**Edwin Ario Abdiwijaya**

[**edwin.abdiwijaya@binus.ac.id**](mailto:edwin.abdiwijaya@binus.ac.id)

**Answer sheet for assignment 3**

// Edwin Ario Abdiwijaya

// edwin.abdiwijaya@binus.ac.id

// Answer for question number 1

#include <stdio.h>

#include <stdlib.h>

struct Node

{

    int data;

    struct Node \*next;

};

int idx = 0;

Node \*getNode(int data)

{

    Node \*newNode = new Node();

    newNode->data = data;

    newNode->next = NULL;

    return newNode;

}

void insert\_any(Node \*\*current, int basho, int data)

{

    if (basho < 0 || basho > idx + 1)

    {

        printf("Invalid Position\n");

    }

    else

    {

        while (basho--)

        {

            if (basho == 0)

            {

                Node \*temp = getNode(data);

                temp->next = \*current;

                \*current = temp;

            }

            else

            {

                current = &(\*current)->next;

            }

        }

        idx++;

    }

}

void print(struct Node\* head) {

    while (head != NULL) {

        printf(" %d",head->data);

        head = head->next;

    }

    printf("\n");

}

int main()

{

    Node \*head = NULL;

    head = getNode(1);

    head->next = getNode(5);

    head->next->next = getNode(9);

    head->next->next->next = getNode(11);

    head->next->next->next->next = getNode(14);

    idx = 5;

    printf("Linked list before insertion: ");

    print(head);

    printf("\n\n");

    int totalInsert, i, data, basho;

    printf("How many times do you want to insert?\n");

    printf(">> ");

    scanf("%d", &totalInsert);

    getchar();

    printf("\n");

    for (i = 0; i < totalInsert; i++)

    {

        printf("Enter the number value you want to input : ");

        scanf("%d", &data);

        getchar();

        printf("Where is the position you want to insert the number? (Start from index 0)\n");

        printf(">> ");

        scanf("%d", &basho);

        getchar();

        insert\_any(&head, basho + 1, data);

        printf("Linked list after insertion of %d at position %d: ", data, basho);

        print(head);

        printf("\n\n\n");

    }

    return 0;

}

// Edwin Ario Abdiwijaya

// edwin.abdiwijaya@binus.ac.id

// Answer for question number 2

#include <stdio.h>

#include <stdlib.h>

struct Node

{

    int data;

    struct Node \*next;

};

Node \*delete\_beg(struct Node \*head)

{

    if (head == NULL)

    {

        return NULL;

    }

    Node \*temp = head;

    head = head->next;

    delete temp;

    return head;

}

void push(struct Node \*\*head\_re, int new\_data)

{

    struct Node \*new\_node = new Node;

    new\_node->data = new\_data;

    new\_node->next = (\*head\_re);

    (\*head\_re) = new\_node;

}

int main()

{

    Node \*head = NULL;

    push(&head, 1);

    push(&head, 3);

    push(&head, 5);

    push(&head, 7);

    push(&head, 9);

    printf("First Deletion:\n");

    printf("Linked List before delete the beginning:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n");

    head = delete\_beg(head);

    printf("Linked List after delete the beginning:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n\n");

    printf("Second Deletion: \n");

    printf("Linked List before delete the beginning:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n");

    head = delete\_beg(head);

    printf("Linked List after delete the beginning:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n\n");

    return 0;

}

// Edwin Ario Abdiwijaya

// edwin.abdiwijaya@binus.ac.id

// Answer for question number 3

#include <stdio.h>

#include <stdlib.h>

struct Node

{

    int data;

    struct Node \*next;

};

Node\* delete\_end(struct Node \*head)

{

    if (head == NULL)

        return NULL;

    if (head->next == NULL)

    {

        delete head;

        return NULL;

    }

    Node \*secol = head;

    while (secol->next->next != NULL)

        secol = secol->next;

    delete (secol->next);

    secol->next = NULL;

    return head;

}

void push(struct Node \*\*head\_re, int new\_data)

{

    struct Node \*new\_node = new Node;

    new\_node->data = new\_data;

    new\_node->next = (\*head\_re);

    (\*head\_re) = new\_node;

}

int main()

{

    Node \*head = NULL;

    push(&head, 1);

    push(&head, 3);

    push(&head, 6);

    push(&head, 7);

    push(&head, 9);

    printf("First deletion: \n");

    printf("Linked List before delete the last:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n");

    head = delete\_end(head);

    printf("Linked List after delete the last:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n\n");

    printf("Second deletion: \n");

    printf("Linked List before delete the last:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n");

    head = delete\_end(head);

    printf("Linked List after delete the lasts:");

    for (Node \*temp = head; temp != NULL; temp = temp->next)

    {

        printf(" %d", temp->data);

    }

    printf("\n\n");

    return 0;

}