Pointer

(TIB11 – Struktur Data)

Pertemuan 7, 8

Sub-CPMK

- Mahasiswa mampu menggunakan variabel *pointer* pada pemrograman untuk menyimpan dan menampilkan data pada suatu alamat memori (C3, A3)

Materi:

1. Pengertian *Pointer*
2. *Variabel Pointer*
3. *Record Pointer*

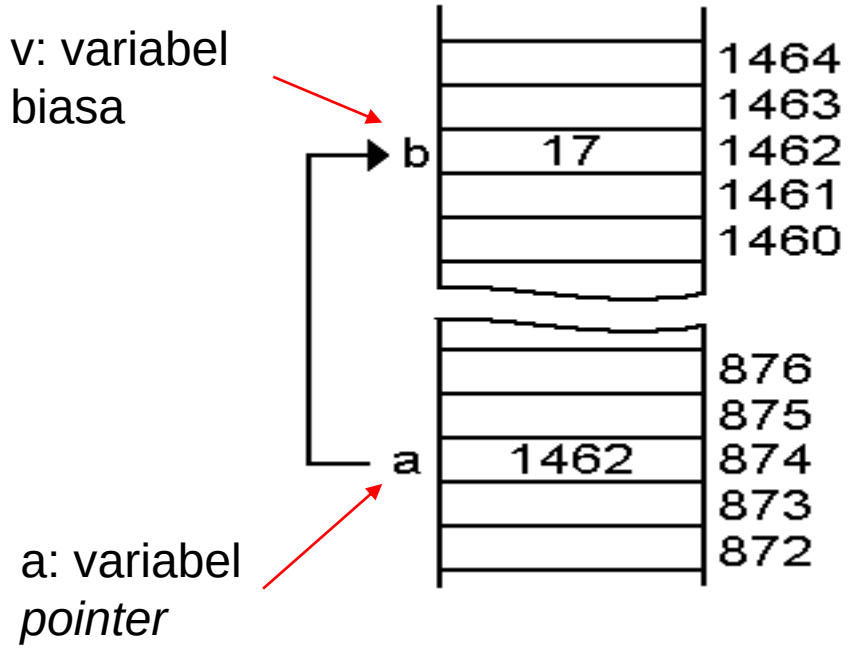
1. Pengertian *Pointer*

1.1. *Pointer*

- *Pointer* adalah variabel yang berisi alamat dari suatu *memory* yang menyimpan data

1.2. Contoh *Pointer*

- Variabel *Pointer a* menunjuk ke alamat variabel *b*.
- variabel *b* menyimpan sebuah bilangan (17)
- Variabel *a* dapat menyimpan alamat dari variabel *b* pada *memory* (1462)



1.3. Alamat *Memory* dan Variabel *Pointer*

- Untuk mengakses data pada alamat *memory* dapat dilakukan hanya dengan mengetahui alamatnya.
- Alamat data tersebut harus tersimpan pada suatu variabel agar dapat diakses melalui variabel yang menyimpan alamat sel *memory* tersebut

1.4. Variabel Dinamis

- Ketika suatu variabel dideklarasikan, maka sistem operasi akan mencari suatu alamat di *memory* dan alamat serialnya yang mampu menampung panjang data yang dideklarasikan pada variabel tersebut
- Demikian juga pada deklarasi suatu variabel *pointer*, maka sistem operasi akan mencari alamat *memory* yang mampu menampung panjang data dengan tipe data *pointer*

1.5. Alokasi *Memory*

- Alokasi *memory* dimaksudkan untuk mencari dan memesan suatu alamat *memory* dengan sel serialnya untuk didaftarkan menyimpan data dengan tipe data yang dimaksudkan untuk ditunjuk oleh suatu variabel pointer
- Penggunaan tipe data *pointer* dimaksudkan tidak untuk tujuan menghemat *memory*, karena selain mendeklarasikan variabel bertipe *pointer*, maka program juga perlu mengalokasikan alamat pada *memory* yang mampu menampung tipe data dari data yang sebenarnya hendak disimpan

2. Variabel *Pointer*

- Variabel *pointer* dimaksudkan menyimpan alamat awal dari suatu *memory* yang berisi sel-sel *memory* yang dialokasikan untuk menyimpan data sesuai dengan tipe data yang dialokasikan

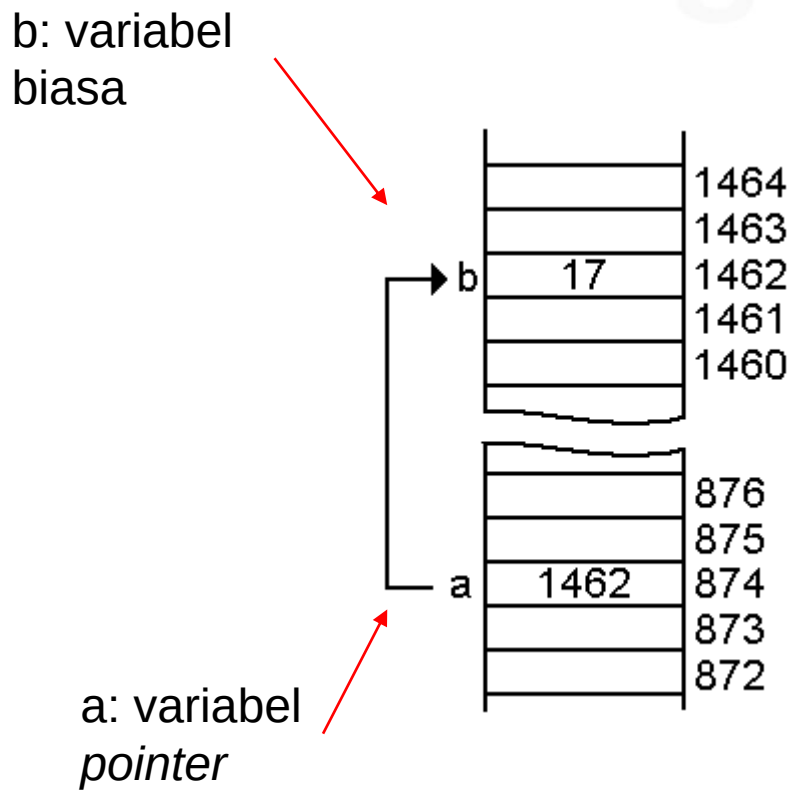
2.1. Deklarasi Variabel bertipe *Pointer*

- Variabel *Pointer* dapat dideklarasikan dengan mencantumkan karakter * di depan variabel yang dideklarasikan
- Contoh:

  int *a;
- Deklarasi ini akan mendaftarkan variabel a untuk menyimpan data bertipe *pointer* yang ditujukan untuk menunjuk ke alamat yang berisi data int

2.2. Menyimpan Alamat *Memory*

- Kembali ke contoh di awal
- Variabel a merupakan variabel *pointer* yang akan menyimpan alamat *memory* dari variabel b
- Karena itu a di deklarasikan dengan perintah:
  int *a;
- Sedangan b dideklarasikan dengan perintah
  int b
- Sedangan b dideklarasikan dengan perintah
  int b
- Alamat *memory* dari variabel b yaitu 1462 dapat disimpan pada variabel a, dengan mencantumkan karakter &didepan variabel b.
  a = &b;



2.3. Alokasi *Memory*

- Pemilihan alamat *memory* yang menyimpan data yang sebenarnya dapat juga dilakukan melalui data yang bertipe *pointer* dengan perintah

```
a = (tipeData*)malloc(sizeof(tipeData));
```

- Contoh

```
int *a
a = (int *)malloc(sizeof(int));
```

3. *Record Pointer*

3.1. Pengunaan *Pointer* pada *Linked-list*

- Pengunaan *Pointer* pada *Linked-list* merupakan suatu hal yang biasa dilakukan.
- Penggunaan ini memerlukan *define* suatu *struct/Record*, yang mana *struct/record* ini diperlakukan sebagai mana layaknya tipe data.
- Struct tersebut digunakan untuk Deklarasi variabel *pointer* yang menyimpan alamat awal data yang sesuai dengan struktur *record* tersebut pada suatu *memory*

3.2. *Pointer - Pascal*

- *Record definition*

```
Type
   RecordName = Record
     VarName : vartype;
     NextPointer : ^RecordName;
   End;
   PointerName = ^RecordName;
```

- *Pointer declaration*

```
Var
   PointerHead : PointerName; PointerCell :
   PointerName;
```

- *Example*

```
Type
   OneCell = Record
     Data1 : char;
     Data2 : Integer;
     Data3 : string;
     Next : ^onesel;
   End;
   Ptr = ^ OneCell;
Var
   Head, P : Ptr;
```

3.2. *Pointer – Pascal* (Lanj.)

- *Assignment*

```
PointerCell^.VarName := value;
PointerCell^.NextPointer^.VarName := value;
```

- *Accessing*

```
PointerCell^.varname
PointerCell^.NextPointer^.VarName
PointerCell^.NextPointer^.NextPointer^.VarName
```

- Inisialisasi/Membentuk *Pointer* Baru

```
new(PointerCell);
```

3.3. *Pascal Record Definition With Two Links*

- *Record definition*

```
Type
   RecordName = Record
     VarName : vartype;
     PreviousPtr : ^RecordName;
     NextPtr : ^RecordName;
   End;
   PointerName = ^RecordName;
```

- *Pointer declaration*

```
Var
   PointerHead : PointerName; PointerCell :
   PointerName;
```

- *Example*

```
Type
   OneCell = Record
     Data1 : char;
     Data2 : Integer;
     Data3 : string;
     Prev : ^onesel;
     Next : ^onesel;
   End;
   Ptr = ^ OneCell;
Var
   Head, P : Ptr;
```

3.4. *Pointer* - C

- *Record definition*

```
struct StructName
{
     vartype VarName;
     struct StructName *NextPtr;
};
```

- *Pointer declaration*

```
struct StructName *PtrHead
struct StructName *PtrCell;
```

- *Example*

```
struct OneCell
{
     char Data1;
     int  Data2;
     char Data3[50];
     struct OneCell Next;
};


Int main(void)
{
   struct OneCell *Head, *Ptr;
   .
   .
}
```

3.4. *Pointer* – C (Lanj.)

- *Assignment*

```
PtrCell->VarName = value;
PtrCell->NextPtr->VarName = value;
PtrCell->NextPtr = NULL;
```

3.4. *Pointer* – C (Lanj.)

- *Accessing*

```
PtrCell->VarName
(*PtrCell).VarName
PtrCell->NextPtr->VarName
(*(*PtrCell).NextPtr).VarName
PtrCell->NextPtr->NextPtr->VarName
(*(*(*PtrCell).NextPtr).NextPtr).VarName
```

3.4. *Pointer* – C (Lanj.)

- Inisialisasi/membentuk *Pointer* Baru

  Menggunakan malloc

```
PtrCell=(struct StructName
*)malloc(sizeof(struct StructName));
```

  Menggunakan new

```
PtrCell=new StructName;
```

3.4. *Pointer* – C (Lanj.)

Note:

- *Assign pointer address to a* PtrCell

```
PtrCell=VarName; //located at stack or global variable
```

- *Pointer attribute*

```
Pointer->Variable //Field may writen as (*Pointer).VariableField
```

- *Statement* `new` *define at* new.h *header file*
- new.h *header file can be used only at* C++

3.5. *C Structure definition* Dengan Dua Links

- *Record definition*

```
struct StructName
{
  vartype VarName;
  struct StructName *PrevPtr;
  struct StructName *NextPtr;
};
```

- *Example*

```
struct OneCell
{
     char Data1;
     int  Data2;
     char Data3[50];
     struct OneCell Prev;
     struct OneCell Next;
};


Int main(void)
{
   struct OneCell *Head, *Ptr;
   .
   .
}
```

3.6. *Record* Dengan Satu Link

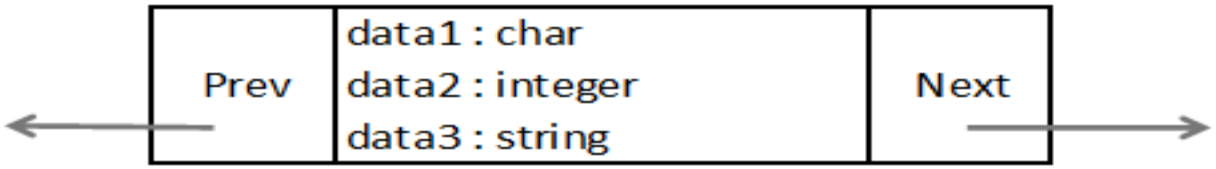
```
struct OneCell
{
    char Data1;
    int  Data2;
    char Data3[50];
    struct OneCell Next;
};
```

```
Type
    OneCell = Record
          Data1 : char;
          Data2 : Integer;
          Data3 : string;
          Next : ^onesel;
    End;
    Ptr = ^ OneCell;
```

3.7. *Record* Dengan Dua Link

```
struct OneCell
{
    char Data1;
    int  Data2;
    char Data3[50];
    struct OneCell Prev;
    struct OneCell Next;
};
```

```
Type
    OneCell = Record
          Data1 : char;
          Data2 : Integer;
          Data3 : string;
          Prev : ^onesel;
          Next : ^onesel;
    End;
    Ptr = ^ OneCell;
```



Ringkasan

- *Pointer* adalah variabel yang berisi alamat dari suatu *memory* yang menyimpan data
- Variabel *Pointer* dapat dideklarasikan dengan mencantumkan karakter * di depan variabel yang dideklarasikan
- Alokasi *memory* dimaksudkan untuk mencari dan memesan suatu alamat *memory* dengan sel serialnya untuk didaftarkan menyimpan data dengan tipe data yang dimaksudkan untuk ditunjuk oleh suatu variabel pointer
- Penggunaan tipe data *pointer* dimaksudkan tidak untuk tujuan menghemat *memory*, karena selain mendeklarasikan variabel bertipe *pointer*, maka program juga perlu mengalokasikan alamat pada *memory* yang mampu menampung tipe data dari data yang sebenarnya hendak disimpan

Contoh Mengakses Memory Dengan Pointer

```
#include <stdio.h>
#include <conio.h>
#include <iostream.h>
void main()
{
  int a, *b, c;
  a = 10;
  b = &a;
  c = *b;
  cout<<"isi variabel a:"<<a<<endl;
  cout<<"isi variabel b (&a):"<<b<<endl;
  cout<<"isi variabel c (*b):"<<c<<endl;
  getch();
}
```

Terimakasih

TUHAN Memberkati Anda

Teady Matius Surya Mulyana (tmulyana@bundamulia.ac.id)