Pointer

Struktur Data Pascal

Pengertian

Pada materi sebelumnya telah dijelaskan mengenai variabel bertipe array, suatu tipe data yang bersifat statis (ukuran dan urutannya sudah pasti). Selain itu ruang memori yang dipakai olehnya tidak dapat dihapus bila variabel bertipe array tersebut sudah tidak digunakan lagi pada saat program dijalankan. Untuk memecahkan masalah diatas, kita dapat menggunakan variabel pointer. Tipe data pointer bersifat dinamis, variabel akan dialokasikan hanya pada saat dibutuhkan dan sesudah tidak dibutuhkan dapat dialokasikan kembali.

Tabel Perbandingan

Kriteria	Array	Pointer
Sifat	Statis	Dinamis
Ukuran	Pasti	Sesuai kebutuhan
Alokasi variabel	Saat program dijalankan sampai selesai	Dapat diatur sesuai kebutuhuan

Deklarasi

Bentuk umum :

Var <namavar> : <^tipedata>

Contoh :

```
Var
   Jumlahdata   : ^integer;';
   Namasiswa    :^string[25];
  Nilaisiswa :^real;
```

Deklarasi Umum dalam Record

Bentuk umum :

```
Tipe
    <namapointer> = <^namarecord>;
    <namarecord> =  record

<item1>:<tipedata1>;

<item2>:<tipedata2>;
                        ...
                        <itemN>:<tipedataN>;
                    end;
Var
    <namavar>:<namapointer>;
```

Contoh Program

```
Type   Pemployee =^Temployee;
Temployee = record
     Name     : string[10];
      Position : char;
       Salary   : longint;        end;
   var   p : pemployee; begin
new(p);
p^.name:='Clark'
p^.position:='S';
p^.salary:=3000;
writeln(p^.name,' ',p^.position,' ',p^.salary);
dispose(p);
 end.
```

Linked List

Apabila setiap kali anda ingin menambahkan data selalu dengan menggunakan variabel pointer yang baru, anda akan membutuhkan banyak sekali variabel pointer(penunjuk).

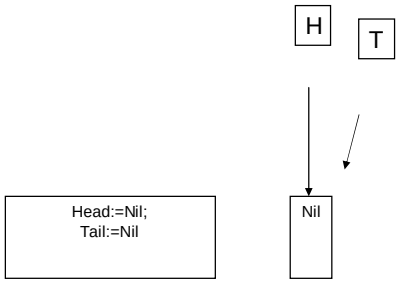
Oleh karena itu ada baiknya jika anda hanya menggunakan satu variabel pointer saja untuk menyimpan banyak data dengan metode yang kita sebut ***Linked List***. Jika diterjemahkan, maka berarti suatu daftar isi yang saling berhubungan

Contoh Penggunaan dlm FIFO (queue)

```
program fifo;
uses crt;
Const=4;
Type
 Point     = ^RecPoint;
 Recpoint = Record
        nama  : string;
        umur  : integer;
        Next    : Point;
        End;
Var
 Head, Tail, Now : Point;
 n:String;
 u,pilih:integer;
```

Create

```
Procedure Create;
   Begin
      Head:=nil;
      Tail:=nil;
   End;
```



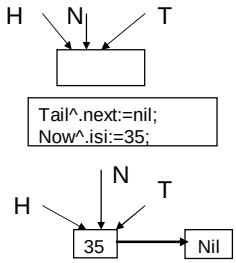
Menginput

```
Procedure INSERT(elemen1:string;elemen2:integer);
Var Now:Point;
   Begin
      New(Now);
      If head = nil then
      Head:=now
      else
      Tail^.next:=now;
      Tail:=Now;
      Tail^.next:=nil;
      Now^.nama:=elemen1;
         Now^.umur:=elemen2;
End;
```

Input

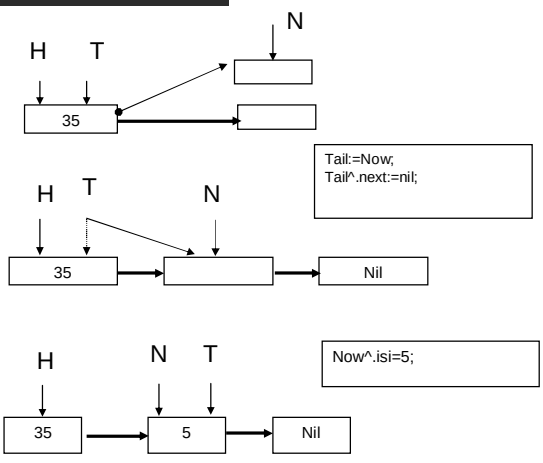
Insert (35)

```
New(Now);
Head=Nil then
        Head:=Now;
Tail:=Now;
```



Insert (5)

```
New(Now);
Head<>nil then
        Tail^.next:=now;
```



Mengecek Posisi Pointer

```
Procedure Cekpointer;
begin
Writeln
('nama now ',now^.nama);
Writeln
('umur now ',now^.umur);
writeln('nama head ',head^.nama);
writeln('umur head ',head^.umur);
writeln('nama tail ',tail^.nama);
writeln('umur tail ',tail^.umur);
readln;
End;
```

Pengecekan, Pencarian

```
Function Empty : Boolean;
Begin
    If head = nil then
    Empty:= true
    else
    empty:= false;
end;
Function Full : Boolean;
Begin
    If head = max then
    Full:= true
    else
    Full:= false;
end;
```

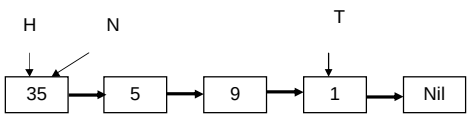
Pencarian...

```
Procedure Find_First;
   Begin
   Now:= head;
   End;

Procedure Find_Next;
   Begin
      If Now^.next <> nil then
         Now:= Now^.next;
   End;
```

Find First

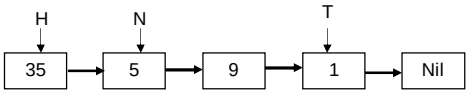
Find_First

```
Procedure Find_First;
        Begin
                Now:= head;
        End;
```



Find Next

```
Procedure Find_Next;
        Begin
                If Now^.next <> nil then
                  Now:= Now^.next;
        End;
```



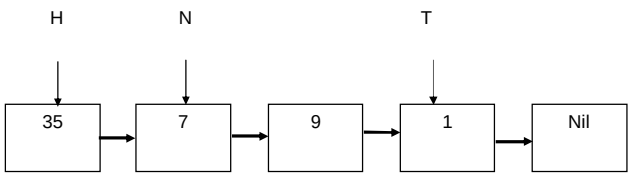
Menyimpan Temp dan Mengupdate

```
Procedure Retrieve;
 Begin
    n:= Now^.nama;
      u:= now^.umur;
 End;

Procedure Update(elemen1: string; elemen2:integer );
 Begin
     Now^.nama:=elemen1;
       now^.umur:=elemen2;

 End;
```

Update

Update(7)

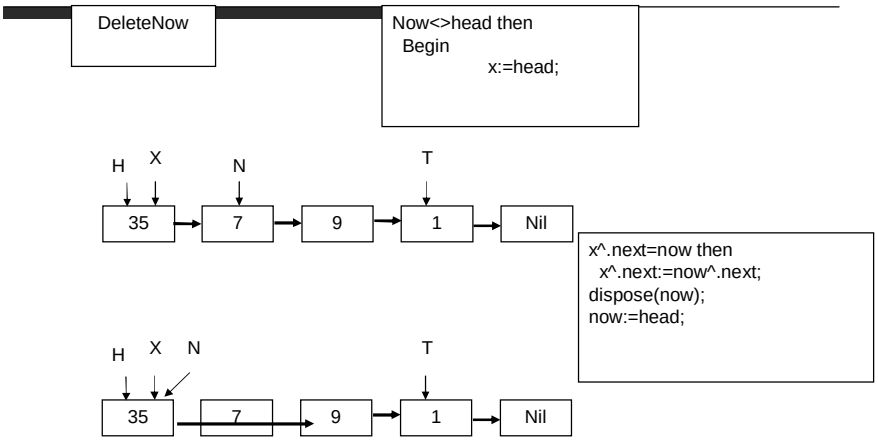
```
Procedure Update(u: TipeData );
Begin
        Now^.isi:=7;
End;
```



Delete Now

```
Procedure DeleteNow;
Var x : point;
   Begin
      If now<>head then
        Begin
          x:=head;
          while x^.next<>now do
            x:=x^.next;
          x^.next:=now^.next;
         end
      else head:= head^.next;
            dispose(Now);
      Now:= head;
   End;
```

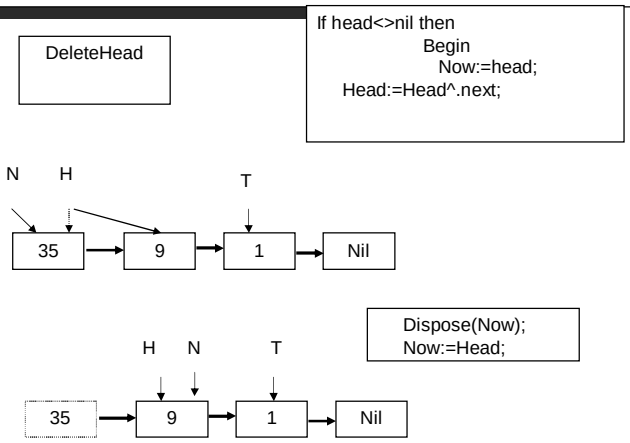
Delete Now



Delete Head

```
Procedure DeleteHead;
  Begin
   If head<>nil then
     Begin
        Now:=head;
        Head:=Head^.next;
        Dispose(Now);
        Now:=Head;
   End;
  End;
```

Delete Head



Clearing

```
Procedure Clear;
 Begin
  While head <> nil do
     Begin
        Now:=head;
        Head:=head^.next;
        Dispose(Now);
     End;
 End;
```