REPRESENTASI FISIK
LIST LINEAR

Danang Wahyu Utomo
danang.wu@dsn.dinus.ac.id
+6285 740 955 623

RENCANA KEGIATAN PERKULIAHAN SEMESTER

W	Pokok Bahasan
1	ADT Stack
2	ADT Queue
3	List Linear
4	List Linear
5	List Linear
6	**Representasi Fisik List Linear**
7	Variasi List Linear
8	**Ujian Tengah Semester**

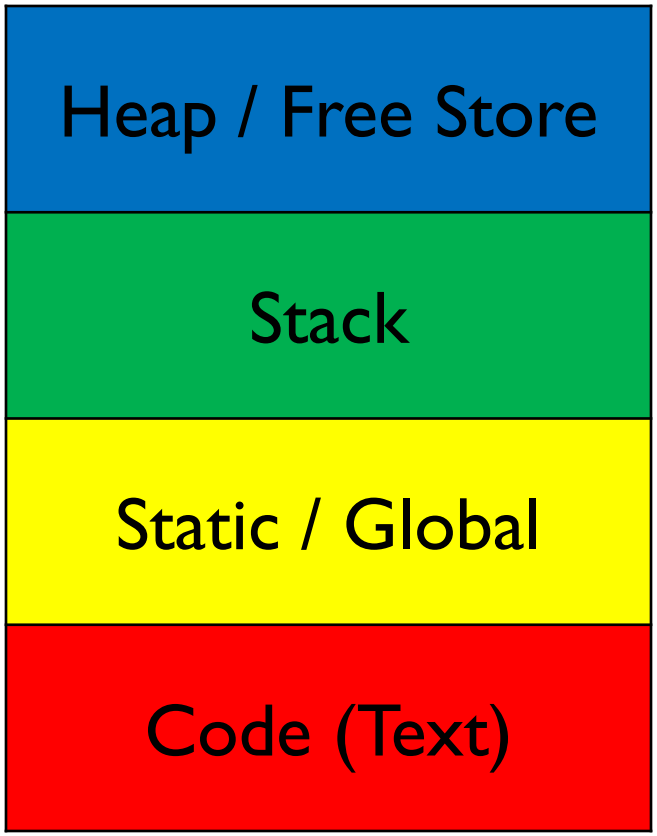
W	Pokok Bahasan
9	Variasi List Linear
10	Variasi List Linear
11	Stack dengan Representasi List
12	Queue dengan Representasi List
13	List Rekursif
14	Pohon dan Pohon Biner
15	Multi List
16	**Ujian Akhir Semester**

Content

- Application Memory
- Representasi Berkait dengan Pointer

Application Memory

- Memori yang dialokasikan pada sebuah program/aplikasi umumnya dibagi menjadi 4 bagian :



Memory yang dapat di request selama program berjalan

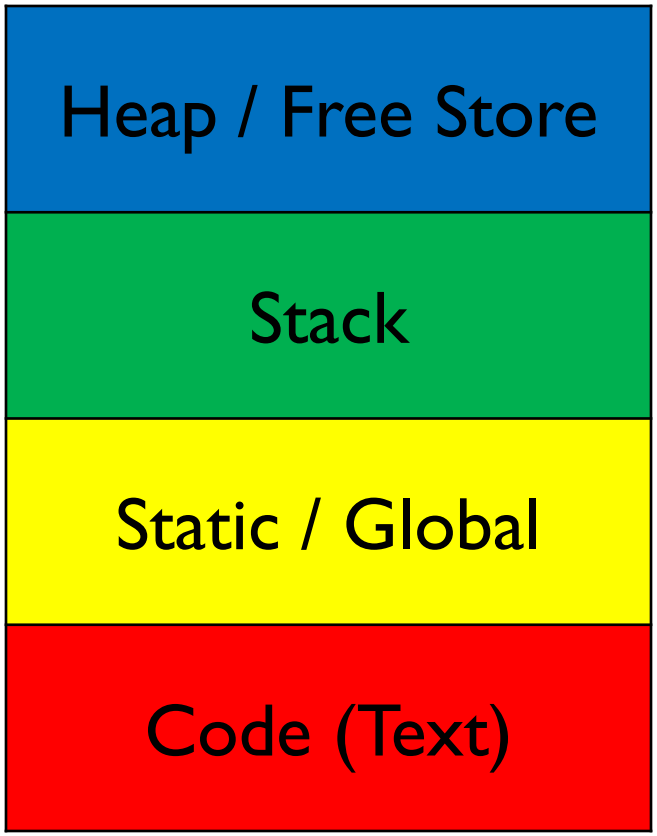
Menyimpan semua informasi tentang Pemanggilan fungsi untuk menyimpan semua Variabel lokal

Menyimpan variable global selama program berjalan

Menyimpan semua perintah yang akan Dijalankan / dieksekusi

Application Memory

- Memori yang dialokasikan pada sebuah program/aplikasi umumnya dibagi menjadi 4 bagian :

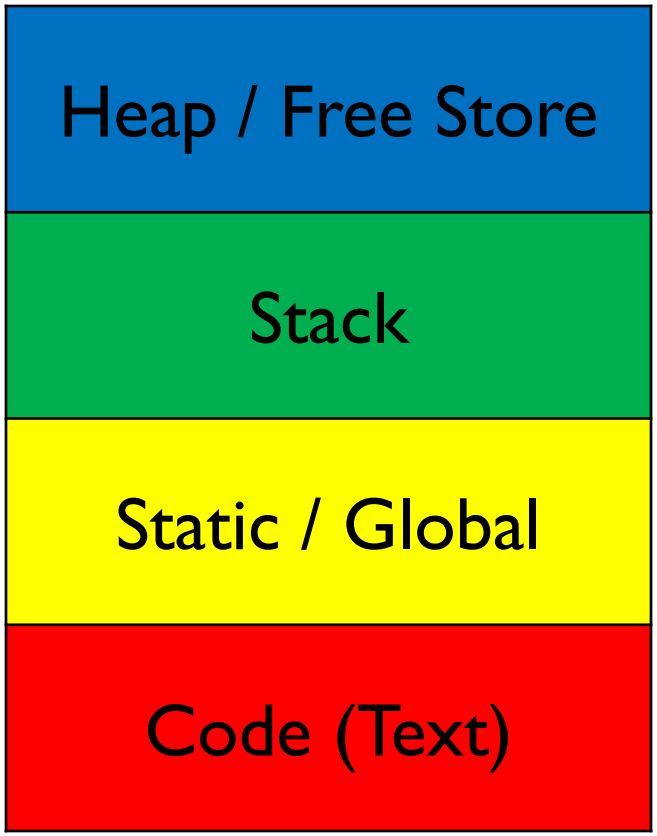


Bersifat dinamis : ukukran memory dapat Berubah ketika program dijalankan

Bersifat statis :
- Ukuran memori ditentukan ketika kompilasi
- ukuran memori tidak dapat berubah

Application Memory

- Memori yang dialokasikan pada sebuah program/aplikasi umumnya dibagi menjadi 4 bagian :



Contoh :
Penggunaan malloc pada bahasa C untuk Mengalokasikan sejumlah memory ketika Program dijalankan

Representasi Berkait dengan Pointer

```
struct node {
   int data;
   struct node * next;
};

typedef struct node List;
List* head;
```

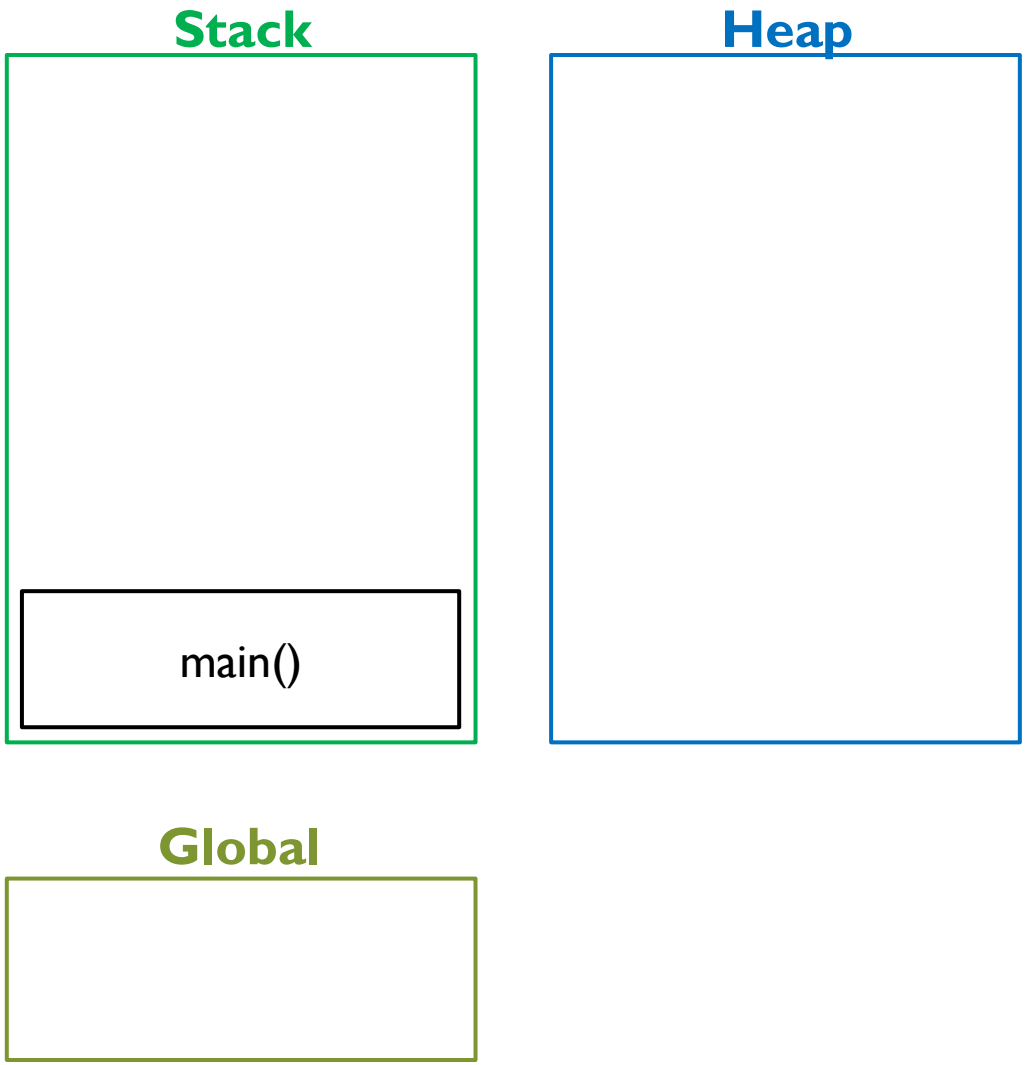
Representasi Berkait dengan Pointer

```
void insertN(int data,int p){    //fungsi insert Node
    List* curr1 = (List*)malloc(sizeof(List));
    curr1->data = data;
    curr1->next = NULL;
    if(p==1){      //jika menambahkan Node di awal
        curr1->next=head;
        head=curr1;
        return;
    }
    List* curr2 = head;
    int j;
    for(j=0;j<p-2;j++){
        curr2 = curr2->next;
    }
    curr1->next=curr2->next;
    curr2->next=curr1;
}
```

Representasi Berkait dengan Pointer

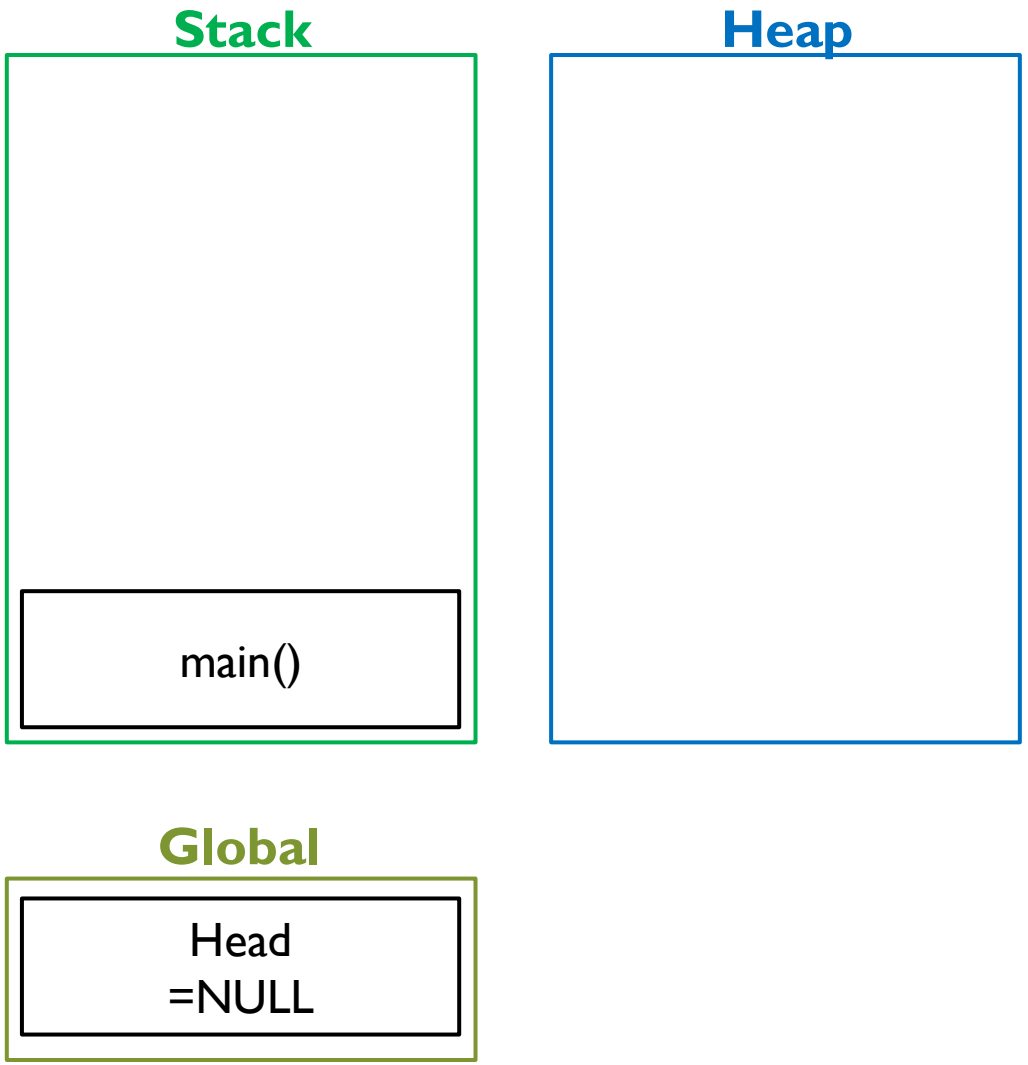
```
int main (){


}
```



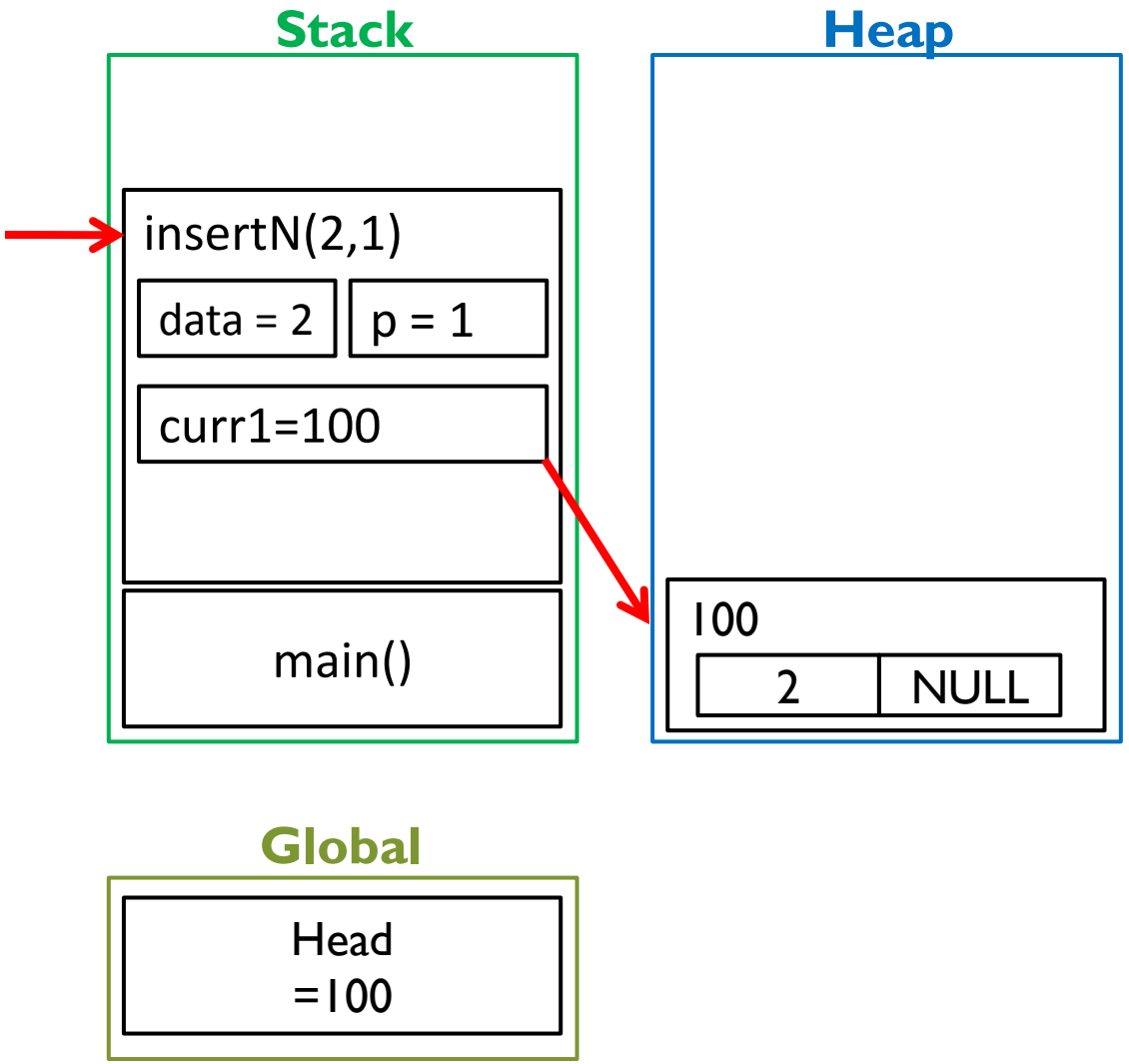
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
}
```



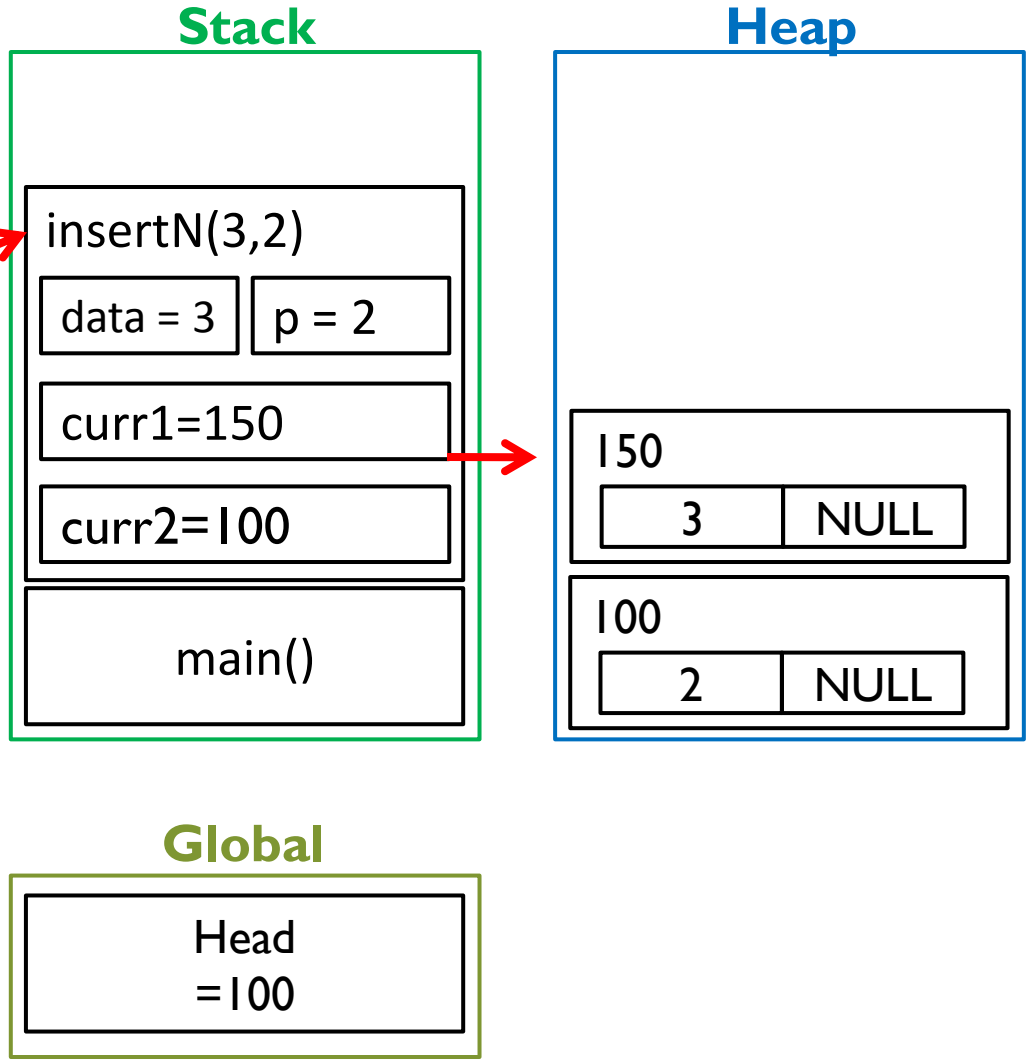
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
}
```



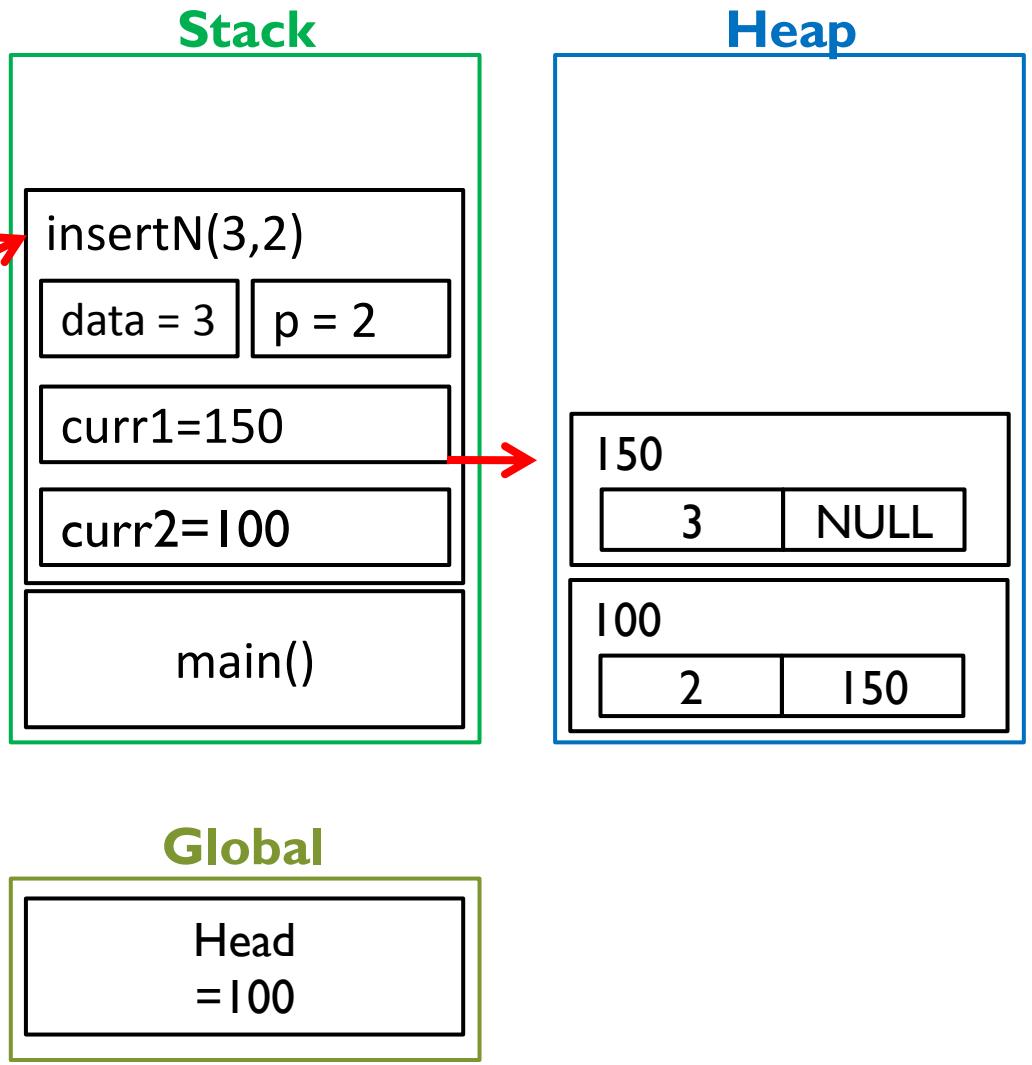
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
}
```



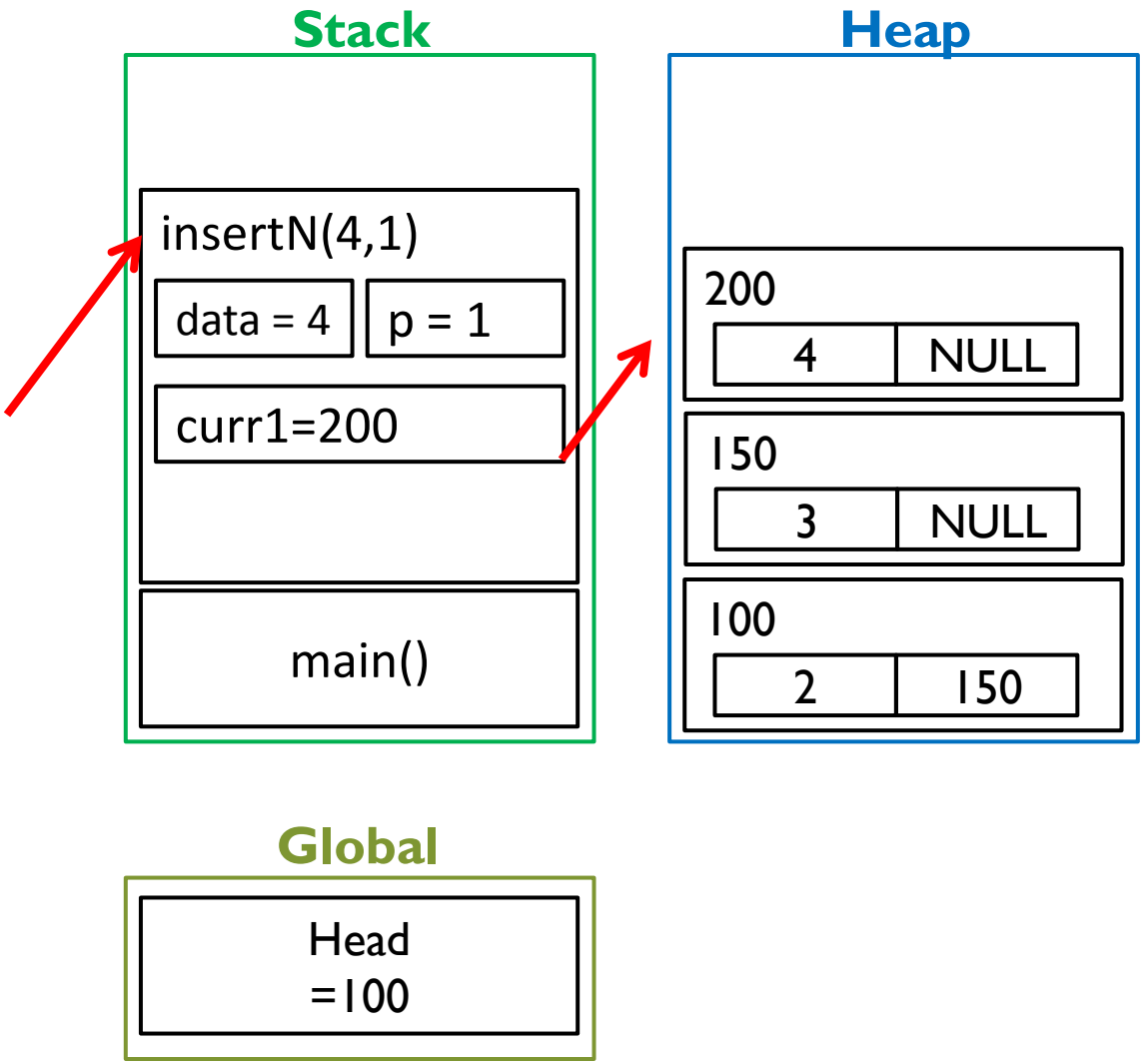
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
}
```



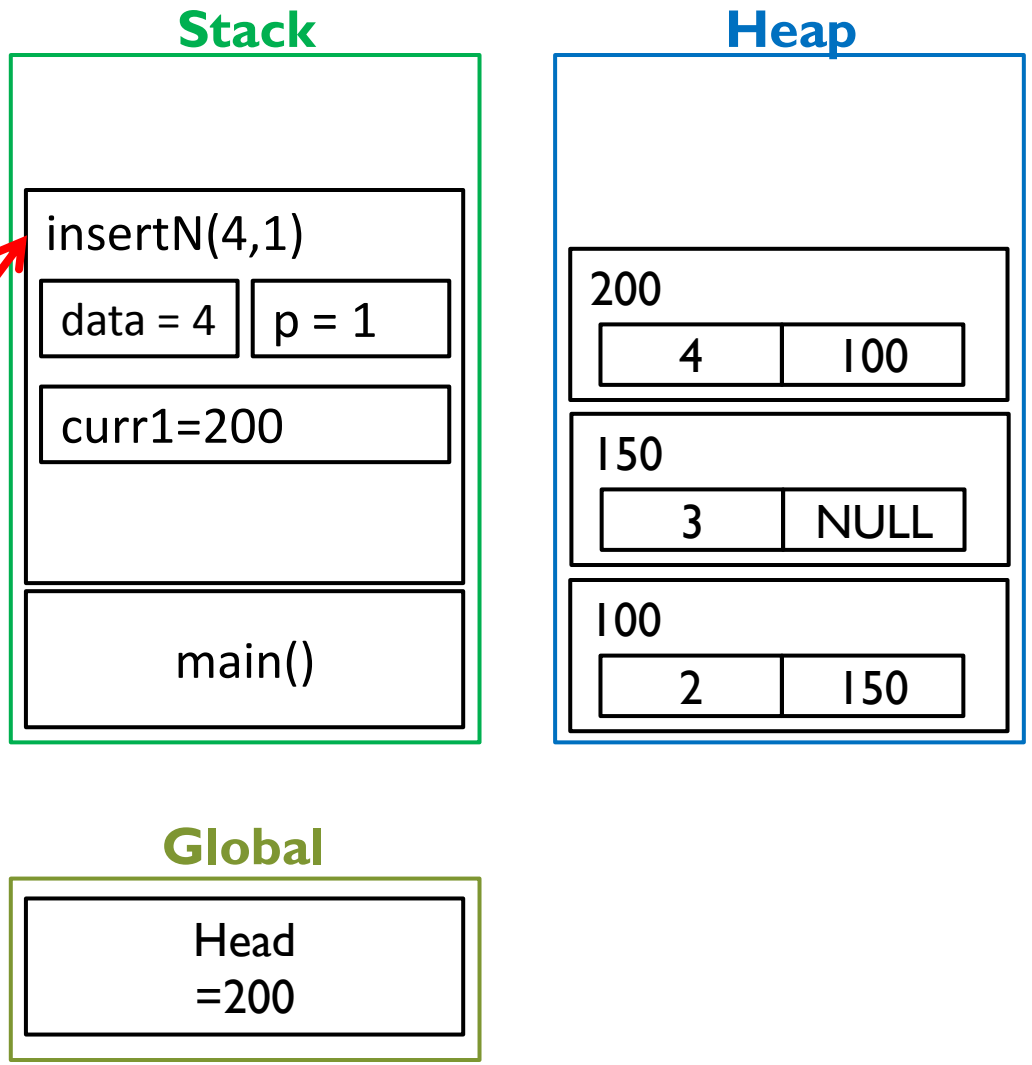
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
  insertN(4,1);
}
```



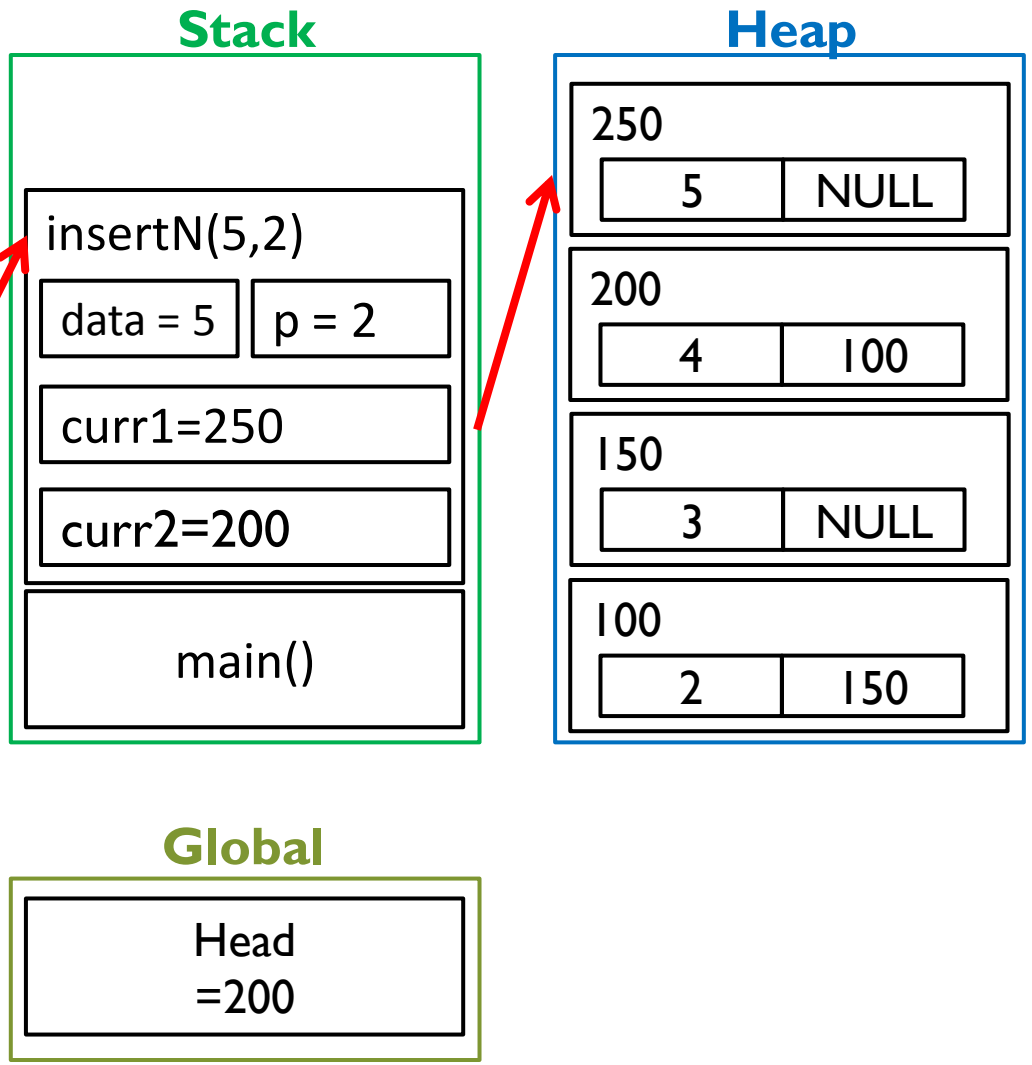
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
  insertN(4,1);
}
```



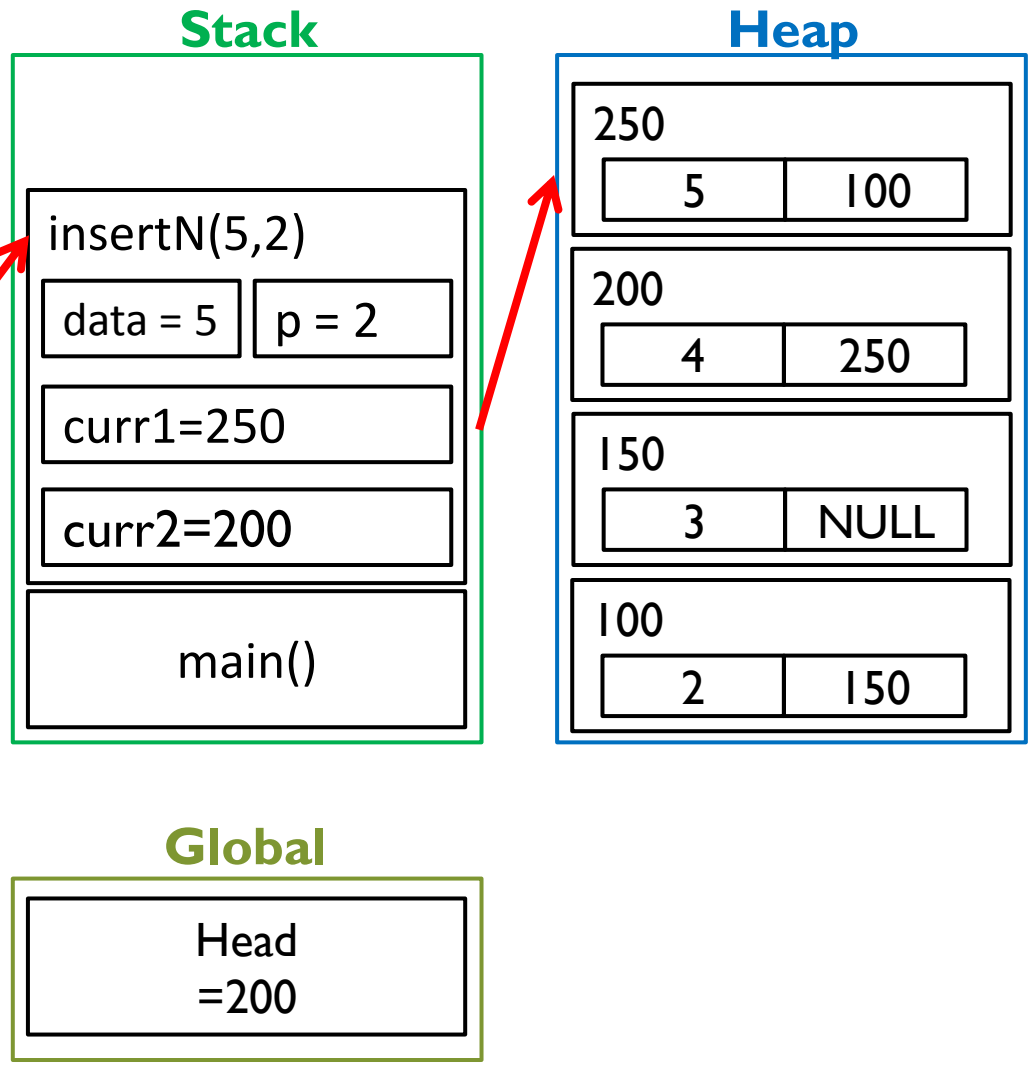
Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
  insertN(4,1);
  insertN(5,2);
}
```



Representasi Berkait dengan Pointer

```
int main (){
  head = NULL;
  insertN(2,1);
  insertN(3,2);
  insertN(4,1);
  insertN(5,2);
}
```



```
int main() {

    head = NULL;

    insertN(2,1);
    insertN(3,2);
    insertN(4,1);
    insertN(5,2);

}
```