Laporan Resmi
Praktikum Algoritma dan Struktur Data
Double Linked List Delete (Data Int)

Dr. Tita Karlita S.Kom, M.Kom

Nama : Marits Ikmal Yasin

Kelas : 1D4 IT B

NRP : 3121600047

1. Insert Awal

Kode :

```
#include<stdio.h>
#include<stdlib.h>

typedef struct simpul Dnode;
struct simpul{
   int data;
   Dnode *prev, *next;
};

void alokasi();
void awal();
void tampil();

Dnode *head = NULL, *p;

int main(){
   char jwb;

   puts("DLL Insert Awal");
   do{
      fflush(stdin);
      alokasi();
      awal();
      fflush(stdin);
      printf("Mau lagi (y/t) ? ");
      jwb = getchar();
   }while(jwb == 'Y' || jwb == 'y');
   tampil();
   return 0;
}

void alokasi(){
   int x;

   printf("Data yang mau disimpan ? ");
   scanf("%d",&x);
   p = (Dnode *) malloc(sizeof(Dnode));
   if(p == NULL){
      puts("Pemesanan Gagal");
      exit(0);
   }
   else{
      p->data = x;
      p->next = NULL;
      p->prev = NULL;
   }
}

void awal(){
   if(head != NULL){
      p->next = head;
      head->prev = p;
   }
   head = p;
}

void tampil(){
   Dnode *baca;

   puts("Isi dari DLL");
   baca = head;
```

```
    while(baca != NULL){
        printf("%d\n",baca->data);
        baca = baca->next;
    }

}
```

Output :

```
DLL Insert Awal
Data yang mau disimpan ? 3
Mau lagi (y/t) ? y
Data yang mau disimpan ? 2
Mau lagi (y/t) ? y
Data yang mau disimpan ? 1
Mau lagi (y/t) ? t
Isi dari DLL
1
2
3
```

2. Insert Akhir

Kode :

```
#include<stdio.h>
#include<stdlib.h>

typedef struct simpul Dnode;
struct simpul{
    int data;
    Dnode *prev, *next;
};

void alokasi();
void akhir();
void tampil();

Dnode *head = NULL, *p;

int main(){
    char jwb;

    puts("DLL Insert Akhir");
    do{
        fflush(stdin);
        alokasi();
        akhir();
        fflush(stdin);
        printf("Mau lagi (y/t) ? ");
        jwb = getchar();
    }while(jwb == 'Y' || jwb == 'y');
    tampil();
    return 0;
}

void alokasi(){
    int x;

    printf("Data yang mau disimpan ? ");
    scanf("%d",&x);
    p = (Dnode *) malloc(sizeof(Dnode));
    if(p == NULL){
        puts("Pemesanan Gagal");
```

```
        exit(0);
    }
    else{
        p->data = x;
        p->next = NULL;
        p->prev = NULL;
    }
}

void akhir(){
    Dnode *tail;

    tail = head;
    if(head == NULL)
        head = p;
    else{
        while(tail->next != NULL)
            tail = tail->next;
        p->prev = tail;
        tail->next = p;
        tail = p;
    }

}

void tampil(){
    Dnode *baca;

    puts("Isi dari DLL");
    baca = head;
    while(baca != NULL){
        printf("%d\n",baca->data);
        baca = baca->next;
    }

}
```

Output :

```
DLL Insert Akhir
Data yang mau disimpan ? 1
Mau lagi (y/t) ? y
Data yang mau disimpan ? 2
Mau lagi (y/t) ? y
Data yang mau disimpan ? 3
Mau lagi (y/t) ? t
Isi dari DLL
1
2
3
```

3. Insert After

Kode :

```
#include<stdio.h>
#include<stdlib.h>

typedef struct simpul Dnode;
struct simpul{
    int data;
    Dnode *prev, *next;
};
```

```
void alokasi();
void akhir();
void tampil();
void setelah();

Dnode *head = NULL, *p;

int main(){
   char jwb;

   puts("DLL Insert Akhir");
   do{
      fflush(stdin);
      alokasi();
      akhir();
      fflush(stdin);
      printf("Mau lagi (y/t) ? ");
      jwb = getchar();
   }while(jwb == 'Y' || jwb == 'y');
   tampil();
   fflush(stdin);
   puts("Insert After");
   alokasi();
   setelah();
   tampil();
   return 0;
}

void alokasi(){
   int x;

   printf("Data yang mau disimpan ? ");
   scanf("%d",&x);
   p = (Dnode *) malloc(sizeof(Dnode));
   if(p == NULL){
      puts("Pemesanan Gagal");
      exit(0);
   }
   else{
      p->data = x;
      p->next = NULL;
      p->prev = NULL;
   }
}

void akhir(){
   Dnode *tail;

   tail = head;
   if(head == NULL)
      head = p;
   else{
      while(tail->next != NULL)
         tail = tail->next;
      p->prev = tail;
      tail->next = p;
      tail = p;
   }

}
void setelah(){
   int key;
   Dnode *after;
```

```
    printf("Sisip after brp ? ");
    scanf("%d",&key);
    after = head;
    while(after->data != key){
        if(after->next == NULL){
            puts("Data Tidak ada dalam DLL");
            exit(0);
        }
        after = after->next;
    }
    if(after == head){
        p->next = head->next;
        p->prev = head;
        head->next = p;
        head->next->prev = p;
    }
    else{
        p->next = after->next;
        p->prev = after;
        after->next = p;
        if(after->next->next != NULL)
            p->next->prev = p;
    }
}

void tampil(){
    Dnode *baca;

    puts("\nIsi dari DLL");
    baca = head;
    while(baca != NULL){
        printf("%d\n",baca->data);
        baca = baca->next;
    }
}
```

Output :

```
DLL Insert Akhir
Data yang mau disimpan ? 1
Mau lagi (y/t) ? y
Data yang mau disimpan ? 2
Mau lagi (y/t) ? y
Data yang mau disimpan ? 3
Mau lagi (y/t) ? t

Isi dari DLL
1
2
3
Insert After
Data yang mau disimpan ? 5
Sisip after brp ? 2

Isi dari DLL
1
2
5
3
```

4. Insert Before

Kode :

```
#include<stdio.h>
#include<stdlib.h>

typedef struct simpul Dnode;
struct simpul{
   int data;
   Dnode *prev, *next;
};

void alokasi();
void akhir();
void tampil();
void sebelum();

Dnode *head = NULL, *p;

int main(){
   char jwb;

   puts("DLL Insert Akhir");
   do{
      fflush(stdin);
      alokasi();
      akhir();
      fflush(stdin);
      printf("Mau lagi (y/t) ? ");
      jwb = getchar();
   }while(jwb == 'Y' || jwb == 'y');
   tampil();
   fflush(stdin);
   puts("Insert Before");
   alokasi();
   sebelum();
   tampil();
   return 0;
}

void alokasi(){
   int x;

   printf("Data yang mau disimpan ? ");
   scanf("%d",&x);
   p = (Dnode *) malloc(sizeof(Dnode));
   if(p == NULL){
      puts("Pemesanan Gagal");
      exit(0);
   }
   else{
      p->data = x;
      p->next = NULL;
      p->prev = NULL;
   }
}

void akhir(){
   Dnode *tail;

   tail = head;
   if(head == NULL)
      head = p;
   else{
```

```
        while(tail->next != NULL)
            tail = tail->next;
        p->prev = tail;
        tail->next = p;
        tail = p;
    }

}

void sebelum(){
    int key;
    Dnode *before;

    printf("Sisip before brp ? ");
    scanf("%d",&key);
    before = head;
    while(before->data != key){
        if(before->next == NULL){
            puts("Data Tidak ada dalam DLL");
            exit(0);
        }
        before = before->next;
    }
    if(before == head){
        p->next = head;
        head->prev = p;
        head = p;
    }
    else{
        p->next = before;
        p->prev = before->prev;
        before->prev->next = p;
        before->prev = p;
    }
}

void tampil(){
    Dnode *baca;

    puts("\nIsi dari DLL");
    baca = head;
    while(baca != NULL){
        printf("%d\n",baca->data);
        baca = baca->next;
    }
}
```

Output :

```
DLL Insert Akhir
Data yang mau disimpan ? 2
Mau lagi (y/t) ? y
Data yang mau disimpan ? 3
Mau lagi (y/t) ? t

Isi dari DLL
2
3
Insert Before
Data yang mau disimpan ? 1
Sisip before brp ? 2

Isi dari DLL
1
2
3
```

5. Menu Insert

Kode :

```
#include<stdio.h>
#include<stdlib.h>

typedef struct simpul Dnode;
struct simpul{
   int data;
   Dnode *prev, *next;
};

void menu();
void alokasi();
void awal();
void akhir();
void setelah();
void sebelum();
void tampil();

Dnode *head = NULL, *p;

int main(){
   menu();

   return 0;
}

void menu(){
   int pilih;

   do{
      puts("Menu Insert");
      puts("1. Insert Awal");
      puts("2. Insert Akhir");
      puts("3. Insert After");
      puts("4. Insert Before");
      puts("5. Keluar");
      printf("Masukkan pilihan Anda : ");
      scanf("%d",&pilih);
      if(head == NULL && pilih == 3 || head == NULL && pilih == 4){
         puts("Tidak bisa melakukan insert after dan before karena DLL masih kosong");
         exit(0);
      }
      else{
         if(pilih != 5)
            alokasi();
         switch(pilih){
            case 1 :
               awal();
               break;
            case 2 :
               akhir();
               break;
            case 3 :
               setelah();
               break;
            case 4 :
               sebelum();
               break;
            case 5 :
               exit(0);
         }
      }
```

```
        tampil();
    }while(pilih != 5);

}

void alokasi(){
    int x;

    printf("Data yang mau disimpan ? ");
    scanf("%d",&x);
    p = (Dnode *) malloc(sizeof(Dnode));
    if(p == NULL){
        puts("Pemesanan Gagal");
        exit(0);
    }
    else{
        p->data = x;
        p->next = NULL;
        p->prev = NULL;
    }
}

void sebelum(){
    int key;
    Dnode *before;

    printf("Sisip before brp ? ");
    scanf("%d",&key);
    before = head;
    while(before->data != key){
        if(before->next == NULL){
            puts("Data Tidak ada dalam DLL");
            exit(0);
        }
        before = before->next;
    }
    if(before == head){
        p->next = head;
        head->prev = p;
        head = p;
    }
    else{
        p->next = before;
        p->prev = before->prev;
        before->prev->next = p;
        before->prev = p;
    }
}

void setelah(){
    int key;
    Dnode *after;

    printf("Sisip after brp ? ");
    scanf("%d",&key);
    after = head;
    while(after->data != key){
        if(after->next == NULL){
            puts("Data Tidak ada dalam DLL");
            exit(0);
        }
        after = after->next;
    }
```

```
    if(after == head){
        p->next = head->next;
        p->prev = head;
        head->next = p;
        head->next->prev = p;
    }
    else{
        p->next = after->next;
        p->prev = after;
        after->next = p;
        if(after->next->next != NULL)
            p->next->prev = p;
    }
}

void akhir(){
    Dnode *tail;

    tail = head;
    if(head == NULL)
        head = p;
    else{
        while(tail->next != NULL)
            tail = tail->next;
        p->prev = tail;
        tail->next = p;
        tail = p;
    }

}

void awal(){
    if(head != NULL){
        p->next = head;
        head->prev = p;
    }
    head = p;
}

void tampil(){
    Dnode *baca;

    puts("\nIsi dari DLL");
    baca = head;
    while(baca != NULL){
        printf("%d\n",baca->data);
        baca = baca->next;
    }
    puts("");
    if(head == NULL)
        puts("Kosong");

}
```

Output :

```
Menu Insert
1. Insert Awal
2. Insert Akhir
3. Insert After
4. Insert Before
5. Keluar
Masukkan pilihan Anda : 1
Data yang mau disimpan ? 3

Isi dari DLL
3

Menu Insert
1. Insert Awal
2. Insert Akhir
3. Insert After
4. Insert Before
5. Keluar
Masukkan pilihan Anda : 2
Data yang mau disimpan ? 4

Isi dari DLL
3
4

Menu Insert
1. Insert Awal
2. Insert Akhir
3. Insert After
4. Insert Before
5. Keluar
Masukkan pilihan Anda : 3
Data yang mau disimpan ? 5
Sisip after brp ? 4

Isi dari DLL
3
4
5
```

```
Menu Insert
1. Insert Awal
2. Insert Akhir
3. Insert After
4. Insert Before
5. Keluar
Masukkan pilihan Anda : 4
Data yang mau disimpan ? 2
Sisip before brp ? 3

Isi dari DLL
2
3
4
5

Menu Insert
1. Insert Awal
2. Insert Akhir
3. Insert After
4. Insert Before
5. Keluar
Masukkan pilihan Anda : 5

Process returned 0 (0x0)
Press any key to continue.
```

Analisa

DLL sendiri merupakan suatu *linked list* dengan memiliki 2 pointer yaitu pointer *next* dan *prev*.

Algoritma umum dalam melakukan insert pada *Double Linked List* yaitu sama seperti *Single Linked List*. Pertama pesan alamat memori untuk menyimpan data. Setelah pemesanan berhasil maka pilih letak dimana data itu akan diletakkan. Terakhir sambungkan dengan *node* yang telah ada agar tidak terputus.

Insert awal pada DLL, apabila data masih kosong maka langsung saja p di*assign* ke head. Lalu, apabila data sudah terisi maka p->next disambungkan dengan head, setelah itu head->prev disambungkan dengan p. Setelah sudah tersambung maka p akan di*assign* ke head agar head selalu memegang data awal.

Insert akhir pada DLL, sama seperti pada SLL kita memerlukan pointer Bantuan yaitu pointer tail. Pointer tail berfungsi mencari di mana data terakhir berada. Setelah data terakhir ketemu maka data yang sudah kita pesan untuk diletakkan di akhir akan diletakkan di sana. Sebelum itu kita perlu menyambungnya dengan cara tail->next akan menunjuk p dan p->prev akan menunjuk ke tail. Setelah itu tail akan dipindahkan ke tail->next.

Insert after pada DLL, kita memerlukan sebuah pointer Bantuan yaitu pointer after. Pointer after berfungsi untuk mencari di mana key itu berada. Setelah itu saat key ditemukan maka data yang kita pesan akan kita simpan / letakkan di data setelah key itu berada. Tentunya kita perlu untuk menyambung node next dan prev nya agar tidak terputus.

Insert before pada DLL, kita memerlukan sebuah pointer Bantuan yaitu pointer before. Pointer before berfungsi untuk mencari di mana key itu berada. Setelah itu saat key ditemukan maka data yang sudah kita pesan tadi akan kita letakkan di data sebelum key itu berada. Tentunya kita perlu untuk menyambung node next dan prev nya agar tidak terputus.

Untuk menu insert, kita dapat memilih jenis insert yang kita inginkan. Akan tetapi, perlu diketahui bahwa pada saat head masih NULL kita tidak bisa melakukan insert after dan insert before.