Pertemuan 7

Waktu : 135 menit

Tujuan Pembelajaran : Mahasiswa mampu menjelaskan teknik pemrograman menggunakan Stack.

Substansi Materi : Stack

Tabulasi Kegiatan Perkuliahan

No	Tahap Kegiatan	Kegiatan Pengajar	Kegiatan Mahasiswa	Media & Alat	Waktu
1	Pendahuluan	1. Membuka pertemuan 2. Mengulang materi pertemuan sebelumnya	Menyimak Bertanya	Papan Tulis	20 Menit
2	Penyajian Materi	1. Pengertian stack 2. Jenis-jenis stack 3. Stack dengan Array	Menyimak Bertanya Menjawab Pertanyaan	Papan Tulis	80 Menit
3	Penutup	1. Menyimpulkan materi pertemuan 2. Memberikan tugas kecil 3. Menutup pertemuan	Menyimak	Papan tulis	35 Menit

MATERI KULIAH

STACK

Stack adalah suatu tumpukan. Konsep utama dari stack adalah LIFO (Last In First Out), yaitu benda yang terakhir masuk ke dalam stack akan menjadi benda pertama yang dikeluarkan dari tumpukan. Dalam pascal ada dua cara penerapan stack, yaitu dengan array dan linked list.

Single Stack dengan Array

Sesuai dengan sifat stack, maka pengambilan/penghapusan elemen dalam stack harus dimulai dari elemen teratas. Deklarasi konstanta, tipe, dan variable yang akan dipakai dalam penjelasan operasi-operasi stack dengan array adalah :

```
Const
Max = {jumlah tumpukan}
Type
        TipeData = {   };
        Stack = array [1..Max] of TipeData;
Var
        Top : TipeData;
```

Operasi-operasi pada Single Stack dengan Array

- Create : Membuat stack baru yang masih kosong

```
Procedure Create;
  Begin
        Top := 0;
  End;
```

- Full : Fungsi untuk memeriksa apakah stack yang ada sudah penuh

```
Function Full : Boolean;
  Begin
        Full := False;
        If top = max then Full := True;
  End;
```

- Push : Menambahkan sebuah elemen-elemen ke dalam stack. Tidak bisa dilakukan lagi jika stack sudah penuh.

```
Procedure Push(elemen:TipeData);
  Begin
      If not Full then
        Begin
            Top := Top+1; { atau Inc(Top) }
            Stack[Top] := elemen;
        End;
  End;
```

- Empty : Fungsi untuk menentukan apakah stack kosong atau tidak.

```
Function Empty : Boolean;
  Begin
  Empty := False;
  If Top = 0 then Empty := True;
  End;
```

- Pop : Mengambil elemen teratas dari stack. Stack tidak boleh kosong.

```
Procedure Pop(elemen:TipeData);
  Begin
      If not Empty then
        Begin
            Elemen := stack[Top];
            Top := Top-1; { atau Dec(Top) }
        End;
  End;
```

- Clear : Mengosongkan stack (Jika top = 0, maka stack dianggap kosong)

```
Procedure Clear;
  Begin
      Top := 0;
  End;
```

Double Stack dengan Array

Merupakan teknik yang dikembangkan untuk menghemat pemakaian memory dalam pembuatan dua stack dengan array. Intinya adalah menggunakan sebuah array untuk menampung dua stack.

Contoh deklarasi konstanta, tipe, dan variable yang akan dipakai dalam operasi-operasi double stack array.

```
Const
Max = {jumlah tumpukan}
Type
      TipeData = {   };
      Stack = array [1..Max] of Byte;
Var
      Top : array[1..2] of Byte;
```

Operasi-operasi pada Double Stack dengan Array

- Create : Membuat stack baru yang masih kosong

```
Procedure Create;
  Begin
      Top[1] := 0;
      Top[2] := max + 1;
```

```
End;
```

- Full : Fungsi untuk memeriksa apakah stack yang ada sudah penuh

```
Function Full : Boolean;
  Begin
      Full := False;
      If top[1]+1 > = top[2] then Full := True;
  End;
```

- Push : Menambahkan sebuah elemen-elemen ke dalam stack. Tidak bisa dilakukan lagi jika stack sudah penuh.

```
Procedure Push(elemen:TipeData; NoStack : Byte);
  Begin
      If not Full then
        Begin
            Case NoStack of
            1 : Top[1] := Top[1] + 1;
            2 : Top[2] := Top[2] -1;
        Stack[Top[NoStack]] := elemen;
        End;
  End;
```

- Empty : Fungsi untuk menentukan apakah stack kosong atau tidak.

```
Function Empty(NoStack : Byte) : Boolean;
  Begin
  Empty := False;
  Case NoStack of
        1 : if Top[1]=0 then
          Empty := True;
        2 : if Top[2] = Max + 1 then
          Empty := True;
  End;
  End;
```

- Pop : Mengambil elemen teratas dari stack. Stack tidak boleh kosong.

```
Procedure Pop(var elemen:TipeData; NoStack:Byte);
  Begin
        If not Empty(NoStack) then
          Begin
                Elemen := stack[Top[NoStack]];
                Case NoStack of
                  1 : Top[1] := Top[1] -1;
                  2 : Top[2] := Top[2] +1;
                End;
          End;
  End;
```

- Clear : Mengosongkan stack (Jika top = 0, maka stack dianggap kosong)

```
Procedure Clear(NoStack:Byte);
  Begin
      Case NoStack Of
        1 : Top[1] := 0;
        2 : Top[2]:= Max + 1;
      End;
  End;
```