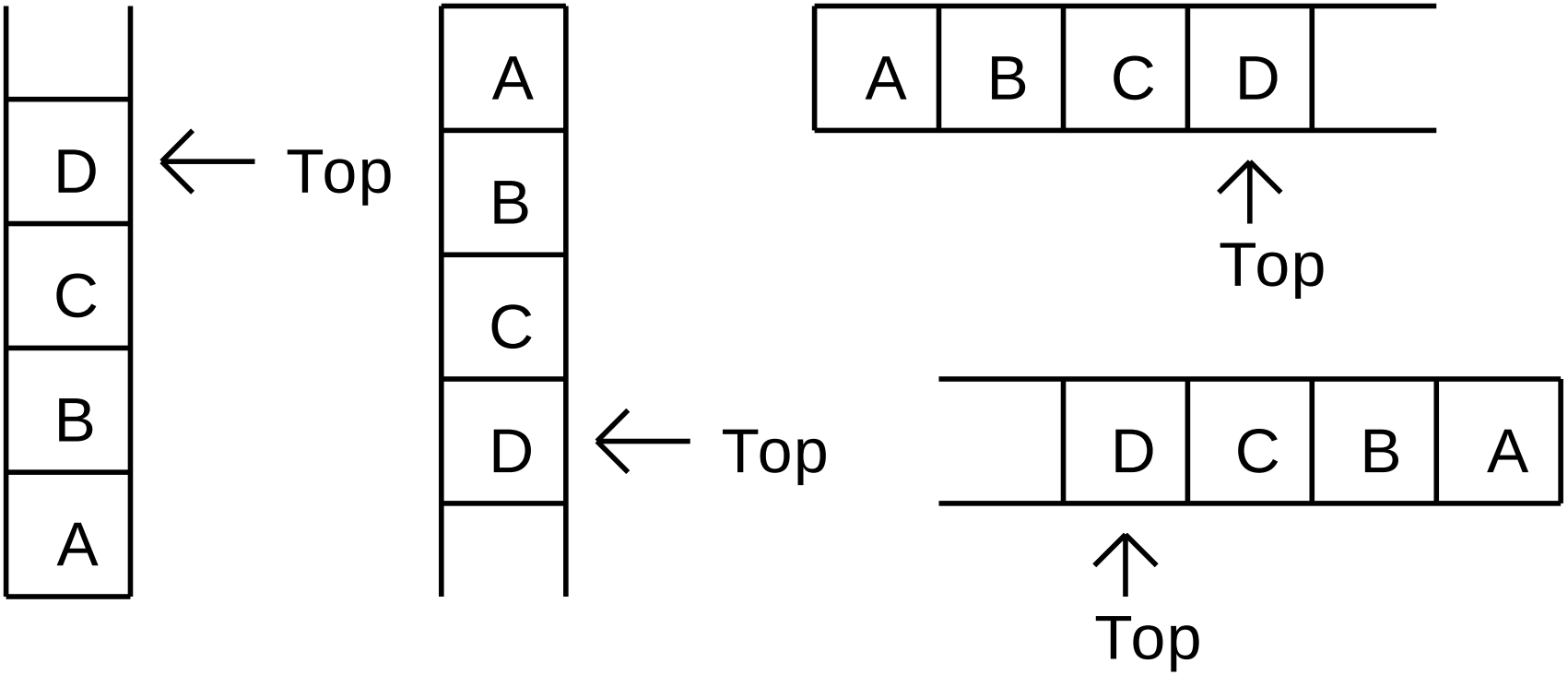
STACK (2)

Brigida Arie Minartiningtyas, M.Kom

Review Stack

- What is Stack??
  - Tumpukan data yang seolah-olah ada data di atas data lain.
  - Konsep utama dalam STACK adalah LIFO (Last In First Out).
  - Penghapusan serta pemasukan elemen pada stack hanya dapat dilakukan di satu posisi, yakni posisi akhir.


D
C
B
A
Top
A
B
C
D
Top
A
B
C
D
Top
D
C
B
A
Top

Operasi pada stack??

- Push
  - Penyisipan (insert)
- Pop
  - Penghapusan (delete)

Deklarasi Awal Stack

```
const  MaxElemen = 255;
type Tumpukan = record
     Isi : array[1..MaxElemen] of integer;
     Atas : 0..MaxElemen;
end;

var T : Tumpukan;

Procedure awal
  begin
            t.atas :=0;
  end;
```

Deklarasi Push

```
procedure PUSH (var T : Tumpukan; nil =
  integer);
  Begin
      if (T.Atas = MaxElemen) then
            write('Tumpukan PENUH Coy…!!');
      else
            begin
                  T.Atas := T.Atas + 1;
                  T.Isi[T.Atas] := nil;
      end;
   end;
```

Deklarasi Pop

```
procedure POP (var T : Tumpukan);
Begin
      if (T.Atas = 0) then
          write('Tumpukan KOSONG Coy…!!');
      else
      begin
        Write('nilai yang diambil : ',T.Isi[T.Atas]);
        T.Atas := T.Atas - 1;
   end;
end;
```

APLIKASI STACK

Pemanfaatan Stack

- Menulis ungkapan-ungkapan menggunakan notasi tertentu
- Dalam penulisan ungkapan **(numeris)**, kita selalu menggunakan **tanda kurung** untuk mengelompokkan bagian mana yang harus dikerjakan lebih dahulu.

Contoh 1

- ((A + B) * (C – D))
- Urutan pengerjaan
  1. Suku (A+B)
  2. Suku (C-D)
  3. Mengalikan hasil dari suku 1 dan suku 2

Contoh 2

- A + B * C – D
- Urutan pengerjaan
  - B*C akan dikerjakan lebih dulu
  - Operator * (perkalian) mempunyai derajat lebih tinggi dibanding operator + (penjumlahan) dan operator – (pengurangan)

Infix

- Cara penulisan ungkapan sering disebut dengan notasi Infix
- **<u>INFIX</u>** : Operator ditulis di antara dua operand
  - Operator??
  - Operand??

Operator

A+B*C

Operand

- Semakin rumit suatu ungkapan semakin banyak yang dibutuhkan tanda kurung.
- Ahli matematikawan **Jan Lukasiewicsz** mengembangkan suatu cara penulisan ungkapan numeris
  - Notasi **Prefix** atau Notasi **Postfix**

Infix	Prefix	Postfix
A+B	+AB	AB+
A+B-C	-+ABC	AB+C-
(A+B)*(C-D)	*+AB-CD	AB+CD-*
A-B/(C*D^E)	-A/B*C^DE	ABCDE^*/-

Infix ke Postfix

Algoritma Infix → Postfix (1)

1. Hanya **OPERATOR** yang disimpan ke Stack
2. Bila isi variabelnya **OPERAND**, maka langsung cetak variabel
3. Operator **kurung buka** dan **kurung tutup** bisa disimpan ke Stack, namun tidak ditulis ke layar
4. Bila isi variabelnya **Kurung Buka** '(', maka simpan (PUSH) Kurung Buka ke Stack.
5. Bila isi variabelnya **OPERATOR** maka periksa:
   a. Bila Stack kosong, maka simpan Operator sekarang ke Stack
   b. Bila Stack paling atas berisi Operator selain tanda kurung, maka bandingkan derajat di Stack dengan variabel sekarang.

Algoritma Infix → Postfix (2)

i. Bila lebih rendah di Stack, maka variabel langsung di-PUSH
ii. Bila lebih tinggi di Stack, maka Stack paling atas di-POP / dicetak, kemudian variabel sekarang masuk/ di-PUSH
iii. Bila Stack paling atas isinya kurung buka '(', maka variabel sekarang langsung di-PUSH/ disimpan ke Stack.
iv. Bila variabel sekarang isinya kurung tutup ')', maka cetak semua Operator di Stack hingga bertemu tanda kurung buka '('.
v. Bila variabel yang ada telah habis diproses dan Stack masih berisi, maka cetak semua Operator yang ada di dalam Stack hingga Stack kosong melomponk

Derajat Operator

Derajat	Operator	Simbol
3	Pangkat	^
2	Kali, bagi	*, /
1	Tambah, kurang	+, -
0	Kurung buka, kurung tutup	(,)

(A+B)/((C-D)*E^F)

Proses Ke	Karakter Dibaca	Isi Tumpukan	Karakter Tercetak	Notasi Postfix yang Terbentuk
1	(	(		
2	A	(	A	A
3	+	+(		A
4	B	+(	B	AB
5	)		+	AB+
6	/	/		AB+
7	(	(/		AB+
8	(	((/		AB+
9	C	((/	C	AB+C
10	-	-((/		AB+C
11	D	-((/	D	AB+CD
12	)	(/	-	AB+CD-
13	*	*(/		AB+CD-
14	E	*(/	E	AB+CD-E
15	^	^*(/		AB+CD-E
16	F	^*(/	F	AB+CD-EF
17	)	*(/	^	AB+CD-EF^
18		(/	*	AB+CD-EF^*
19		/		AB+CD-EF^*
20			/	AB+CD-EF&*/