kode) {
    StackKonversi24 stack = new StackKonversi24();
    while (kode > 0) {
        int sisa = kode % 2;
        stack.push(sisa);
        kode = kode / 2;
    }
    String biner = new String();
    while (!stack.isEmpty()) {
        biner += stack.pop();
    }
    return biner;
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

```
    System.out.println("Kode unik dalam biner: " + konversiDesimalkeBiner(delete.kode));
    return delete;
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 13
Masukkan nama barang: Setrika
Masukkan nama kategori: Elektronik
Barang Setrika berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih Operasi: 2
Barang Setrika diambil dari Gudang
Kode unik dalam biner: 1101
```

Pertanyaan :

1. *Pada method konversiDesimalKeBiner, ubah kondisi perulangan menjadi while (kode != 0), bagaimana hasilnya? Jelaskan alasannya!*
   Jawab : Hasil akan sama dengan kondisi kode > 0. Hal ini dikarenakan semua angka jika dibagi 2 akan menghasilkan angka selain 0 kecuali angka 1. Sehingga ketika kode dibagi hingga menunjukkan hasil 1, kode akan menghasilkan angka 0.5 yang jika dibaca integer akan menjadi 0.

2. *Jelaskan alur kerja dari method konversiDesimalKeBiner!*
   Jawab : Pada perulangan pertama dilakukan pengisian pada stack dengan angka hasil dari modulo kode dengan 2, angka ini (0 atau 1) akan dimasukkan kedalam stack secara berurutan hingga kode == 0. Setelah kode habis, maka perulangan yang kedua melakukan pop pada stack yang kemudian dimasukkan kedalam String biner, sehingga biner akan mengambil data dari atas stack kemudian meletakkannya sebagai huruf paling kiri hingga stack habis.

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

2.3 Percobaan 3

```java
package P9;

public class Postfix24 {
    int n, top;
    char[] stack;

    Postfix24(int total) {
        n = total;
        top = -1;
        stack = new char[n];
        push(c:'(');
    }

    void push(char c) {
        top++;
        stack[top] = c;
    }

    char pop() {
        char item = stack[top];
        top--;
        return item;
    }

    boolean isOperand(char c) {
        return (c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z') || (c >= '0' && c <= '9') || c == ' ' || c == '.';
    }

    boolean isOperator(char c) {
        return c == '^' || c == '%' || c == '/' || c == '*' || c == '-' || c == '+';
    }
```

```java
package P9;

import java.util.Scanner;

public class PostfixMain24 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String P, Q;
        System.out.print(s:"Masukkan ekspresi matematika (infix): ");
        Q = sc.nextLine();
        Q = Q.trim();
        Q = Q + ")";

        int total = Q.length();

        Postfix24 post = new Postfix24(total);
        P = post.konversi(Q);
        System.out.println("Postfix: " + P);

    }
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

```
int derajat(char c) {
    switch (c) {
        case '^':
            return 3;
        case '%':
            return 2;
        case '/':
            return 2;
        case '*':
            return 2;
        case '-':
            return 1;
        case '+':
            return 1;
        default:
            return 0;
    }
}

String konversi(String Q) {
    String P = "";
    char c;
    for (int i = 0; i < n; i++) {
        c = Q.charAt(i);
        if (isOperand(c)) {
            P = P + c;
        }
        if (c == '(') {
            push(c);
        }
        if (c == ')') {
            while (stack[top] != '(') {
                P = P + pop();
            }
            pop();
        }
        if (isOperator(c)) {
            while (derajat(stack[top]) >= derajat(c)) {
                P = P + pop();
            }
            push(c);
        }
    }
    return P;
}
```

```
Masukkan ekspresi matematika (infix): a+b*(c+d-e)/f
Postfix: abcd+e-*f/+
```

Pertanyaan :

1. *Pada method derajat, mengapa return value beberapa case bernilai sama? Apabila return value diubah dengan nilai berbeda-beda setiap case-nya, apa yang terjadi?*

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : STACK

Jawab : Hal ini dikarenakan beberapa operator memiliki derajat yang sama seperti + dan – ataupun / dan *. Dalam hal ini return value dapat diubah namun harus tetap sesuai dengan urutan derajat, sehingga operator yang memiliki derajat sama harus diberikan nilai value yang berdekatan(boleh tidak urut) namun tetap memperhatikan nilai derajat dari operator lain.

2. *Jelaskan alur kerja method konversi!*
   Jawab : Pertama-tama deklarasikan String kosongan sebagai tempat menyimpan hasil postfix. Kemudian lakukan perulangan sebanyak panjang infix untuk melakukan pengecekan pada setiap char pada String yang akan diubah. Lalu jika char dengan index i(sesuai perulangan) merupakan operand, maka akan langsung dimasukkan ke String kosongan tadi, jika char[i] merupakan ‘(’ maka masukkan ‘(’ ke stack, jika char[i] adalah ‘)’ keluarkan satu persatu operator pada stack, masukkan ke String infix hingga menemukan ‘(’ lalu keluarkan ‘(’ dari stack, dan jika char[i] adalah operator, bandingkan dengan operator pada stack, jika derajatnya lebih tinggi, keluarkan operator dari stack,

3. Pada method konversi, apa fungsi dari potongan kode berikut?
   ```
   c = Q.charAt(i);
   ```
   Jawab : Memasukkan huruf ke-i pada String Q untuk dibandingkan pada method.