IlmuKomputer.Com
Ikatlah ilmu dengan menuliskannya

Search And Insert Problem
(Sorted Double Linkedlist Solution)

Fadlika Dita Nurjanto
fadlikadn@gmail.com
http://fadlikadn.wordpress.com

Problem Search And Insert merupakan salah satu problem yang banyak digunakan dalam kehidupan sehari-hari. Proses dalam Search And Insert adalah mengecek sebuah angka yang diberikan. Jika angka yang dimasukkan telah ada di dalam daftar, maka tidak perlu melakukan apa-apa. Akan tetapi jika angka yang diberikan tidak ada di dalam daftar, maka angka tersebut harus dimasukkan ke dalam list. Search And Insert dapat diselesaikan dengan berbagai cara dan beragam format, salah satunya adalah format terurut. Artinya, angka-angka yang ada di dalam daftar harus diurutkan sedemikian rupa sehingga pencarian lebih mudah dilakukan.

Implementasi problem search and insert bisa diimplementasikan dengan berbagai bahasa pemrograman. Di sini implementasi menggunakan bahasa C/C++.

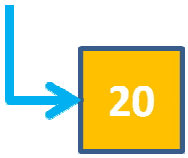
Ada baiknya kita lihat ilustrasi berikut :

Misalkan, ada sebuah barisan angka desimal. Angka-angka tersebut adalah angka 32,30,4,45,12. Barisan tersebut belumlah terurut. Kesulitan akan terasa saat kita melakukan pencarian sebuah angka dalam barisan tersebut dikarenakan angka-angka yang ada belumlah terurut. Setelah diurutkan, barisan akan menjadi seperti ini :

Data yang terurut mempermudah kita melakukan pencarian, dengan cara membandingkan masing-masing anggota barisan dengan angka yang akan kita cari.

Ketentuan :
Saat kita mencari sebuah angka dan angka itu tidak ada di dalam barisan, maka angka itu harus dimasukkan ke dalam barisan dan barisan tersebut harus tetap urut. Jika angka yang dicari ada di dalam daftar, maka tidak perlu diproses lebih lanjut.

Sebagai contoh, dimasukkan angka 20



Karena angka 20 tidak ada di dalam barisan, maka angka 20 harus dimasukkan ke dalam barisan. Sehingga barisan menjadi seperti ini :

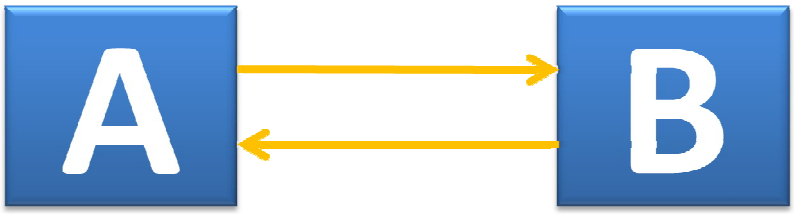
(Ingat, barisan harus dalam keadaan terurut setelah angka dimasukkan)

Sekarang pertanyaannya, bagaimana membuat sebuah program yang mampu mendeteksi sebuah angka di dalam barisan, dan secara otomatis akan dimasukkan jika angka tersebut tidak ada di dalam barisan.

Salah satu teknik yang bisa dilakukan adalah dengan menggunakan teknik linkedlist. Perlu diketahui, linkedlist adalah salah satu representasi struktur data yang dibangun menggunakan tipe data buatan (*struct)* yang dihubungkan oleh pointer yang ada di dalam elemen-elemennya.

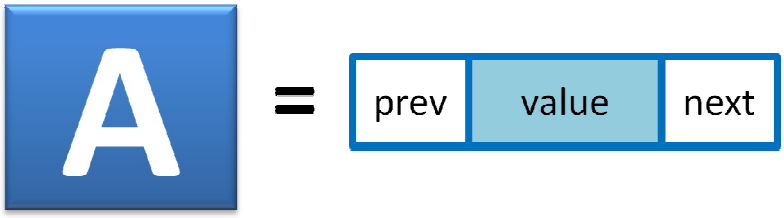
Linkedlist yang digunakan di sini adalah *double linkedlist*, artinya antara masing-masing elemen mempunyai hubungan timbal balik dengan elemen yang persis di sampingnya.

Ilustrasi Double Linkedlist



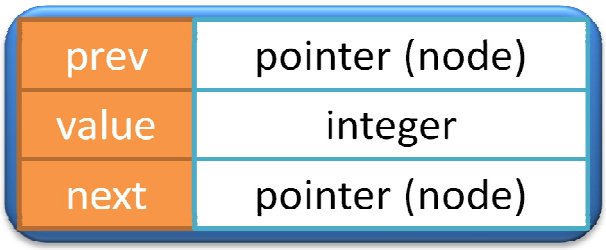
Elemen A dan B mempunyai arah ke elemen yang lain. Dari A ke B, demikian sebaliknya. Elemen A dan B bertipe data struct, yaitu data buatan. Data buatan adalah data yang dibuat secara manual dengan mengkombinasikan tipe data yang ada kemudian dibungkus ke dalam tipe data yang baru.

Di dalam elemen A, terdapat beberapa tipe data yang mempunyai fungsinya sendiri. (Misal, tipe data buatan bernama tipe data *Node*.



Elemen A bertipe data *Node*. Tipe data *Node* ini mempunyai 3 tipe data di dalamnya, yaitu tipe data 2 Node (pointer) dan integer. Ilustrasinya seperti ini :

Tipe data *Node*



Keterangan :

Prev : digunakan sebagai penunjuk ke elemen sebelumnya pada barisan bilangan.
Value : digunakan untuk menampung angka yang ada di dalam sebuah *Node*.
Next : digunakan sebagai penunjuk ke elemen berikutnya pada barisan bilangan.

Implementasi tipe data buatan pada C/C++

```
typedef struct Node
{
   int Value;
   struct Node *Next;
   struct Node *Prev;
};
```

Berikut adalah source lengkap problem Search And Insert

```
/*
Teknik Informatika
Institut Teknologi Sepuluh Nopember
Fadlika Dita Nurjanto
E-mail :  fadlikadn@gmail.com
Blog   :  fadlikadn.wordpress.com

Problem Search and Insert
Solution with sorted linked list
*/

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

```
typedef struct Node
{
   int Value;
   struct Node *Next;
   struct Node *Prev;
};
typedef struct Node Linked;

//fungsi untuk membuat node baru
void make_node(int X, Linked **P)
{
   *P = (Linked *) malloc (sizeof(Linked));
   if(*P != NULL)
   {
      (*P)->Value = X;
      (*P)->Next  = NULL;
      (*P)->Prev  = NULL;
   }
   else
      printf("Make new node has failed !\n");
}

//fungsi pertama kali yang dipanggil, untuk membuat First dan Last menjadi NULL
void Initial(Linked **F_loc, Linked **L_loc)
{
   *F_loc = NULL;
   *L_loc = NULL;
   printf("F and L set to NULL\n");
}

//insert first, F & L dalam keadaan NULL
void insert_first(Linked **F_loc, Linked **L_loc, Linked **P_loc)
{
   (*P_loc)->Next = NULL;
   (*P_loc)->Prev = NULL;
   *F_loc = *P_loc;
   *L_loc = *P_loc;
   printf("Initial first has change F=%d and L=%d\n",(*F_loc)->Value,(*L_loc)->Value);
}

void search_value(int X, Linked **P_loc, Linked **Q_loc, Linked **F_loc, Linked **L_loc)
{
   *Q_loc = *F_loc;
   while((*Q_loc)->Next!=NULL)
   {
      if((*Q_loc)->Value == (*P_loc)->Value)
      {
         printf("Value has found, nothing to do\n");
         break;
      }
      else
      {
         if((*Q_loc)->Value < (*P_loc)->Value)
         {
```

```
            *Q_loc = (*Q_loc)->Next;
        }
        else if((*Q_loc)->Value > (*P_loc)->Value)
        {
            printf("Value not found, insert into sorted list\n");
            if((*Q_loc)->Prev != NULL)
            {
                (*P_loc)->Prev = (*Q_loc)->Prev;
                ((*Q_loc)->Prev)->Next = *P_loc;
                (*Q_loc)->Prev = *P_loc;
                (*P_loc)->Next = *Q_loc;
            }
            else if((*Q_loc)->Prev == NULL)
            {
                (*P_loc)->Next = *Q_loc;
                (*Q_loc)->Prev = *P_loc;
                *F_loc = *P_loc;
            }
            printf("Insert into sorted list has successfully\n");
            break;
        }
    }
}
if((*Q_loc)->Next==NULL)
{
    if((*Q_loc)->Value == (*P_loc)->Value)
    {
        printf("Value has found, nothing to do\n");
    }
    else
    {
        if((*Q_loc)->Value < (*P_loc)->Value)
        {
            printf("%d lebih kecil dari %d\n",(*Q_loc)->Value,(*P_loc)->Value);
            (*Q_loc)->Next = *P_loc;
            (*P_loc)->Prev = *Q_loc;
            (*P_loc)->Next = NULL;
            printf("Insert into sorted list has successfully\n");
        }
        else if((*Q_loc)->Value > (*P_loc)->Value)
        {
            printf("%d lebih besar dari %d\n",(*Q_loc)->Value,(*P_loc)->Value);
            if((*Q_loc)->Prev != NULL)
            {
                (*P_loc)->Prev = (*Q_loc)->Prev;
                ((*Q_loc)->Prev)->Next = *P_loc;
                (*Q_loc)->Prev = *P_loc;
                (*P_loc)->Next = *Q_loc;
                printf("Insert into sorted list has successfully\n");
            }
            else if((*Q_loc)->Prev == NULL)
            {
                printf("Element only one\n");
                (*P_loc)->Next = *Q_loc;
```

```
            (*Q_loc)->Prev = *P_loc;
            *F_loc = *P_loc;
            printf("F = %d, L= %d\n",(*F_loc)->Value, (*L_loc)->Value);
            printf("Insert into sorted list has successfully\n");
          }
        }
      }
    }
  }

//fungsi utama, akan memanggil fungsi-fungsi yang lain
void main_process(int X, Linked **P_loc, Linked **Q_loc, Linked **F_loc, Linked **L_loc)
{
  if(*F_loc == NULL && *L_loc == NULL) //jika list masih kosong
  {
    make_node(X,&*P_loc);
    insert_first(&*F_loc,&*L_loc,&*P_loc);
  }
  else //jika list sudah berisi
  {
    make_node(X,&*P_loc); //alokasi memori dulu
    //proses search
    search_value(X,&*P_loc,&*Q_loc,&*F_loc,&*L_loc);
    //void search_value(int X, Linked **P_loc, Linked **Q_loc, Linked **F_loc, Linked **L_loc)
  }
}

void list_print_all(Linked **Q_loc, Linked **F_loc)
{
  if(*F_loc!=NULL)
  {
    *Q_loc = *F_loc;
    int valid = 1;
    while(valid == 1)
    {
      printf("%d ",(*Q_loc)->Value);
      if((*Q_loc)->Next==NULL) break;
      else *Q_loc = (*Q_loc)->Next;
    }
    printf("\n");
  }
  else printf("List haven't created\n");
}

int main()
{
  Linked *P, *F, *L, *Q;
  int val;
  Initial(&F,&L);
  printf("Search and Insert with Sorted Linked List\n");
  printf("Type 1 to check an integer and type 0 to exit\n");
  printf("Type 2 to print list\n");
  int ch;
  while(scanf("%d",&ch) && ch != 0)
```

```
  {
    if(ch==1)
    {
      printf("Type an integer to check in the list\n");
      scanf("%d",&val);
      //void main_process(int X, Linked **P_loc, Linked **Q_loc, Linked **F_loc, Linked **L_loc)
      main_process(val,&P,&Q,&F,&L);
    }
    else if(ch==2)
    {
      //void list_print_all(Linked **Q_loc, Linked **F_loc)
      list_print_all(&Q,&F);
    }
  }
  return 0;
}
```

Keterangan Fungsi :

1. ***Make Node***
   Fungsi ini digunakan untuk mengalokasikan sebuah tempat pada memori yang disimpan pada variabel *P.

2. ***Initial***
   Fungsi ini digunakan untuk inisialisasi awal linkedlist, yaitu mereset elemen FIRST dan LAST pada linkedlist menjadi NULL (FIRST dan LAST digunakan dalam pencarian).

3. ***Insert First***
   Fungsi ini digunakan untuk memasukkan sebuah angka ke dalam alamat alokasi memori yang telah dibuat pada linkedlist untuk pertama kalinya. (FIRST dan LAST masih bersta.tus NULL)

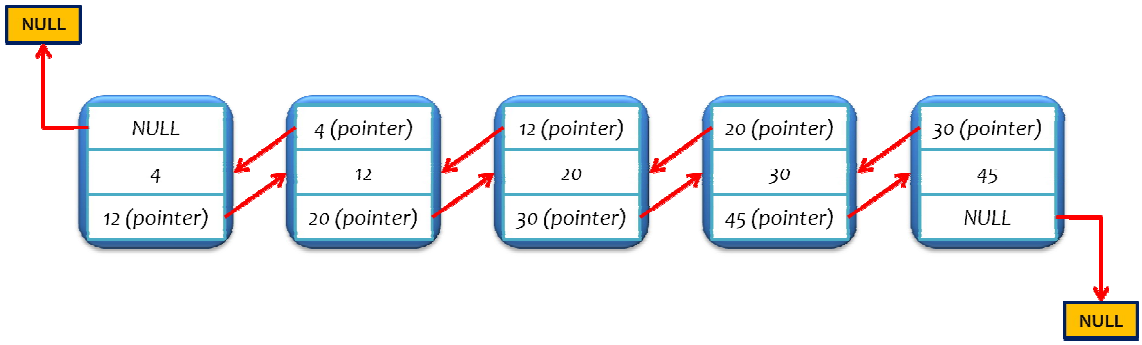
4. ***Search Value***
   Fungsi ini digunakan untuk mencari angka di dalam list. Jika angka ditemukan, tidak dilakukan apa-apa. Tetapi bila angka tidak ditemukan, maka angka dimasukkan ke dalam list dengan keadaan terurut (dari kecil ke besar). Fungsi Search Value merupakan fungsi yang serbaguna karena dapat menangani beberapa kondisi di dalam list.

5. ***Main Process***
   Fungsi ini digunakan untuk mengatur jalannya input yang masuk dan membaginya sesuai kondisi yang ada. Jika elemen FIRST dan LAST masih bernilai NULL (belum ada list), maka fungsi yang akan dipanggil adalah fungsi *Insert First*, tetapi bila list sudah ada, fungsi yang akan dijalankan adalah fungsi *Search Value*.

6. ***List Print All***
   ungsi ini digunakan untuk mencetak semua elemen yang ada di dalam sebuah list. (Dari awal sampai akhir), untuk mengecek apakah list sudah urut atau belum.

7. ***Main***
   Fungsi ini merupakan fungsi utama di dalam program ini. Fungsi main adalah fungsi pokok yang ada di dalam sebuah program. Di sini, fungsi main berfungsi untuk meng-handle inputan di dalam program.

Linkedlist yang terbentuk akan menjadi seperti ini :



Representasi dengan Sorted Double Linkedlist

Referensi

Horowitz; Sahni; Anderson; Freed. 2008. Fundamentals Of Data Structure In C (2nd Edition).Silicon Press

Biografi Penulis

Fadlika Dita Nurjanto. Menyelesaikan pendidikan di SD 2 Wonosobo tahun 2004, SMP 1 Wonosobo tahun 2007, dan SMK 1 tahun 2010. Sekarang berstatus sebagai mahasiswa Teknik Informatika Institut Teknologi Sepuluh Nopember, Surabaya. Artikel menarik lainnya bisa ditemukan di http://fadlikadn.wordpress.com. Sharing dengan penulis bisa kirim email ke fadlikadn@gmail.com.