

**Nama : Fadia Nur Shafitri**

**NPM : G1F021010**

**Tugas ASD / Stack (Prefix, Infix, Post Fix)**

### **Soal**

1. Tulis algoritma untuk mengonversi ekspresi infiks ke postfix dan infix ke prefix
2. Tulis algoritma untuk mengevaluasi ekspresi postfix dan prefix

### **Pembahasan**

#### **Algoritma untuk mengonversi ekspresi infiks ke postfix**

1. Inisialisasi struktur data dengan membuat sebuah stack kosong, baca ungkapan dalam bentuk infix, dan tentukan derajat operator misalnya ( : 0 ; + & - : 1; \* & / : 2; ^ : 3).
2. Lakukan pembacaan karakter dari Infix, berikan ke R.
3. Test Nilai R, Jika a. ( Langsung di Push b. Operand, Langsung di Tulis c. ) lakukan Pop sampai ketemu buka kurung, tetapi tanda kurung tidak perlu di tulis. d. Operator, Jika stack dalam keadaan kosong atau derajat R lebih tinggi dibandingkan dengan di ujung stack, lakukan Push, jika tidak lakukan POP.
4. Jika pembacaan terhadap infix sudah selesai, namun stack belum kosong lakukan POP.

#### **Algoritma untuk mengonversi ekspresi infiks ke prefix**

1. Menulis ekspresi dalam bentuk infix.
2. Menyiapkan stack kosong.

```
public StackX(int s)
{
    maxSize = s;
    stackArray = new long[maxSize];
    top
    = -1;
}
```

3. Menyiapkan fungsi push ( ) dan pop ( ).

- push( ) public void  
push(long j)  
{ stackArray[++top] =  
j;  
}
- pop( ) public long  
pop()  
{ return stackArray[top--]  
];  
}

4. Scan variabel dari kanan ke kiri dalam bentuk infix.
5. Jika operand langsung ditulis di notasi.
6. Jika ketemu tutup kurung " ) ", maka akan dipush ke stack.
7. Jika ketemu tanda kurung " ( ", maka pop( ) bagian top dari stack hingga tanda tutup kurung " ) " dan ditulis ke dalam notasi. Tanda tutup kurung " ) " juga di pop ( ) tapitidak perlu ditulis ke dalam notasi.
8. Jika operator dan stack masih kosong, maka push ( ) operator ke dalam stack.
9. Jika operator sebelumnya yang berada pada stack memiliki derajat yang lebih rendah, maka push ( ) operator yang lebih tinggi derajatnya ke dalam stack.
10. Jika operator sebelumnya yang berada pada stack memiliki derajat yang lebih tinggi, maka pop ( ) operator yang berada pada stack dan ditulis ke notasi. Sedangkan operator yang derajatnya lebih rendah di push ke dalam stack.
11. Ulangi langkah 5 sampai 10.
12. Jika ekspresi telah berakhir dan stack belum kosong, maka pop ( ) isi stack dan tulsi kedalam notasi.

### **Algoritma untuk mengevaluasi ekspresi postfix**

1. Scan sting Postfix dari kiri ke kanan.
2. Bila ketemu operand, Push(operand).
3. Bila ketemu operator, Pop dua kali yaitu Pop(X) dan Pop(Y).
4. Z= Y operator X 5. Push (Z).
5. Ulangi langkah 2 s/d 5 hingga seluruh simbol di dalam stirng terbaca.

### **Algoritma untuk mengevaluasi ekspresi Infix**

Algoritma Evaluasi Infix sudah lengkap, yaitu terdiri dari:

1. Algoritma konversi Infix ke Postfix.
2. Algoritma Evaluasi Postfix.