STRUKTUR DATA

LIST GANDA REPRESENTASI DINAMIS

Rosa Ariani Sukamto

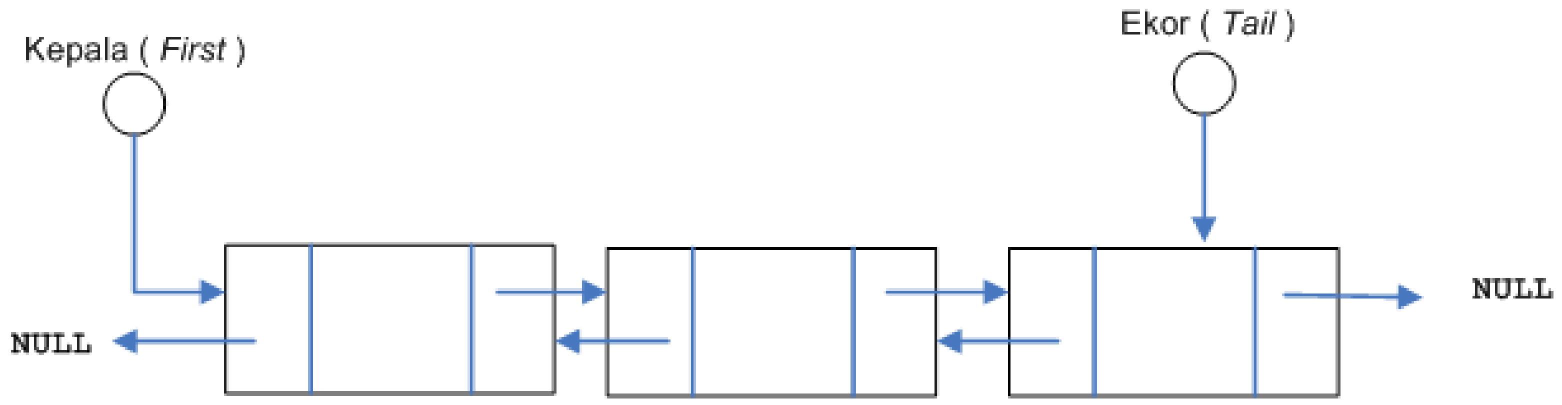
ROSA ARIANI SUKAMTO

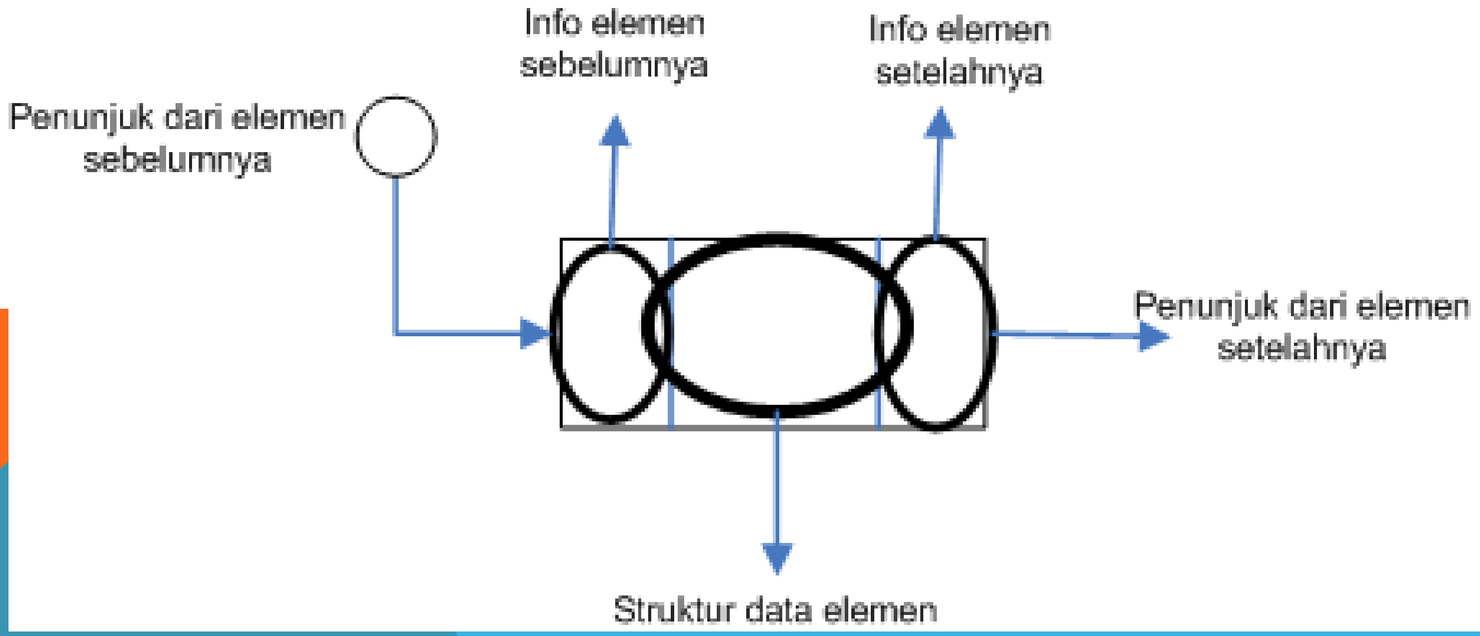
Blog: http://hariiniadalahhadiah.wordpress.com

Facebook: https://www.facebook.com/rosa.ariani.sukamto

Email: rosa_if_itb_01@yahoo.com

LIST GANDA


Kepala (*First*)
Ekor (*Tail*)
NULL
NULL


Info elemen sebelumnya
Info elemen setelahnya
Penunjuk dari elemen sebelumnya
Penunjuk dari elemen setelahnya
Struktur data elemen

DEKLARASI ELEMEN

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>

typedef struct{
  char  nim[10];
  char nama[50];
  char nilai[2];
}nilaiMatKul;

typedef struct elmt *alamatelmt;
typedef struct elmt{
  nilaiMatKul elmt;
  alamatelmt prev;
  alamatelmt next;
} elemen;

typedef struct{
  elemen *first;
  elemen *tail;
}list;
```

CREATE LIST

```
void createList(list *L){

   (*L).first = NULL;
   (*L).tail = NULL;

}
```

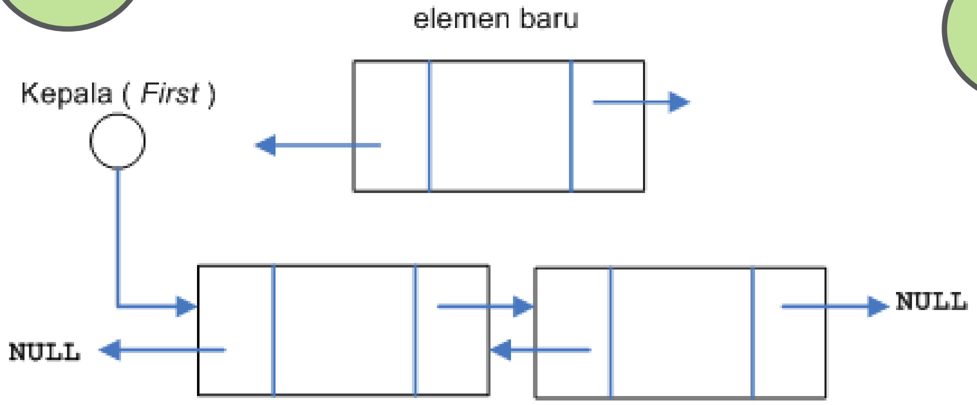
COUNT ELEMENT

```
int countElement(list L){
   int hasil = 0;

  if(L.first !=NULL){
    /*list tidak kosong*/

    elemen *elmt;

    /*inisialisasi*/
    elmt = L.first;
```

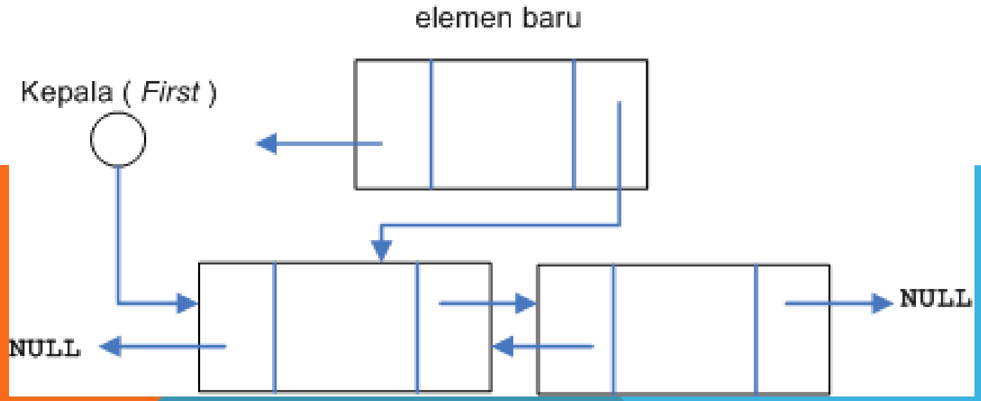
```
    while(elmt != NULL){
       /*proses*/
       hasil = hasil + 1;

       /*iterasi*/
       elmt = elmt->next;
     }

   }

   return hasil;
 }
```

PENAMBAHAN ELEMEN DI AWAL LIST (ADDFIRST)

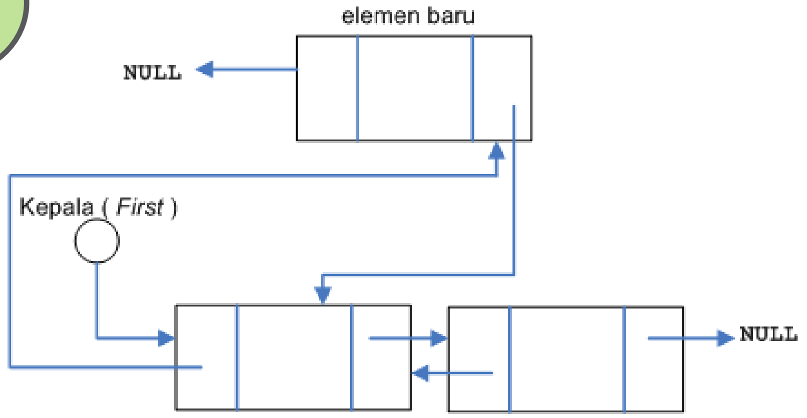
1



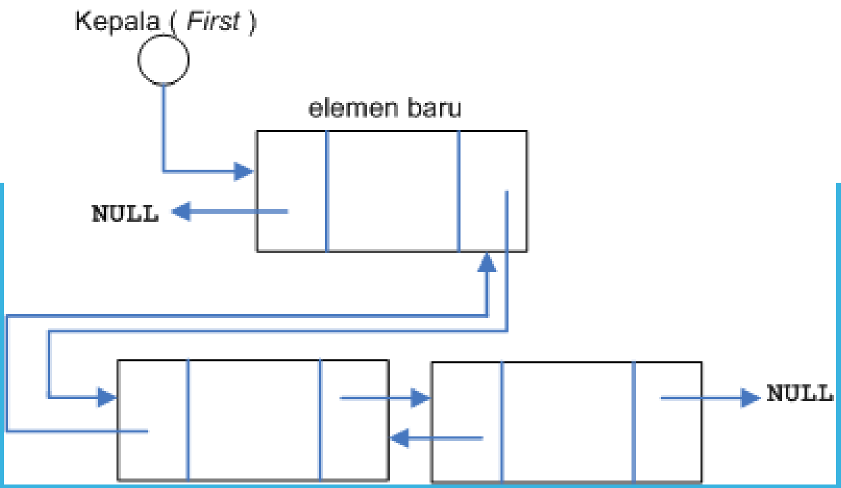
2



3



4

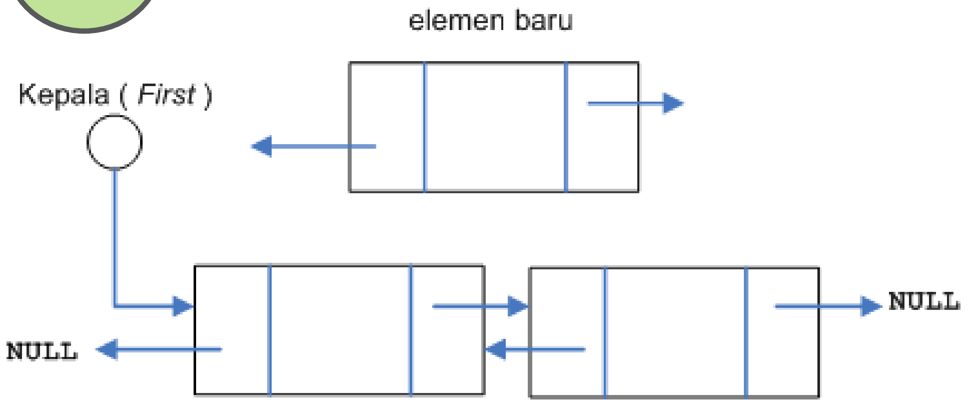


ADD FIRST

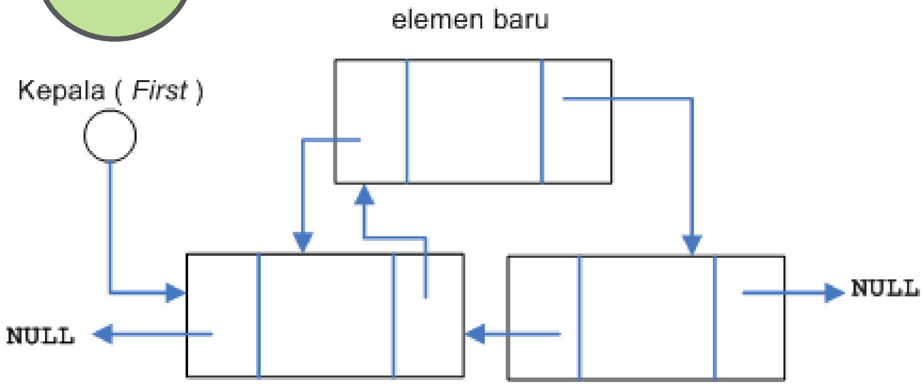
```
void addFirst(char nim[], char nama[], char nilai[], list *L){
  elemen *baru;
  baru = (elemen *) malloc (sizeof (elemen));
  strcpy(baru->elmt.nim, nim);
  strcpy(baru->elmt.nama, nama);
  strcpy(baru->elmt.nilai, nilai);
  if((*L).first == NULL){
        baru->prev = NULL;
        baru->next = NULL;
        (*L).tail = baru;
  }else{
        baru->next = (*L).first;
        baru->prev = NULL;
        (*L).first->prev = baru;
  }
  (*L).first = baru;
  baru = NULL;
}
```

PENAMBAHAN ELEMEN DI TENGAH (ADDAFTER)

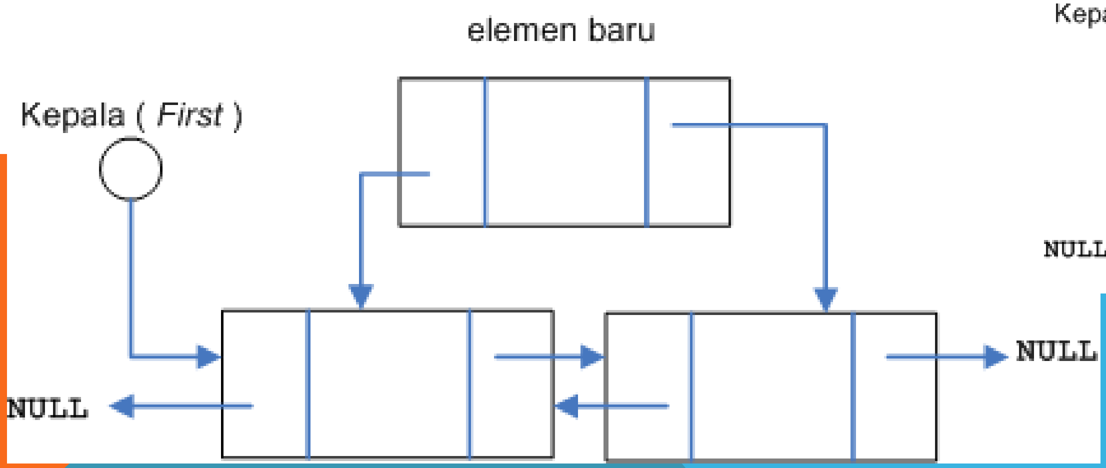
1



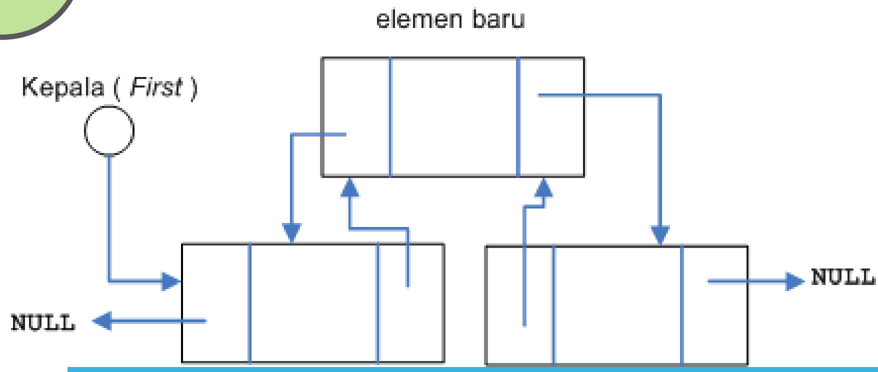
3



2



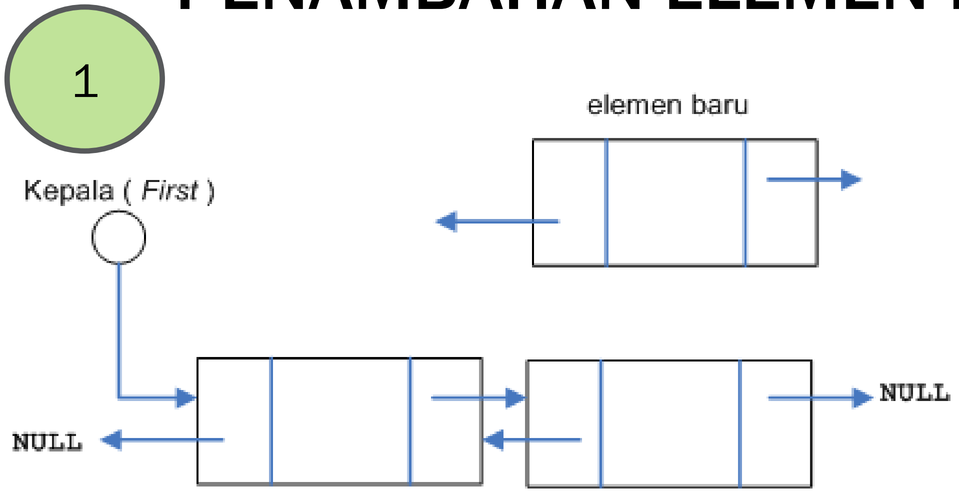
4

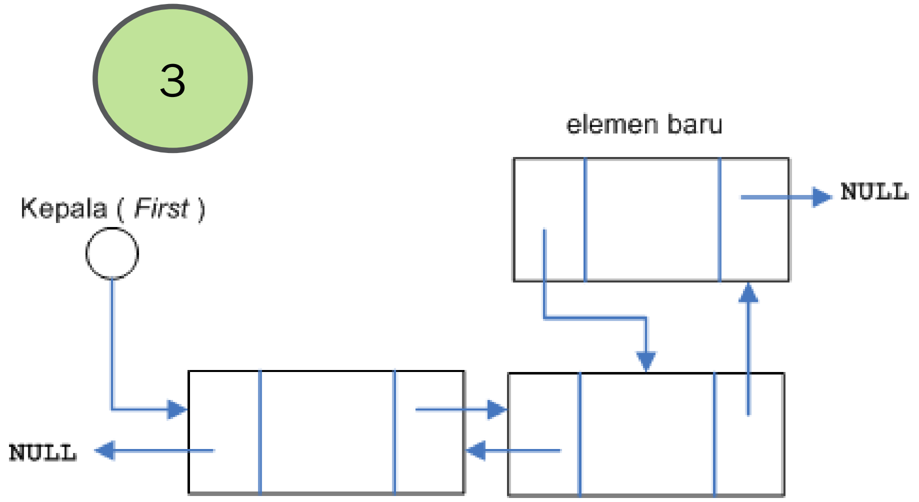


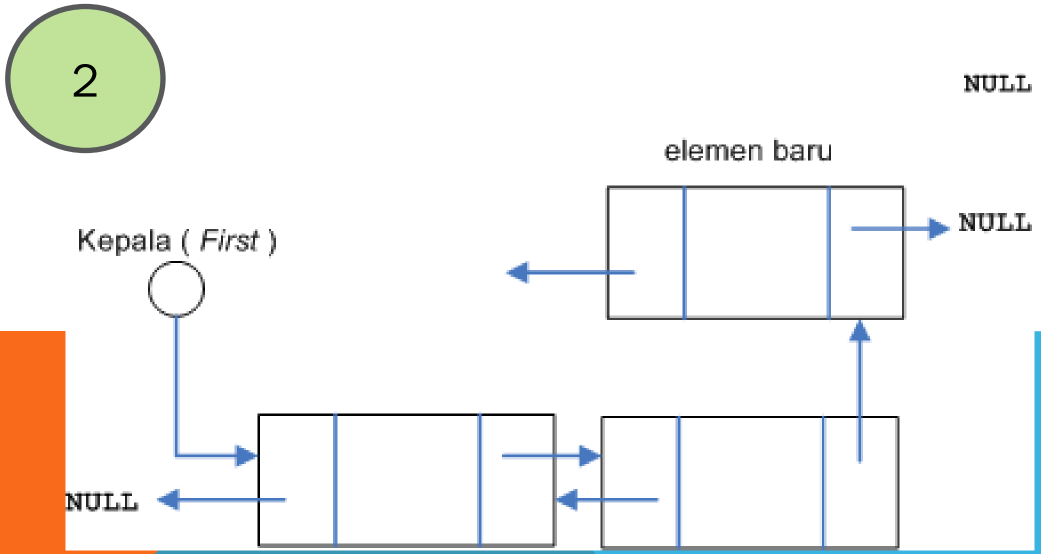
ADD AFTER

```
void addAfter(elemen *prev, char nim[], char nama[], char
   nilai[], list *L){

  elemen *baru;
  baru = (elemen *) malloc (sizeof (elemen));
  strcpy(baru->elmt.nim, nim);
  strcpy(baru->elmt.nama, nama);
  strcpy(baru->elmt.nilai, nilai);
  if(prev->next == NULL){
        baru->next = NULL;
        (*L).tail = baru;
  }else{
        baru->next = prev->next;
        baru->next->prev = baru;
  }
  baru->prev = prev;
  prev->next = baru;
  baru = NULL;
}
```

PENAMBAHAN ELEMEN DI AKHIR (ADDLAST)





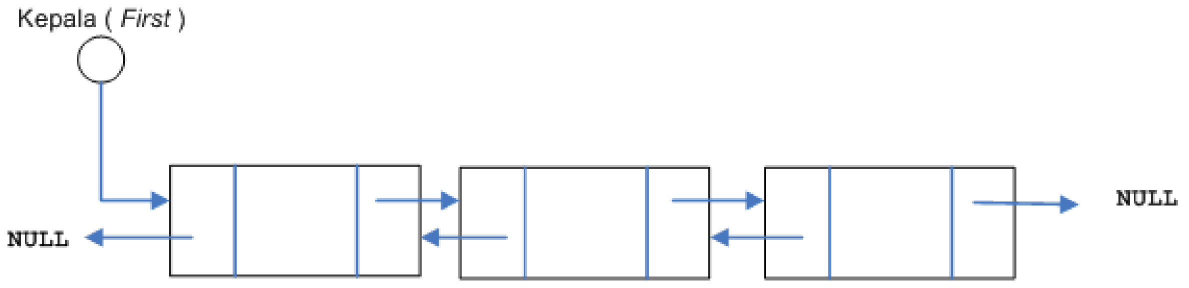


ADD LAST

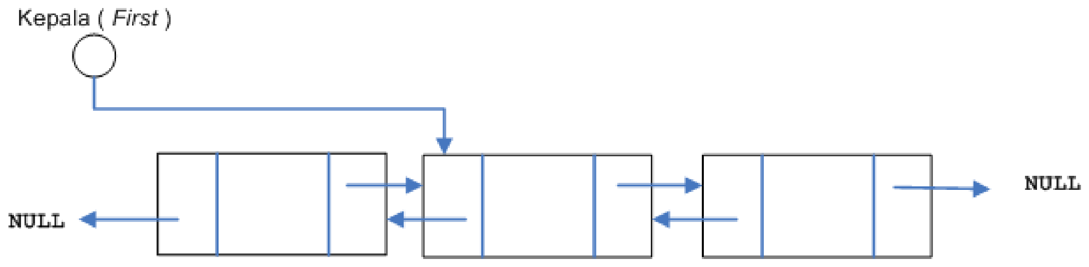
```
void addLast(char nim[], char nama[],
    char nilai[], list *L){

  if((*L).first == NULL){
    /*jika list adalah list kosong*/
    addFirst(nim, nama, nilai, L);
  }
  else{
    /*jika list tidak kosong*/
    elemen *baru;
    baru = (elemen *) malloc (sizeof
   (elemen));
    strcpy(baru->elmt.nim, nim);
    strcpy(baru->elmt.nama, nama);
    strcpy(baru->elmt.nilai, nilai);
    baru->next = NULL;
    (*L).tail->next = baru;
    baru->prev = (*L).tail;
    (*L).tail = baru;
    baru = NULL;
  }

}
```

HAPUS ELEMEN AWAL (DELFIRST)

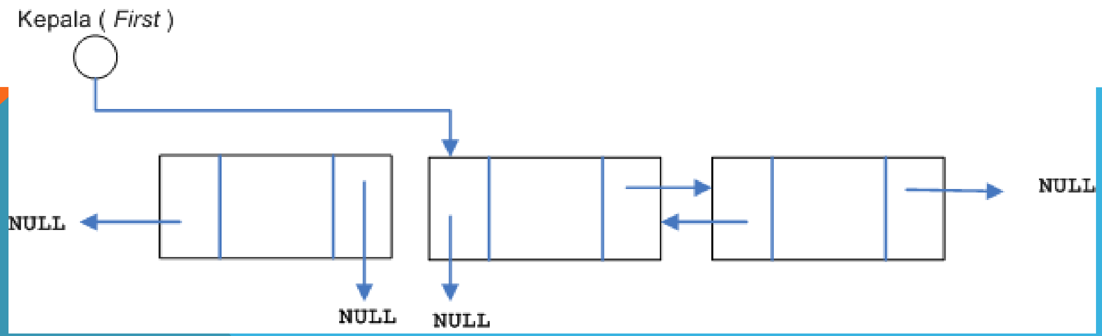
1



2



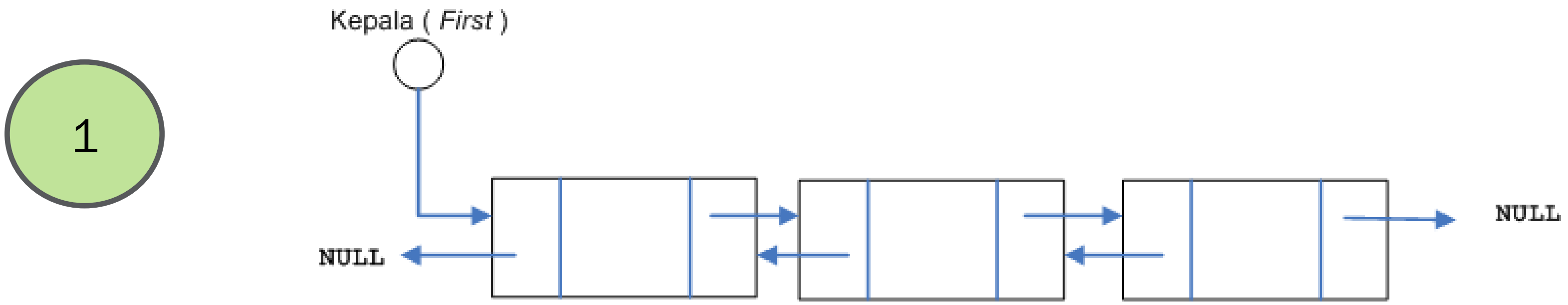
3

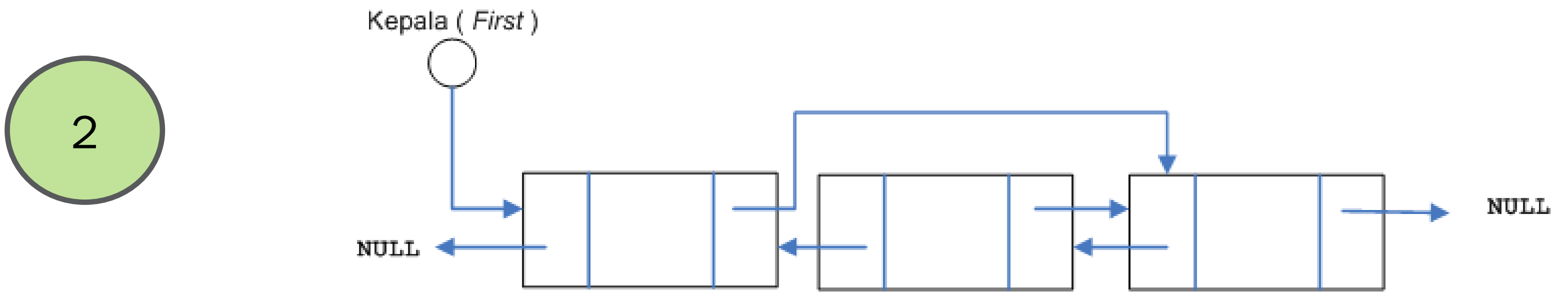


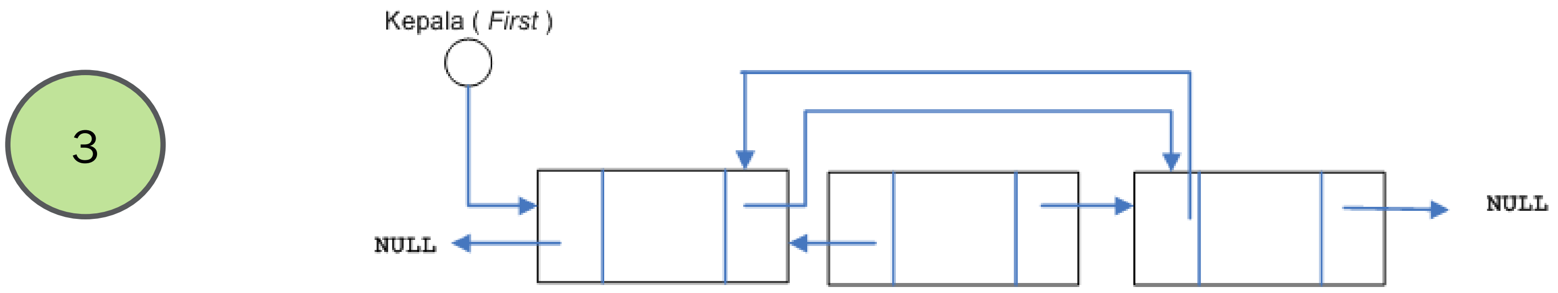
DEL FIRST

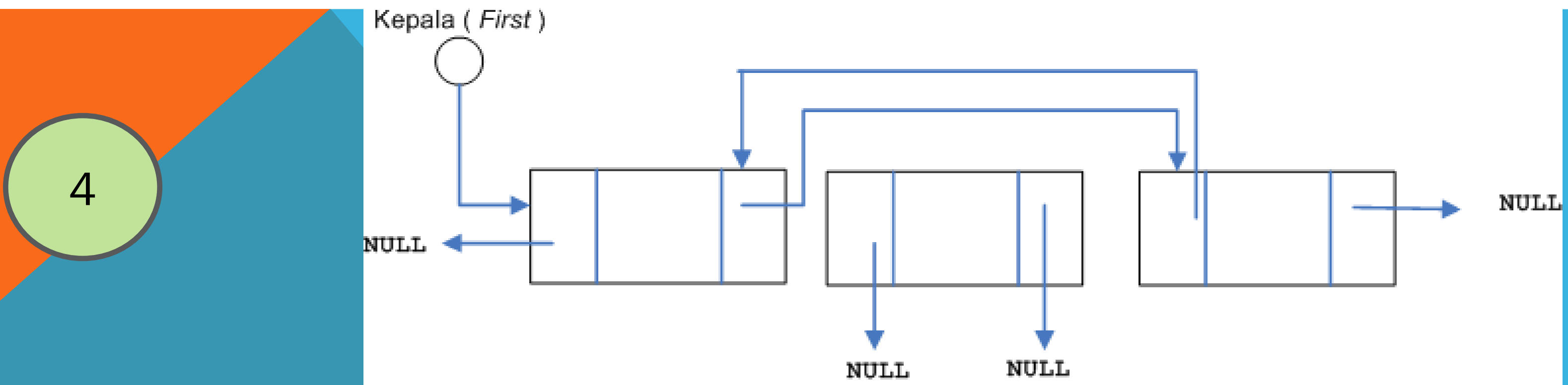
```
void delFirst(list *L){


  if((*L).first != NULL){
    /*jika list bukan list kosong*/
    elemen *hapus = (*L).first;
    if(countElement(*L) == 1){
        (*L).first = NULL;
        (*L).tail = NULL;
    }else{
        (*L).first = (*L).first->next;
         (*L).first->prev = NULL;
        hapus->next = NULL;
    }
    free(hapus);
  }
}
```

HAPUS ELEMEN TENGAH (DELAFTER)







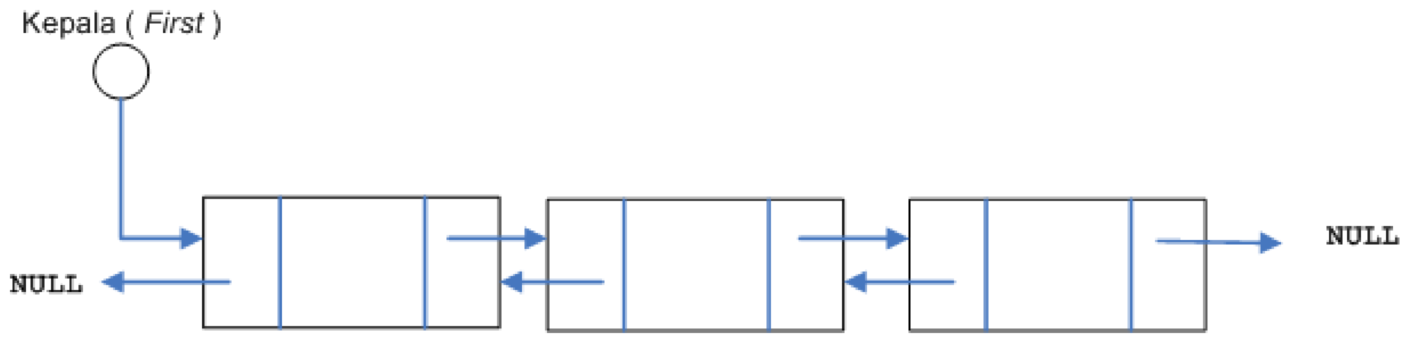


DEL AFTER

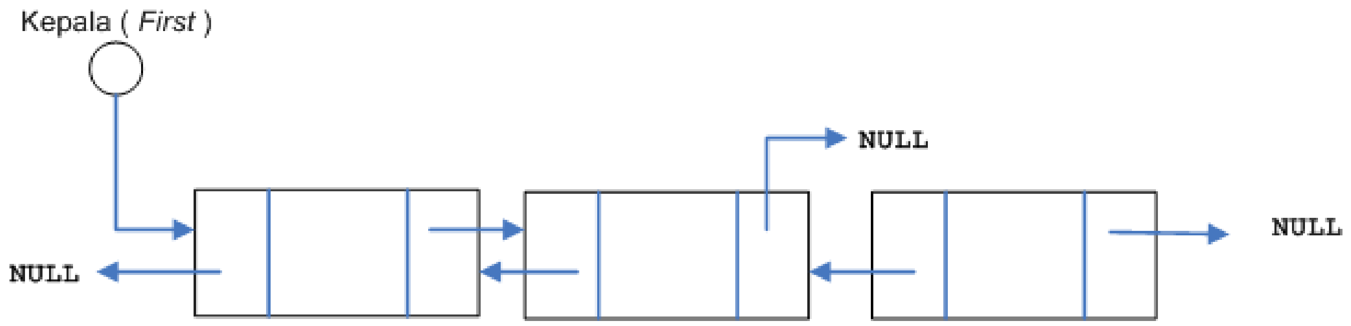
```
void delAfter(elemen *prev, list *L){

   elemen *hapus = prev->next;
   if(hapus->next == NULL){
      prev->next = NULL;
   }else{
      prev->next = hapus->next;
      hapus->next->prev = prev;
      hapus->next = NULL;
   }
   hapus->prev = NULL;
   free(hapus);

}
```

HAPUS DI AKHIR (DELLAST)

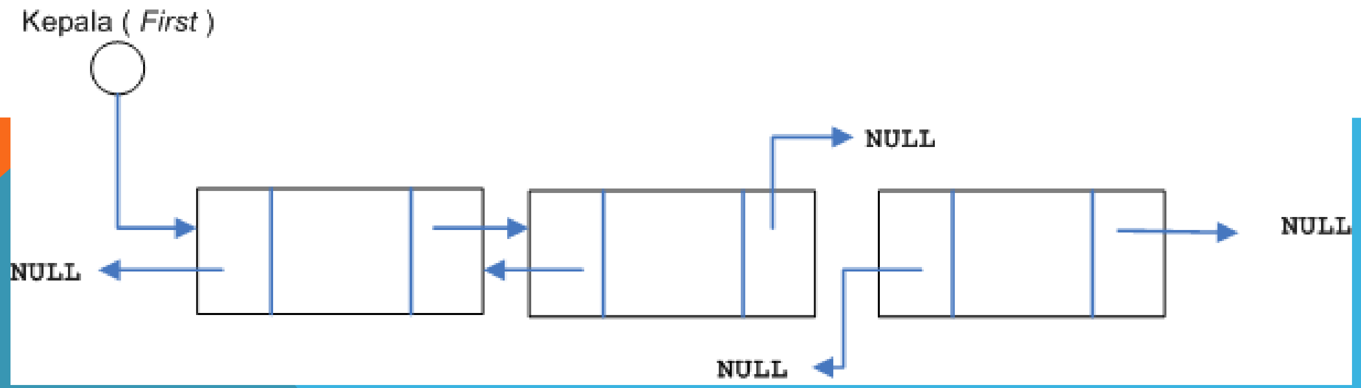
1



2



3



DEL LAST

```
void delLast(list *L){

  if((*L).first != NULL){
    /*jika list tidak kosong*/
    if(countElement(*L) == 1){
      /*list terdiri dari satu
   elemen*/
      delFirst(L);
    }
    else{
      /*mencari elemen terakhir
   list*/
      elemen *hapus =
   (*L).tail;
```

```
      (*L).tail = hapus->prev;
      (*L).tail->next = NULL;
      hapus->prev = NULL;
      free(hapus);
    }
  }
}
```

PRINT ELEMENT

```
void printElement(list L){
  if(L.first != NULL){
    /*jika list tidak kosong*/
    /*inisialisasi*/
    elemen *elmt = L.first;
    int i = 1;
   while(elmt != NULL){
      /*proses*/
      printf("elemen ke : %d\n",
   i);
      printf("nim : %s\n",
        elmt->elmt.nim);
      printf("nama : %s\n",
        elmt->elmt.nama);
      printf("nilai : %s\n",
        elmt->elmt.nilai);
      printf("------------\n");

        /*iterasi*/
      elmt = elmt->next;
        i = i + 1;
      }
    }
    else{
     /*proses jika list kosong*/
      printf("list kosong\n");
    }
}
```

DEL ALL

```
void delAll(list *L){

 if(countElement(*L) != 0){
  int i;

  for(i=countElement(*L);i>=1;i--){
  /*proses menghapus elemen list*/
    delLast(L);
  }
 }
}
```

MAIN

```
int main(){

  list L;

  createList(&L);
  printElement(L);

  printf("=================\n");

  addFirst("1", "Orang_1", "A",
   &L);
  addAfter(L.first, "2",
   "Orang_2", "A", &L);
  addLast("3", "Orang_3", "A",
   &L);
  printElement(L);
  printf("=================\n");
  delLast(&L);
  delAfter(L.first, &L);
  delFirst(&L);
  printElement(L);

   printf("=================\
   n");

  return 0;

}
```

DAFTAR PUSTAKA

S, Rosa A. dan M. Shalahuddin. 2010. Modul Pembelajaran: Struktur Data. Modula: Bandung.