TUGAS 4 PRAKTIKUM STRUKTUR DATA

Jobsheet 4

Disusun Oleh :

Fitri Waldi : 22343021

Dosen Pengampu :

Randi Proska Sandra, S.Pd., M.Sc

PROGRAM STUDI S1 INFORMATIKA (NK)

JURUSAN TEKNIK ELEKTRONIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI PADANG

2023

Nomor Program	Baris Program	Petikan Source Code	Penjelasan

1	7-12	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

	15	void push(struct Node** head_ref, int new_data)
		{
		// 1. alokasi memori untuk node baru
		struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
		
		// 2. isi data pada node baru
		new_node->data = new_data;
		
		// 3. hubungkan node baru dengan head dan previous dengan NULL
		new_node->next = (*head_ref);
		new_node->prev = NULL;
		
		// 4. jika head bukan NULL, ubah previous dari head menjadi node baru
		if ((*head_ref) != NULL)
		(*head_ref)->prev = new_node;
		
		// 5. pindahkan head untuk menunjuk ke node baru
		(*head_ref) = new_node;
		}
			Fungsi push digunakan untuk menambahkan simpul baru pada awal linked list

	17-21	// 1. alokasi memori untuk	Mengalokasikan

		node baru
		struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
		
		// 2. isi data pada node baru
		new_node->data = new_data;
			memori untuk simpul baru dengan menggunakan fungsi '**malloc'** dan ukurannya disesuaikan dengan ukuran '**struct Node'.** Mengisi data pada simpul baru dengan nilai '**new_data**'.

	23-25	// 3. hubungkan node baru dengan head dan previous dengan NULL
		new_node->next = (*head_ref);
		new_node->prev = NULL;
			Menghubungkan simpul baru dengan simpul head yang sudah ada.**' Next'** dari simpul baru akan menunjuk ke simpul head dan **'prev'** diisi dengan **'NULL'** karena simpul baru akan diletakkan di awal linked list.

	27-29	// 4. jika head bukan NULL, ubah previous dari head menjadi node baru if ((*head_ref) != NULL) (*head_ref)->prev = new_node;	Jika simpul head tidak **'NULL'**, artinya linked list sudah memiliki simpul lain selain simpul head, maka previous dari simpul head akan diubah agar menunjuk ke simpul baru.

2	7-11	struct Node {
		int data;
		struct Node* next; // Pointer to next node
		struct Node* prev; // Pointer to previous node
		};
			'struct Node' adalah struktur yang digunakan untuk merepresentasikan simpul atau node dalam linked list.

	7-11	struct Node { int data;	Setiap simpul memiliki dua

		struct Node* next; // Pointer to next node
		struct Node* prev; // Pointer to previous node
		};
			anggota, yaitu **'data'** yang merupakan nilai yang disimpan dalam simpul, dan **'next'** dan **'prev'** yang merupakan pointer yang menunjukkan ke simpul berikutnya dan sebelumnya dalam linked list.

	25-26	// 5. move the head to point to the new node (*head_ref) = new_node;	Pindahkan pointer **'head_ref'** untuk menunjuk ke simpul baru karena simpul baru sudah menjadi simpul pertama di linked list.

	29-49	void insertAfter(Node* prev_node, int new_data) {
		// 1. check if the given prev_node is NULL
		if (prev_node == NULL) {
		printf("the given previous node cannot be NULL");
		return;
		}
		// 2. allocate new node
		Node* new_node = (Node*)malloc(sizeof(Node));
		// 3. put in the data
		new_node->data = new_data;
		// 4. Make next of new node as next of prev_node
		new_node->next = prev_node->next;
		// 5. Make the next of prev_node as new_node
			Fungsi untuk menyisipkan sebuah simpul (node) baru setelah simpul tertentu dalam sebuah linked list.

		prev_node->next = new_node;
		// 6. Make prev_node as previous of new_node
		new_node->prev = prev_node;
		// 7. Change previous of new_node's next node
		if (new_node->next != NULL) {
		new_node->next->prev = new_node;
		}
		}	

	51-64	void printList(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 **'printList'** digunakan untuk menampilkan data yang tersimpan dalam linked list secara berurutan, baik dari depan maupun belakang.

3	27-48	void append(Node **head_ref, int new_data) {
		// 1. Allocate node
		Node *new_node = (Node *)malloc(sizeof(Node));
		// 2. Put in the data
		new_node->data = new_data;
		// 3. This new node is going to be the last node, so make next of it as NULL
		new_node->next = NULL;
		// 4. If the Linked List is empty, then make the new
			Fungsi **'append'** yang digunakan untuk menambahkan node baru di akhir linked list.

		node as head
		if (*head_ref == NULL) {
		new_node->prev = NULL;
		*head_ref = new_node;
		return;
		}
		// 5. Else traverse till the last node
		Node *last = *head_ref;
		while (last->next != NULL)
		last = last->next;
		// 6. Change the next of last node
		last->next = new_node;
		// 7. Make last node as previous of new node
		new_node->prev = last;
		}	

	27	void append(Node **head_ref, int new_data)	Mendeklarasikan variabel pointer ke pointer **'head_ref'** yang berfungsi sebagai head node dari linked list.
	28-32	// 1. Allocate node Node *new_node = (Node *)malloc(sizeof(Node)); // 2. Put in the data new_node->data = new_data;	Mengalokasikan memori untuk node baru dan mengisi data pada node tersebut.

	32-37	// 3. This new node is going to be the last node, so make next of it as NULL
		new_node->next = NULL;
		// 4. If the Linked List is empty, then make the new node as head
		if (*head_ref == NULL) {
		new_node->prev = NULL;
		*head_ref = new_node;
			Mengatur pointer **'next'** pada node baru sebagai NULL karena node baru akan diletakkan di akhir linked list.

	45	last->next = new_node;	Pointer 'next' pada node terakhir diubah menjadi node baru.

4	13-23	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 ini tidak dapat dijalankan jika tidak ada declarasi #include <stdlib.h>

	68-80	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;
		}
			Driver program yang digunakan untuk menguji fungsi-fungsi yang telah didefinisikan sebelumnya.

	72-74	push(&head, 7);
		push(&head, 1);
		push(&head, 4);
			Fungsi **‘push’** tiga kali untuk memasukkan data dengan nilai 7, 1, dan 4 ke dalam linked list.

	75	insertBefore(&head, head->next, 8);	Fungsi **‘insertBefore’** untuk menyisipkan node baru dengan nilai 8 sebelum node kedua dalam linked list.

	78	getchar();	Fungsi **‘getchar()’** untuk menunggu pengguna menekan tombol enter sebelum program selesai dieksekusi.