REVIEW JOBSHEET 4 "DOUBLY LINKED LIST"



Dosen pengampu: Randi Proska Sandra, M.Sc

Kode Kelas : 202323430157

Disusun Oleh:

Muhammad Devin Rahadi 23343076

PROGRAM STUDI INFORMATIKA (NK) FAKULTAS TEKNIK UNIVERSITAS NEGERI PADANG 2024

No Program	Baris Program	Petikan Source Code	Penjelasan
1	5-10	<pre>struct Node { int data; struct Node *next; struct Node *prev; };</pre>	Deklarasi struktur baru dengan nama node(simpul). Next dan prev adalah variable pointer yang akan digunakan untuk mengarah ke simpul sebelum atau setelah sebuah simpul baru dibuat
1	1-2	<pre>#include <stdio.h> #include <stdlib.h></stdlib.h></stdio.h></pre>	#include <stdio.h>: salah satu library yang ada pada Bahasa C, digunakan untuk menyertakan file standar input-output, seperti "printf" #include <stdlib.h>: library yang digunakan untuk alokasi memori, proses control, dan generate angka acak</stdlib.h></stdio.h>
1	10-25	<pre>void push(struct Node** head_ref, int new_data) { /* 1. allocate node */ struct Node* new_node = (struct Node*)malloc(sizeof(struct Node)); /* 2. put in the data */ new_node->data = new_data; /* 3. Make next of new node as head and previous as NULL */ new_node->next = (*head_ref); new_node->prev = NULL; /* 4. change prev of head node to new node */ if ((*head_ref) != NULL) (*head_ref)->prev = new_node; /* 5. move the head to point to the new node */</pre>	void push(struct Node** head_ref, int new_data): Ini adalah deklarasi fungsi push. Fungsi ini mengambil dua parameter: head_ref, yang merupakan pointer ke pointer menuju kepala (head) dari linked list, dan new_data, yang merupakan data yang akan dimasukkan ke dalam node baru. struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));: Baris ini mengalokasikan memori untuk node baru menggunakan fungsi malloc(). Ukuran alokasi memori sesuai dengan ukuran struktur struct Node. Fungsi malloc() mengembalikan alamat

		(*hoad nof) = nov mode:	memori dari node baru
		(*head_ref) = new_node;	
		}	yang telah dialokasikan.
			new_node->data =
			new_data;: Baris ini
			menetapkan data baru
			(new_data) ke dalam node
			yang baru saja
			dialokasikan.
			new_node->next =
			(*head_ref);: Ini mengatur
			pointer next dari node baru
			untuk menunjuk ke node
			yang saat ini menjadi
			kepala dari linked list
			(yang ditunjuk oleh
			head_ref).
			new_node->prev =
			NULL;: Karena node baru
			akan menjadi kepala dari
			linked list, maka pointer
			prev dari node baru diatur
			menjadi NULL,
			menandakan bahwa tidak
			ada node sebelumnya.
			if ((*head_ref) != NULL)
			(*head ref)->prev =
			new_node;: Ini memeriksa
			apakah linked list tidak
			kosong. Jika tidak kosong,
			maka pointer prev dari
			node yang saat ini menjadi
			kepala linked list (yang
			ditunjuk oleh head_ref)
			diubah untuk menunjuk ke
			node baru.
			(*head_ref) = new_node;:
			Akhirnya, pointer head_ref
			diubah untuk menunjuk ke
			node baru, menjadikannya
			sebagai kepala baru dari
4	26.46		linked list.
1	26-40	void printList(struct	Source code ini merupakan
		Node* node)	implementasi dari fungsi
		{	printList yang digunakan
		struct Node* last;	untuk mencetak isi dari
		printf("\nTraversal in	suatu linked list. Fungsi ini
		forward direction \n");	menerima parameter

		<pre>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; } }</pre>	berupa pointer ke node pertama dari linked list. Pertama, fungsi mencetak isi linked list secara berurutan dari node pertama hingga terakhir dengan menggunakan perulangan while yang berjalan selama node tidak NULL. Setiap kali iterasi, nilai data dari node saat ini dicetak, kemudian pointer node digeser ke node berikutnya. Setelah selesai mencetak isi linked list secara berurutan, fungsi melakukan pencetakan ulang dari belakang ke depan. Ini dilakukan dengan memanfaatkan pointer last yang telah disimpan saat iterasi pertama. Pencetakan dimulai dari last dan berlanjut mundur ke node sebelumnya menggunakan pointer prev. Proses ini terus dilakukan hingga pointer last menjadi NULL. Jadi, fungsi ini mencetak isi linked list dua kali, pertama dari awal ke akhir
			pertama dari awal ke akhir, dan kedua dari akhir ke awal.
2	1-2	<pre>#include <stdio.h> #include <stdlib.h></stdlib.h></stdio.h></pre>	#include <stdio.h>: Mendefinisikan pustaka standar untuk input-output dalam bahasa C, yang berisi fungsi-fungsi seperti printf dan scanf. #include <stdlib.h>: Mendefinisikan pustaka standar untuk fungsi- fungsi umum dalam bahasa C, termasuk alokasi dan</stdlib.h></stdio.h>

			dealokasi memori dinamis
			dengan malloc dan free.
2	4-9	// Structure of the node	data: Variabel bertipe int
		struct Node	yang menyimpan nilai data
		{	yang akan disimpan dalam
		int data;	node.
		struct Node *next; //	next: Pointer yang
		Pointer to next node	menunjuk ke node
		struct Node *prev; //	berikutnya dalam linked
		Pointer to previous node	list. Dalam konteks ini,
		};	linked list adalah
			rangkaian node-node yang
			terhubung satu sama lain.
			prev: Pointer yang menunjuk ke node
			sebelumnya dalam linked
			list. Ini digunakan dalam
			implementasi linked list
			dua arah (doubly linked
			list), di mana setiap node
			memiliki koneksi ke node
			sebelumnya dan
			sesudahnya.
2	29-35	void insertAfter(struct	prev_node: Pointer ke
		Node* prev_node, int	node sebelumnya, setelah
		new_data)	posisi mana node baru
		{	akan disisipkan. new data: Data yang akan
		/*1. check if the	disimpan di dalam node
		<pre>given prev_node is NULL */ if (prev node ==</pre>	baru yang akan disisipkan.
		NULL) {	our a yang akan aisisipkan.
		printf("the	
		given previous node cannot	
		be NULL");	
		return;	
2	50-64	void printList(struct	Pertama, variabel last dari
		Node* node)	tipe struct Node*
		{	dideklarasikan untuk
		struct Node* last;	menyimpan pointer ke
		printf("\nTraversal	node terakhir dari linked
		in forward direction \n");	list.
		while (node != NULL)	Fungsi kemudian
		{	mencetak isi linked list
		printf(" %d ", node-	secara berurutan dari awal
		>data);	ke akhir dengan
		last = node;	perulangan while. Dalam perulangan tersebut, nilai
		<pre>node = node->next;</pre>	permangan tersebut, miai

```
data dari setiap node
                        }
                               printf("\nTraversal
                                                          dicetak, kemudian pointer
                       in reverse direction \n");
                                                          node digeser ke node
                              while (last != NULL)
                                                          berikutnya.
                                                          Setelah selesai mencetak
                       {
                              printf(" %d ", last-
                                                          isi linked list secara
                                                          berurutan, fungsi
                       >data);
                                                          melakukan pencetakan
                              last = last->prev;
                                                          ulang dari belakang ke
                        }
                                                          depan. Ini dilakukan
                       }
                                                          dengan menggunakan
                                                          variabel last yang telah
                                                          disimpan saat iterasi
                                                          pertama. Pencetakan
                                                          dimulai dari last dan
                                                          berlanjut mundur ke node
                                                          sebelumnya menggunakan
                                                          pointer prev.
                                                          Proses ini terus dilakukan
                                                          hingga pointer last menjadi
                                                          NULL, sehingga semua
                                                          node telah dicetak dari
                                                          akhir ke awal.
           65-77
2
                       int main()
                                                          Pertama, sebuah pointer
                                                          head ke node pertama dari
                                                          linked list dideklarasikan
                              /* Start with the
                                                          dan diinisialisasi sebagai
                       empty list */
                                                          NULL. Ini menandakan
                              struct Node* head =
                                                          bahwa linked list masih
                       NULL;
                                                          kosong saat awalnya.
                              push(&head, 6);
                                                          Kemudian, beberapa
                              push(&head, 5);
                                                          operasi dilakukan untuk
                              push(&head, 2);
                                                          mengubah linked list:
                              insertAfter(head-
                                                          Tiga panggilan fungsi push
                       >next, 5);
                                                          digunakan untuk
                              printf("Created DLL
                                                          menambahkan node-node
                       is: ");
                                                          baru ke depan linked list.
                              printList(head);
                                                          Angka 6, 5, dan 2
                              getchar();
                                                          ditambahkan ke linked list
                              return 0;
                                                          secara berurutan, sehingga
                       }
                                                          linked list akan memiliki
                                                          urutan 2, 5, 6.
                                                          Panggilan fungsi
                                                          insertAfter digunakan
                                                          untuk menyisipkan nilai 5
                                                          setelah node kedua (node
                                                          dengan nilai 5).
```

			Setelah itu, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga pengguna menekan tombol Enter sebelum program berakhir. Nilai 0 dikembalikan untuk menandakan bahwa program telah berjalan dengan sukses dan berakhir tanpa ada masalah.
3	1-2	<pre>#include <stdio.h> #include <stdlib.h></stdlib.h></stdio.h></pre>	#include <stdio.h>: Mendefinisikan pustaka standar untuk input-output dalam bahasa C, yang berisi fungsi-fungsi seperti printf dan scanf. #include <stdlib.h>: Mendefinisikan pustaka standar untuk fungsi- fungsi umum dalam bahasa C, termasuk alokasi dan dealokasi memori dinamis dengan malloc dan free.</stdlib.h></stdio.h>
3	5-9	<pre>// Structure of the node struct Node { int data; struct Node *next; // Pointer to next node struct Node *prev; // Pointer to previous node };</pre>	data: Variabel bertipe int yang menyimpan nilai atau data yang ingin disimpan dalam simpul (node) linked list. next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul

			(node) sebelumnya dalam linked list. Penggunaan pointer ini umumnya terdapat pada linked list tipe doubly linked list, di mana setiap simpul memiliki koneksi ke simpul sebelumnya dan sesudahnya.
3	49-62	<pre>void printList(struct Node *node) { struct Node *last = NULL; 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; } }</pre>	Pertama, sebuah pointer last dari tipe struct Node* dideklarasikan dan diinisialisasi sebagai NULL. Pointer ini akan digunakan untuk menyimpan alamat dari node terakhir dalam linked list. Fungsi kemudian mencetak isi linked list secara berurutan dari awal ke akhir dengan menggunakan perulangan while. Dalam perulangan tersebut, nilai data dari setiap node dicetak menggunakan printf, kemudian pointer last diupdate untuk menunjuk ke node saat ini, dan pointer node digeser ke node berikutnya. Setelah selesai mencetak isi linked list secara berurutan, fungsi melakukan pencetakan ulang dari belakang ke depan. Ini dilakukan dengan menggunakan variabel last yang telah disimpan saat iterasi pertama. Pencetakan dimulai dari last dan berlanjut mundur ke node sebelumnya menggunakan pointer prev.

			Proses ini terus dilakukan
			hingga pointer last menjadi
			NULL, sehingga semua
			node telah dicetak dari
			akhir ke awal.
3	64-79	<pre>int main() {</pre>	Pertama, sebuah pointer
		// Start with the	head dari tipe struct Node*
		empty list	dideklarasikan dan
		struct Node *head =	diinisialisasi sebagai
		NULL;	NULL. Ini menandakan
		// Insert 6. So linked	bahwa linked list masih
		list becomes 6->NULL	kosong saat awalnya.
			Kemudian, beberapa
		append(&head, 6);	operasi dilakukan untuk
		// Insert 7 at the	mengubah linked list:
		beginning. So linked list	Panggilan fungsi append
		becomes 7->6->NULL	digunakan untuk
		push(&head, 7);	menambahkan node
		// Insert 1 at the	dengan nilai 6 ke akhir
		beginning. So linked list	-
		becomes 1->7->6->NULL	linked list. Sehingga linked
		<pre>push(&head, 1);</pre>	list akan menjadi 6-
		// Insert 4 at the	>NULL.
		end. So linked list	Panggilan fungsi push
		becomes 1->7->6->4->NULL	digunakan untuk
		append(&head, 4);	menambahkan node
		printf("Created DLL	dengan nilai 7 di awal
		is: ");	linked list. Sehingga linked
		<pre>printList(head);</pre>	list akan menjadi 7->6-
		<pre>getchar();</pre>	>NULL.
		return 0;	Panggilan fungsi push
		}	kembali digunakan untuk
			menambahkan node
			dengan nilai 1 di awal
			linked list. Sehingga linked
			list akan menjadi 1->7->6-
			>NULL.
			Panggilan fungsi append
			kembali digunakan untuk
			menambahkan node
			dengan nilai 4 di akhir
			linked list. Sehingga linked
			list akan menjadi 1->7->6-
			>4->NULL.
			Setelah semua operasi
			penambahan selesai
			dilakukan, pesan "Created
			DLL is: " dicetak sebagai
<u> </u>	I	l	

			penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga pengguna menekan tombol Enter sebelum program berakhir. Nilai 0 dikembalikan untuk menandakan bahwa program telah berjalan dengan sukses dan berakhir tanpa ada masalah.
3	11-24	<pre>void push(struct Node **head_ref, int new_data) { // Allocate node struct Node *new_node = (struct Node *)malloc(sizeof(struct Node)); // Put in the data new_node->data = new_data; // Make next of new node as head and previous as NULL new_node->next = *head_ref; new_node->prev = NULL; // Change prev of head node to new node if (*head_ref != NULL)</pre>	Pertama, fungsi ini melakukan alokasi memori untuk node baru menggunakan fungsi malloc, sehingga variabel new_node menunjuk ke alamat memori yang baru dialokasikan. Selanjutnya, nilai new_data dimasukkan ke dalam variabel data pada node baru yang telah dialokasikan. Kemudian, pointer next dari node baru diarahkan ke node pertama (head) dari linked list yang sudah ada sebelumnya. Pointer prev diatur sebagai NULL karena node baru akan menjadi node pertama, sehingga tidak memiliki node sebelumnya. Setelah itu, langkahlangkah untuk mengubah status node pertama dan pointer head_ref dilakukan: Jika linked list tidak kosong (pointer head_ref tidak NULL), maka

i contract of the contract of	I		
			pointer prev dari node
			pertama diubah menjadi
			menunjuk ke node baru,
			karena node baru akan
			menjadi node pertama.
			Pointer head_ref diarahkan
			untuk menunjuk ke node
			baru yang telah
			ditambahkan, sehingga
			node baru menjadi node pertama dalam linked list.
4	1-2	#include <stdio.h></stdio.h>	#include <stdio.h>:</stdio.h>
4	1-2	#include <stdio.n></stdio.n>	Mendefinisikan pustaka
		#Include (Stallb.ii)	standar untuk input-output
			dalam bahasa C, yang
			berisi fungsi-fungsi seperti
			printf dan scanf.
			#include <stdlib.h>:</stdlib.h>
			Mendefinisikan pustaka
			standar untuk fungsi-
			fungsi umum dalam bahasa
			C, termasuk alokasi dan
			dealokasi memori dinamis
			dengan malloc dan free.
4	4-9	// Structure of the node	data: Variabel bertipe int
		struct Node	yang menyimpan nilai atau
		{	data yang ingin disimpan
		int data;	dalam simpul (node)
		ation at Nada Visavita //	linked list.
		struct Node *next; //	
		Pointer to next node	next: Pointer yang
		Pointer to next node struct Node *prev; //	next: Pointer yang menunjuk ke simpul
		Pointer to next node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam
		Pointer to next node struct Node *prev; //	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini,
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list.
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam linked list. Penggunaan
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam linked list. Penggunaan pointer ini umumnya
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam linked list. Penggunaan pointer ini umumnya terdapat pada linked list
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam linked list. Penggunaan pointer ini umumnya terdapat pada linked list tipe doubly linked list, di
		Pointer to next node struct Node *prev; // Pointer to previous node	next: Pointer yang menunjuk ke simpul (node) berikutnya dalam linked list. Dengan menggunakan pointer ini, kita bisa melakukan traversal atau perjalanan ke simpul berikutnya dalam linked list. prev: Pointer yang menunjuk ke simpul (node) sebelumnya dalam linked list. Penggunaan pointer ini umumnya terdapat pada linked list

			simpul sebelumnya dan
			sesudahnya.
4	66-78	<pre>int main() { /* Start with the empty list */ 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; }</pre>	Fungsi main: Fungsi ini merupakan titik masuk (entry point) dari program. Pertama, pointer head diinisialisasi dengan nilai NULL, menandakan bahwa linked list kosong. Kemudian, tiga node baru dengan nilai data 6, 5, dan 2 dimasukkan ke awal linked list menggunakan fungsi push. Selanjutnya, node baru dengan nilai data 5 dimasukkan setelah node kedua (dengan nilai data 5) menggunakan fungsi insertAfter. Akhirnya, isi dari linked list dicetak menggunakan fungsi printList.
4	11	<pre>void push(struct Node** head_ref, int new_data)</pre>	Fungsi push: Fungsi ini beroperasi pada doubly linked list. Parameternya adalah: head_ref: Pointer ke pointer yang menunjuk ke head (awal) dari doubly linked list. new_data: Sebuah bilangan bulat yang mewakili nilai data dari node baru yang akan dimasukkan.
4	51-65	<pre>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");</pre>	Fungsi printList: Fungsi ini beroperasi pada doubly linked list. Parameternya adalah: node: Pointer ke struct Node, yang merupakan head (awal) dari doubly linked list.

printf(" %d ", last->data); last = last->prev; } } 1		T	T	
A			<pre>last = last->prev;</pre>	
/* Start with the empty list */ struct Node* head = NULL; push(&head, 6); push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } // Start with the empty list */ struct Node* head = NULL; push(&head, 6); push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } // Start with the empty diinisialisasi sebagai NULL. Ini menandakan bahwa linked list masih kosong saat awalnya. // Kemudian, beberapa operasi dilakukan untuk menambahkan node dengan nilai 6 di awal linked list. Sehingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Schingga linked list akan menjadi 2->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Schingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga	1	41-53		_
list */ struct Node* head = NULL; push(&head, 6); push(&head, 5); push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } freturn 0; } substant				1
struct Node* head = NULL; push(&head, 6); push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } struct Node* head = NULL; push(&head, 6); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } struct Node* head = NULL; push(&lead, 6); push(&lead, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } struct Node* head = NULL; printf("Created DLL is: "list akan linked list: pangilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Schingga linked list akan menjadi 5->6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Schingga linked list akan menjadi 2->5->6->NULL. Setclah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: "dicetak sebagai penanda, yang dilkuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
push(&head, 6); push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } freturn 0; } managilan fungsi push dengan nilai 6 di awal linked list. Schingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Schingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Schingga linked list. Schingga linked list. Schingga linked list. Schingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang dilkuti oleh pemanggilan fungsi printList untuk menetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			· · · · · · · · · · · · · · · · · · ·	_
push(&head, 5); push(&head, 2); printf("Created DLL is: "); printList(head); getchar(); return 0; } return 0; } push(&head, 2); printList(head); getchar(); return 0; } return 0; } linked list. Schingga linked list & kembali digunakan untuk menambahkan node dengan nilai 6 di awal linked list. Schingga linked list akan menjadi 6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6->NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang dilkuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			-	bahwa linked list masih
printf("Created DLL is: "); printList(head); getchar(); return 0; } return 0; } return 0; } printList(head); getchar(); return 0; } return 0; printList(head); getchar(); printList(head); getchar(); printList(head); getchar() igunakan untuk menambahkan node dengan nilai 6 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				kosong saat awalnya.
mengubah linked list: printList(head); getchar(); return 0; linked list. Sehingga linked list akan menjadi 6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6->NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			<pre>push(&head, 2);</pre>	_
printList(head); getchar(); return 0; } Panggilan fungsi push digunakan untuk menambahkan node dengan nilai 6 di awal linked list. Sehingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				_
getchar(); return 0; } linked list. Sehingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			, , ,	
return 0; } menambahkan node dengan nilai 6 di awal linked list. Schingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Schingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Schingga linked list akan menjadi 2->5->6- >NULL. Schingga linked list akan menjadi 2->5->6- >NULL. Schingga linked list akan menjadi 2->5->6- >NULL. Schingga linked list akan menjadi 2->5->6- >NULL schingga linked list akan menjadi 5->6- >NULL schingga linked list akan menj				
return 0; dengan nilai 6 di awal linked list. Sehingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			getchar();	
linked list. Sehingga linked list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			return 0:	
list akan menjadi 6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			_	
Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6->NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga			,	list akan menjadi 6-
kembali digunakan untuk menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6->NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6->NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
menambahkan node dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
dengan nilai 5 di awal linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				_
linked list. Sehingga linked list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
list akan menjadi 5->6- >NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
>NULL. Panggilan fungsi push kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
kembali digunakan untuk menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
menambahkan node dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
dengan nilai 2 di awal linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				_
linked list. Sehingga linked list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
list akan menjadi 2->5->6- >NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
>NULL. Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
Setelah semua operasi penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				<u> </u>
penambahan selesai dilakukan, pesan "Created DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
DLL is: " dicetak sebagai penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				_
penanda, yang diikuti oleh pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				-
pemanggilan fungsi printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
printList untuk mencetak isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				1
isi dari linked list yang telah dimodifikasi. getchar() digunakan untuk menunggu hingga				
telah dimodifikasi. getchar() digunakan untuk menunggu hingga				1 *
getchar() digunakan untuk menunggu hingga				I
menunggu hingga				
				u u
11 00				pengguna menekan tombol

	Enter sebelum program
	berakhir.
	Nilai 0 dikembalikan untuk
	menandakan bahwa
	program telah berjalan
	dengan sukses dan
	berakhir tanpa ada
	masalah.