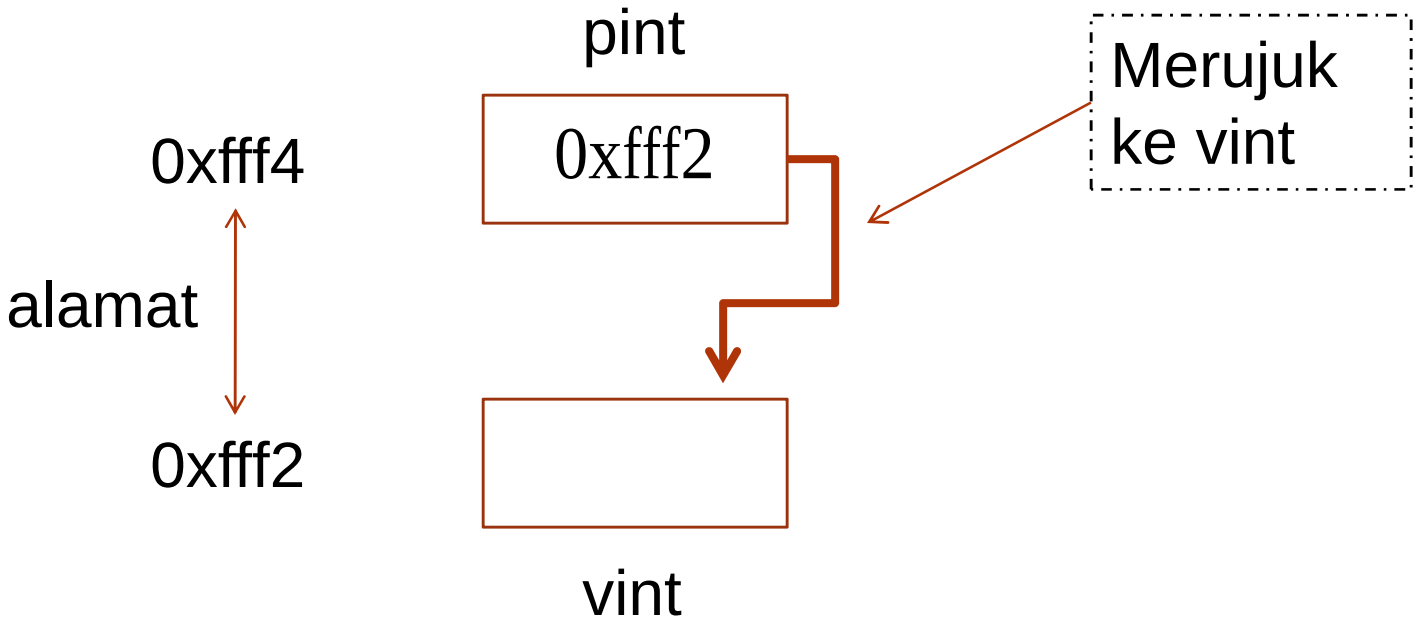
8. Pointer

S. Indriani L, M.T

Alamat dan Pointer

- Pointer sesungguhnya berisi alamat dari suatu data, bukan data sebagaimana pada variabel yang telah Anda kenal.
- Setiap byte didalam memori komputer memiliki sebuah alamat. Didalam memori inilah variabel disimpan.
- Pada saat program dimuat didalam memori, variabel akan diletakkan dengan sendirinya pada alamat tertentu.
- Suatu pointer didalam program dimaksudkan untuk menunjuk ke suatu alamat memori.

- Misal:
  pint adalah pointer
  vint adalah variabel yang terletak didalam memori beralamatkan `0xfff2`.
  maka pointer *pint* dapat diatur agar menunjuk ke variabel *vint*.



Mengetahui alamat suatu variabel

- Caranya:

  tambahkan operator alamat, berupa simbol ‘&’ didepan nama variabel, lalu dengan mengirimkan ke cout, alamat dari suatu variabel akan ditampilkan ke layar

```
#include<iostream.h>
#include<conio.h>

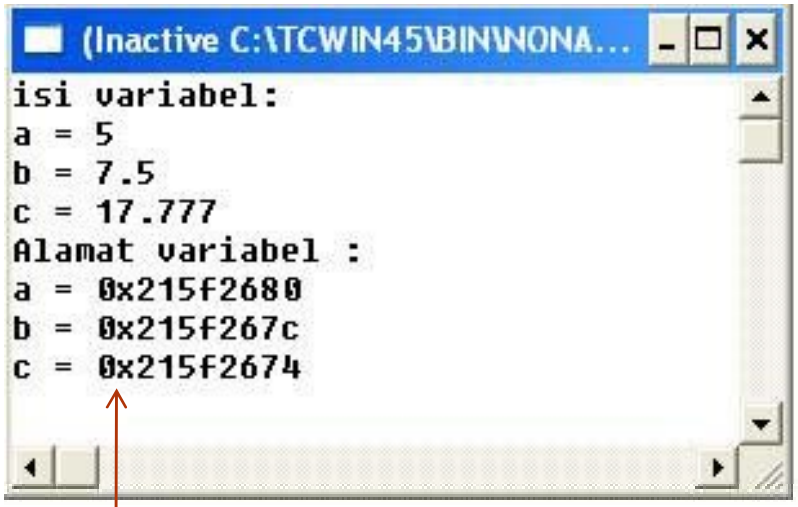
void main()
{
  int a=5;
  float b=7.5;
  double c=17.777;

  clrscr();

  cout<<"isi variabel:"<<endl;
  cout<<"a = "<<a<<endl;
  cout<<"b = "<<b<<endl;
  cout<<"c = "<<c<<endl;

  cout<<"Alamat variabel : "<<endl;
  cout<<"a = "<<&a<<endl;
  cout<<"b = "<<&b<<endl;
  cout<<"c = "<<&c<<endl;
}
```

Hasil:



Perlu diketahui:
Notasi `0x` pada hasil program diatas menyatakan notasi heksadesimal

Mendefinisikan variabel pointer

- Suatu variabel pointer dapat difenisikan dengan bentuk sbb:

```
tipe_data *nama_variabel;
```

keterangan:

- *tipe_data* dapat berupa sembarang tipe seperti halnya pada pendefinisian variabel bukan pointer.
- *nama_variabel* adalah nama variabel pointer.

- Beberapa contoh pendefinisian variabel pointer:

```
int *pint        //pointer ke int
char *pch        //pointer ke char
float *pfl       //pointer ke float
```

Mengisi variabel pointer

- Supaya suatu variabel pointer menunjuk ke variabel lain, mula-mula harus diisi dengan alamat dari variabel yang hendak ditunjuk.
- Misal:

```
int vint = 55;
int *pint;
```

- Variabel pointer pint dapat diatur agar menunjuk ke vint dengan cara sbb:

```
pint = &vint;
```

- Pernyataan diatas berarti: "`pint` diisi dengan alamat dari `vint`".
- Operator & menyatakan "alamat dari"

```
#include<iostream.h>
#include<conio.h>

void main()
{
  int vint=55;
  int *pint;

  clrscr();

  pint = &vint;

  cout<<"alamat vint ="<<&vint<<endl;
  cout<<"pint = "<<pint<<endl;
}
```



Ternyata menampilkan isi dari pointer itu sendiri, bukan isi dari variabel vint

Mengakses nilai yang ditunjuk pointer

- Berdasarkan program sebelumnya, nilai vint dapat diakses melalui pint setelah pernyataan:

  ```
  pint = &vint;
  ```

  Caranya dengan melibatkan operator “tak langsung”. Operator ini berupa simbol * dan diletakkan didepan nama variabel pointer.

- Contoh:

  ```
  *pint
  ```

  berarti “nilai yang ditunjuk oleh pint”

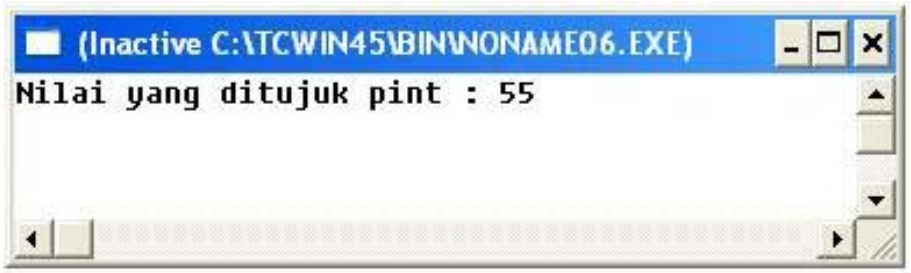
```
#include<iostream.h>
#include<conio.h>

void main()
{
  int vint=55;
  int *pint;

  clrscr();

  pint = &vint;

  cout<<"Nilai yang ditujuk pint : "<<*pint<<endl;
}
```



Pointer dan Array

- Pointer dan Array mempunyai hubungan yang dekat. Secara internal array juga menyatakan alamat.
- Misalnya:

```
int tgl_lahir[]={26, 3, 1979};
```

dan

```
int *ptgl;
```

agar `ptgl` menunjuk ke array, diperlukan pernyataan berupa:

```
ptgl = tgl_lahir;
```

Perhatikan dengan seksama skema tersebut!

- Tidak ada tanda ‘&’ didepan `tgl_lahir`.
- Ini disebabkan nama array sebenarnya sudah menyatakan alamat. Oleh karena itu tanda & tidak diperlukan.

```
#include<iostream.h>
#include<conio.h>

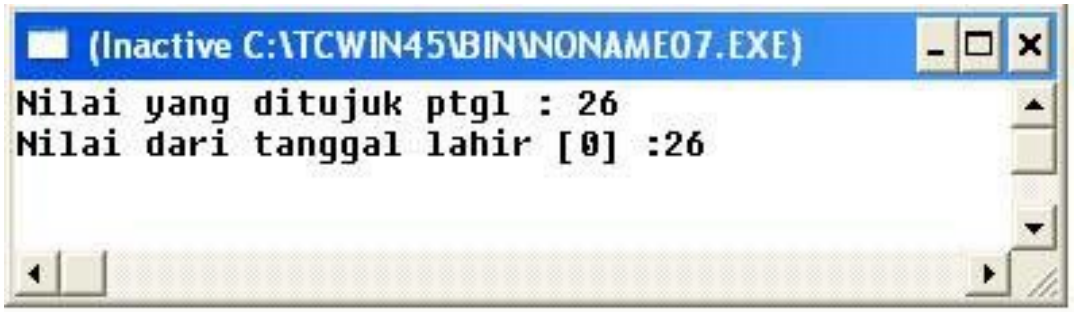
void main()
{
  clrscr();

  int tgl_lahir[] = {26, 3, 1979};
  int *ptgl;

  ptgl = tgl_lahir;

  cout<<"Nilai yang ditujuk ptgl : "<<*ptgl<<endl;

  cout<<"Nilai dari tanggal lahir [0] :"<<tgl_lahir [0]<<endl;
}
```



- Seluruh elemen array `tgl_lahir` dapat ditampilkan melalui pernyataan:

```
for (int i=0; i<3; i++)
    cout<<tgl_lahir[i]<<endl;
```

bentuk seperti tgl_lahir[i] dapat diubah dalam notasi pointer seperti berikut:

```
for(int i=0; i<3; i++)
    cout <<*(tgl_lahir + i)<<endl;
```

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();

  int tgl_lahir[] = {26, 3, 1979};
  int *ptgl;

  ptgl = tgl_lahir;

  for( int i=0; i<3; i++)
       cout <<*(ptgl +i)<<endl;
}
```



Pointer dan String

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();

  char tokoh[]="Barrack_Obama";
  char *ptokoh ="Barrack_Obama";

  cout<<"tokoh ="<<tokoh<<endl;
  cout<<"ptokoh = "<<ptokoh<<endl;

  ptokoh++; //tokoh++; tidak diperkenankan

  cout<<"ptokoh = "<<ptokoh<<endl;
}
```



Tampak dengan pointer dengan mudah dapat diubah (misalnya ptokoh). Dengan cara seperti ini, kalau semula ptokoh menunjukkan "Barrack_Obama", kini menunjukkan "arrack Obama"

Array Pointer

- Suatu array bisa digunakan untuk menyimpan sejumlah pointer.
- Contoh:

  ```
  char *namahari[7];
  ```

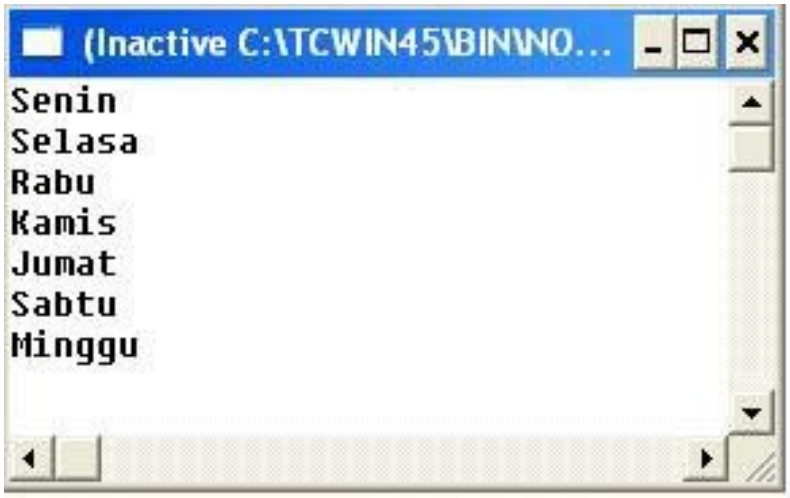
  merupakan pernyataan untuk mendefinisikan array `namahari` yang berisi 7 buah elemen yang berupa pointer. Pointer ini menunjuk ke data bertipe `char`.

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();

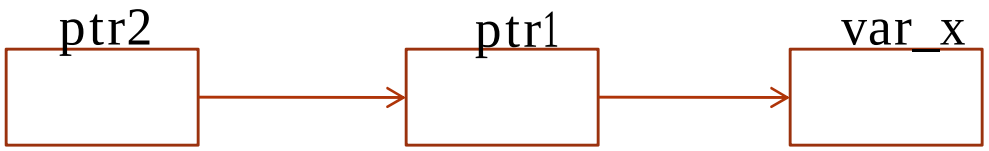
  char *namahari[]= {"Senin", "Selasa", "Rabu", "Kamis",
  "Jumat", "Sabtu", "Minggu"};

  for (int i=0; i<7; i++)
       cout<<namahari[i]<<endl;
}
```



Pointer menunjuk Pointer

- Suatu pointer bisa saja menunjuk ke pointer lain.



- Untuk membentuk rantai pointer seperti diatas diperlukan pendefinisian sbb:

```
int var_x;
int *ptr1;        // satu tanda * di depan ptr1
int **ptr 2       // dua tanda * didepan ptr2
```

- Agar ptr1 menunjuk ke variabel var_x, perintah yang diperlukan berupa:

```
ptr1= & var_x;
```

- Sedangkan supaya ptr2 menunjuk ke ptr1, diperlukan perintah berupa:

```
ptr2= &ptr1;
```

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();

  int var_x=273;
  int *ptr1;
  int **ptr2;

  cout<<"var x = "<<var_x<<endl;

  ptr1=&var_x;
  ptr2=&ptr1;

  cout<<"*ptr1 ="<<*ptr1<<endl;

  cout<<"**ptr2 = "<<**ptr2<<endl;
}
```

