PRAKTIKUM STRUKTUR DATA

JOB 4

DOUBLY LINKED LIST



OLEH:

NAJWA KURNIA

23343058

INFORMATIKA (NK)

DEPARTEMEN ELEKTRONIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI PADANG

2024

Analisis program

|  |  |  |  |
| --- | --- | --- | --- |
| No program | Baris program | Petikan source code | Penjelasan |
| 1 | 4-9 | struct Node  {  int data;  struct Node \*next;  struct Node \*prev;  }; | Deklarasi struktur baru dengan nama  node (simpul). Next dan prev adalah  variable pointer yang akan digunakan  untuk mengarahkan ke simpul  sebelum atau setelah sebuah simpul  baru dibuat |
| 1 | 10-19 | void push(struct Node\*\* head\_ref, int new\_data)  {  struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));  new\_node->data = new\_data;  new\_node->next = (\*head\_ref);  new\_node->prev = NULL;  if ((\*head\_ref) != NULL)  (\*head\_ref)->prev = new\_node;  (\*head\_ref) = new\_node;  } | Deklarasi fungsi push yang mengambil parameter berupa pointer ke pointer ke struct node. Mengalokasikan memori untuk node baru dengan fungsi malloc untuk menyimpan satu node baru dalam linked list. Data disimpan dalam bagian data dari node baru yang dibuat. Node baru diatur untuk menunjuk ke node yang saat ini menjadi kepala dalam linked list. Memeriksa jika linked list tidak kosong maka node yang sebelumnya menjadi kepala diatur menunjuk ke node baru yang akan menjadi kepala. |
| 1 | 20-28 | void printList(struct Node\* node)  {  struct Node\* last;  printf("\nTraversal in forward direction \n");  while (node != NULL) {  printf(" %d ", node->data);  last = node;  node = node->next;  } | Deklarasi fungsi ‘printList’ untuk mengatur output atau mencetak linked list yang sudah di buat. Mendeklarasi struct node dengan pointer last. Mencetak “traversal in forward direction”dan Melakukan perulangan dengan argument memeriksa simpul tidak kosong maka node dicetak, last menjadi simpul, simpul menunjuk ke node selanjutnya. |
| 1 | 29-34 | printf("\nTraversal in reverse direction \n");  while (last != NULL) {  printf(" %d ", last->data);  last = last->prev;  }  } | Mencetak “traversal in reverse direction”, melakukan perulangan dengan argument memeriksa last tidak kosong maka node dicetak, node last menunjuk ke node sebelumnya. |
| 1 | 35-45 | int main()  {  struct Node\* head = NULL;  push(&head, 6);  push(&head, 5);  push(&head, 2);  printf("Created DLL is: ");  printList(head);  getchar();  return 0;  } | Deklarasi fungsi utama untuk menjalankan fungsi push dengan mengisi argument dan fungsi untuk mencetak linked list. |
| 2 | 3-8 | struct Node  {  int data;  struct Node \*next;  struct Node \*prev;  }; | Deklarasi struct dengan nama node.  Variable data, struct node pointer next, dan struct node pointer prev (sebelumnya) didalam struct node. |
| 2 | 9-22 | void push(Node\*\* head\_ref, int new\_data)  {  Node\* new\_node = new Node();  new\_node->data = new\_data;  new\_node->next = (\*head\_ref);  new\_node->prev = NULL;  if ((\*head\_ref) != NULL)  (\*head\_ref)->prev = new\_node;  (\*head\_ref) = new\_node;  } | Fungsi void dengan nama push dengan parameter struct dua pointer head\_ref, dan variable new\_data tipe data integer. Inisialisasi node baru dan menyimpan alamatnya dalam pointer new\_node. Membuat node baru dan menyimpannya ke node yang baru dibuat. Node baru pointer next menetapkan node baru menjadi node pertama dari linked list. Node baru pointer sebelumnya kosong. Memeriksa jika linked list pertama kosong , maka pointer prev dari node pertama diatur untuk menunjuk ke node baru. Node baru jadi kepala dari linked list. |
| 2 | 23-36 | void insertAfter(struct Node\* prev\_node, int new\_data)  {  if (prev\_node == NULL) {  printf("the given previous node cannot be NULL");  return;  }  struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));  new\_node->data = new\_data;  new\_node->next = prev\_node->next;  prev\_node->next = new\_node;  new\_node->prev = prev\_node;  if (new\_node->next != NULL)  new\_node->next->prev = new\_node;  } | Fungsi void insert after dengan parameter struct node pointer prev\_node dan variable new\_data. Memeriksa jika node sebelumnya kosong, maka mencetak “the given previous node cannot be NULL”. Alokasikan memori Struct node pointer new\_node. New\_node pointer data membuat data baru. Node baru pointer next menjadikan node sebelumnya menjadi setelah data baru. Node sebelumnya pointer next membuat node baru. Node baru pointer prev menjadi node sebelumnya. Memeriksa jika node baru pointer next tidak kosong maka node baru pointer next pointer prev menjadi node baru. |
| 2 | 37-51 | void printList(struct Node\* node)  {  struct Node\* last;  printf("\nTraversal in forward direction \n");  while (node != NULL) {  printf(" %d ", node->data);  last = node;  node = node->next;  }  printf("\nTraversal in reverse direction \n");  while (last != NULL) {  printf(" %d ", last->data);  last = last->prev;  }  } | Fungsi void print list untuk mencetak linked list dengan parameter struct node pointer node. Inisialisasi Struct node pointer last. Mencetak “transversal in fordward direction”, melakukan perulangan jika node tidak kosong maka cetak node menunjuk ke data. Last menjadi node, node menunjuk ke next.  Mencetak “tranversal in reverse direction. Melakukan perulangan untuk mencek last tidak kosong maka cetak isi dari last menunjuk ke data. Last menunjuk ke prev/sebelum. |
| 2 | 52-63 | int main()  {  struct Node\* head = NULL;  push(&head, 6);  push(&head, 5);  push(&head, 2);  insertAfter(head->next, 5);  printf("Created DLL is: ");  printList(head);  getchar();  return 0;  } | Fungsi main/utama untuk menjalankan fungsi dan output hasil dari linked list. Inisialisasi struct node pointer head kosong. Inisialisasi fungsi push dengan argument head dan nilai 6, head dan nilai 5, dan head dan nilai 2. Inisialisasi fungsi insertAfter untuk menmabahkan data baru dengan argument head menunjuk next dan nilai 5. Mencetak “creat DLL is:”. Inisialisasi fungsi printlist dengan argument head. |
| 3 | 3-7 | struct Node  {  int data;  struct Node \*next;  struct Node \*prev;  }; | Deklarasi struct dengan nama node. Inisialisasi variable data dengan tipe data integer, struct pointer ke next node, dan struct pointer ke node sebelumnya. |
| 3 | 9-18 | void push(Node\*\* head\_ref, int new\_data)  {  Node\* new\_node = new Node();  new\_node->data = new\_data;  new\_node->next = (\*head\_ref);  new\_node->prev = NULL;  if ((\*head\_ref) != NULL)  (\*head\_ref)->prev = new\_node;  (\*head\_ref) = new\_node;  } | Deklarasi fungsi void dengan nama push, parameter double pointer head\_ref, dan variable new\_data dengan tipe data integer. Alokasikan node baru ke pointer node. Node baru mengambil data baru. Membuat next pada node baru sebagai kepala linked list dan node sebelumnya dikosongkan. Memeriksa jika kepala linked list tidak kosong, maka ubah kepala node sebelumnya menjadi node baru. Pindahkan kepala linked list ke node baru. |
| 3 | 19-36 | void append(struct Node\*\* head\_ref, int new\_data)  {  struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));  struct Node\* last = \*head\_ref;  new\_node->data = new\_data;  new\_node->next = NULL;  if (\*head\_ref == NULL) {  new\_node->prev = NULL;  \*head\_ref = new\_node;  return;  }  while (last->next != NULL)  last = last->next;  last->next = new\_node;  new\_node->prev = last;  return;  } | Fungsi void dengan variable append, dengan parameter double pointer head\_ref dari struct node dan variable new\_data dengan tipe data integer. Alokasi memori untuk pointer new\_node dari struct node. Pointer last dari struct node jadi pointer head\_ref/kepala linked list. Mengambil new\_node yang menunjuk ke data yang akan membuat data baru. Mengosongkan new\_node yang menunjuk next. Mengecek jika pointer head\_ref kosong, maka node baru yang menunjuk sebelum/prev kosong, pointer head\_ref/kepala linked list berisi node baru. Melakukan perulangan untuk mengecek last yang menunjuk next tidak kosong, maka last akan menunjuk next. Mengubah last yang menunjuk next menjadi node baru. Node baru yang menunjuk prev menjadi node last. |
| 3 | 37-51 | void printList(struct Node\* node)  {  struct Node\* last;  printf("\nTraversal in forward direction \n");  while (node != NULL) {  printf(" %d ", node->data);  last = node;  node = node->next;  }  printf("\nTraversal in reverse direction \n");  while (last != NULL) {  printf(" %d ", last->data);  last = last->prev;  }  } | Fungsi void untuk mencetak linked list dengan parameter pointer node dari struct node. . Inisialisasi Struct node pointer last. Mencetak “transversal in fordward direction”, melakukan perulangan jika node tidak kosong maka cetak node menunjuk ke data. Last menjadi node, node menunjuk ke next.  Mencetak “tranversal in reverse direction. Melakukan perulangan untuk mencek last tidak kosong maka cetak isi dari last menunjuk ke data. Last menunjuk ke prev/sebelum. |
| 3 | 52-63 | int main()  {  struct Node\* head = NULL;  append(&head, 6);  push(&head, 7);  push(&head, 1);  append(&head, 4);  printf("Created DLL is: ");  printList(head);  getchar();  return 0;  } | Fungsi main/utama untuk menjalankan program dan mencetak hasil dari program. Memulai pointer dari struct node menjadi kosong. Mendeklarasi fungsi append dengan argument head dan nilai 6 yang akan di tampilkan. Deklarasi fungsi push dengan argument head dan nilai 7, head dan nilai 1. Deklarasi fungsi append dengan argument head dan nilai 4. Mencetak “created DLL is:”, menjalankan fungsi printlist dengan argument head. |
| 4 | 4-8 | struct Node {  int data;  struct Node\* next;  struct Node\* prev;  }; | Deklarasi struct dengan variable node yang berisi variable data, pointer next dari struct node, dan pointer prev dari struct node. |
| 4 | 9-18 | void push(struct Node\*\* head\_ref, int new\_data)  {  struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));  new\_node->data = new\_data;  new\_node->next = (\*head\_ref);  new\_node->prev = NULL;  if ((\*head\_ref) != NULL)  (\*head\_ref)->prev = new\_node;  (\*head\_ref) = new\_node;  } | Fungsi push dengan parameter double pointer head\_ref dari struct node dan variable new\_data. Alokasi memori new\_node dari struct node. Node baru yang menunjuk data berisi data baru. Node baru yang menunjuk next berisi head\_ref/kepala linked list. Node baru menunjuk prev dikosongkan. Mengecek jika kepala linked list tidak kosong, maka pointer kepala linked list menunjuk prev yang berisi node baru. Kepala linked list berisi node baru. |
| 4 | 20-26 | void insertBefore(struct Node\*\* head\_ref, struct Node\* next\_node, int new\_data)  {  /\*1. check if the given next\_node is NULL \*/  if (next\_node == NULL) {  printf("the given next node cannot be NULL");  return;  } | Fungsi void insertBefore dengan parameter double pointer head\_ref dari struct node, pointer next\_node dari struct node, variable new\_data. Mengecek jika node selanjutnya kosong, maka mencetak “the given next node cannot be NULL”, mengembalikan ke fungsi void. |
| 4 | 27-44 | struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node));  new\_node->data = new\_data;  new\_node->prev = next\_node->prev;  next\_node->prev = new\_node;  new\_node->next = next\_node;  if (new\_node->prev != NULL)  new\_node->prev->next = new\_node;  else  (\*head\_ref) = new\_node;  } | Alokasi memori struct node untuk pointer new\_node. Node baru menunjuk data yang berisi data baru. Node baru menunjuk prev yang berisi node selanjutnya menunjuk prev. node selanjutnya menunjuk prev yang berisi node baru. Node baru menunjuk selanjutnya yang berisi node selanjutnya. Mengecek jika node baru yang menunjuk prev tidak kosong, maka node baru yang menunjuk prev dan menunjuk next yang akan berisi node baru. Jika kondisi kondisi kosong, maka kepala linked list akan berisi node baru. |
| 4 | 61-73 | int main()  {  /\* Start with the empty list \*/  struct Node\* head = NULL;  push(&head, 7);  push(&head, 1);  push(&head, 4);  insertBefore(&head, head->next, 8);  printf("Created DLL is: ");  printList(head);  getchar();  return 0;  } | Fungsi main/utama untuk menjalankan program dari fungsi. Dimulai dengan inisialisasi pointer head yang kosong dari struct node. Inisialisasi fungsi push untuk menambahkan linked list dengan argument head dan nilai 7, head dan nilai 1, head dan nilai 4. Inisialisasi fungsi insertBefore dengan argument head, head menunjuk ke next dan nilai 8. Mencetak “created DLL is:”. Inisialisasi fungsi printList dengan argument head untuk membuat linked list. |