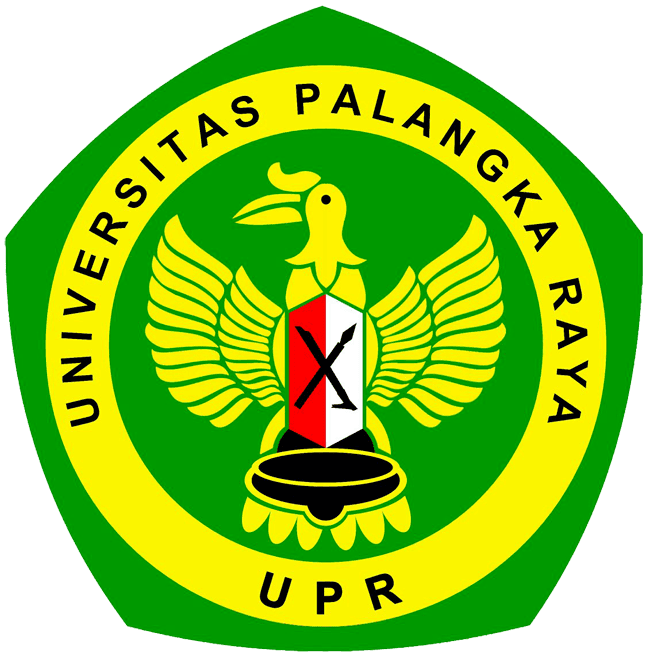
**LAPORAN RENCANA PRAKTIKUM**

**STRUKTUR DATA**



|  |  |  |
| --- | --- | --- |
| **NAMA** | **:** | **INDRA FIQI RIPANI** |
| **NIM** | **:** | **213010503002** |
| **KELAS** | **:** | **F** |
| **MODUL** | **:** | **III (*LINKED LIST*)** |

**Program Studi S1 Teknik Informatika**

**Fakultas Teknik**

**Universitas Palangka Raya**

**Palangka Raya, Kalimantan Tengah**

**2022**

**MODUL III**

***LINKED LIST***

1. **TUJUAN PRAKTIKUM**
2. Mahasiswa memahami struktur data *linked list*.
3. Mahasiswa mampu menggunakan struktur data *linked list* di dalam menyelesaikan masalah pemrograman.
4. **Landasan Teori**

*Linked list* adalah suatu cara untuk menyimpan data dengan struktur sehingga *programmer* dapat secara otomatis menciptakan suatu tempat baru untuk menyimpan data kapan saja diperlukan. *Linked list* dikenal juga dengan sebutan senarai berantai adalah stuktur data yang terdiri dari urutan *record* data dimana setiap *record* memiliki *field* yang menyimpan alamat/referensi dari *record* selanjutnya (di dalam urutan). Elemen data yang dihubungkan dengan *link* pada *linked list* disebut Node. Biasanya dalam suatu linked list, terdapat istilah *head* dan *tail*.

Jenis *linked list* (yang akan dipelajari) adalah:

1. *Single Linked List*
2. *Double Linked List*
3. *Circular Linked List*
4. *Multiple Linked List*

Ada 5 proses dasar dalam *linked list*:

1. Proses Inisialisasi

* Proses awal 🡪 menyatakan *linked list* belum ada
* Algoritma:

|  |
| --- |
| First = Null;  Last = Null; |

1. Proses Simpul Baru

* Instruksi:

|  |
| --- |
| P = (simpul\*) malloc(sizeof(simpul)); |

* Algoritma:

|  |
| --- |
| void Buat\_Simpul (int x){  P = (simpul\*) malloc(sizeof(simpul));  if (P!=NULL){  P → Info = x;  }  else cout << “simpul gagal dibuat”;  } |

1. Membuat Simpul Awal

Syarat:

1. *Linked list* belum ada
2. Sudah ada simpul yang akan dijadikan simpul awal

Algoritma:

|  |
| --- |
| void Awal (){  First = P;  Last = P;  P → Link = NULL;  } |

1. Menambahkan Simpul Baru ke dalam *Linked List* (*Insert*)

Syarat:

1. *Linked list* sudah ada.
2. Sudah ada simpul yang akan ditambahkan *linked list*.
3. *Insert* Kanan/Akhir

Algoritma:

|  |
| --- |
| void Ins\_Akhir (){  Last → Link = P;  Last = P;  P → Link = NULL;  } |

1. *Insert* Kiri/Awal

Algoritma:

|  |
| --- |
| void Ins\_Awal (){  P → Link = First;  First = P;  } |

1. *Insert* Tengah

Algoritma:

|  |
| --- |
| void Ins\_Tengah (){  P → Link = Q → Link;  Q → Link = P;  } |

1. Menghapus Sebuah Simpul dari *Linked List* (Delete)

Syarat:

1. *Linked list* sudah ada.
2. *Delete* Kanan/Akhir

Algoritma:

|  |
| --- |
| void Del\_Akhir (){  Free (Last);  Last = Q;  Last → Link = NULL;  } |

1. *Delete* Kiri/Awal

Algoritma:

|  |
| --- |
| void Del\_Awal (){  Q = First;  First = Q → Link;  Free (Q);  } |

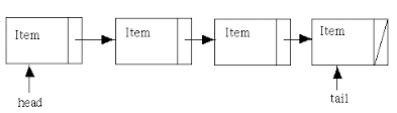
1. *Delete* Tengah

Algoritma:

|  |
| --- |
| void Del\_Tengah (){  R = Q → Link;  Q → Link = R → Link;  Free (R);  } |

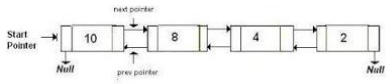
***Single Linked List***

*Single Linked List* merupakan suatu *linked list* yang hanya memiliki satu variabel *pointer* saja. Di mana pointer tersebut menunjuk ke node selanjutnya. Biasanya *field* pada *tail* menunjuk ke NULL.



***Double Linked List***

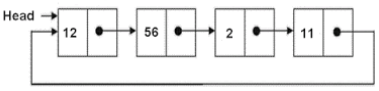
*Double Linked List* merupakan suatu *linked list* yang memiliki dua variabel *pointer* yaitu *pointer* yang menunjuk ke node selanjutnya dan *pointer* yang menunjuk ke node sebelumnya. Setiap *head* dan *tail*nya juga menunjuk ke NULL.



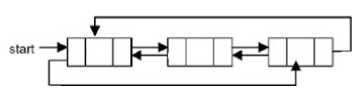
***Circular Linked List***

*Circular Linked List* merupakan suatu *linked list* dimana *tail* (node terakhir) menunjuk ke *head* (node pertama). Jadi tidak ada *pointer* yang menunjuk NULL. Ada 2 jenis *Circular Linked List*, yaitu:

* *Circular Single Linked List*

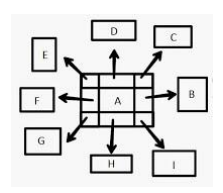


* *Circular Double Linked List*



***Multiple Linked List***

*Multiple Linked List* merupakan suatu *linked list* yang memiliki lebih dari 2 buat variabel pointer.



1. **TUGAS**
2. Buatlah sebuah *linked list non circular* yang berisi nama lengkap dan nim anda!
3. Buatlah sebuah program dengan menggunakan *single linked list non circular* dengan fungsi-fungsi (buat dengan menggunakan menu):

* Menambah data (dari depan dan dari belakang)
* Menghapus data (dari depan dan dari belakang)
* Mencetak data

1. Buat sebuah program *double linked list* dengan fungsi-fungsi (buat dengan menggunkan menu):

* Menambahkan data (dari depan sesudah simpul dan dari belakang sesudah simpul)
* Menghapus data (dari tengah)
* Data yang dimasukkan langsung ke tampil dalam menu