****

**TAYLOR’S UWE DUAL AWARDS PROGRAMMES**

Data Structures and Algorithms (ITS60504)

**March 2023 SEMESTER**

**Individual Practical Test Question (10%)**

**Exam: 1hr 30mins**

|  |  |
| --- | --- |
| Name | Dev Mani Maharjan |
| StudentId | 2211127 |
| UniversityID | 0355610 |

|  |  |
| --- | --- |
| **Question No** | **Marks Obtained** |
| 1 | **/5** |
| 2 | **/5** |
| 3 | **/10** |
| Total Marks | **/20** |

Contents

[Question no 1. 3](#_Toc135126152)

[Code of question no 1: 3](#_Toc135126153)

[Output of question no1: 4](#_Toc135126154)

[Question no 2 5](#_Toc135126155)

[Question no 2 code: 5](#_Toc135126156)

[Output of Question no 2: 6](#_Toc135126157)

[Question no 3: 7](#_Toc135126158)

[Code of question no 3: 7](#_Toc135126159)

[Output Of Question no 3: 8](#_Toc135126160)

# Question no 1.

## Code of question no 1:

#include <stdio.h>

int top = -1;

char stacks [3];

void push (char number){

if (top==2){

    printf("%c cannot be inserted into a stack as stack is full.\n",number);

}

else{

top++;

stacks [top] = number;

printf ("%c is inserted into stack.\n",number);

}

}

char pop (){

    if (top==-1){

        printf("Stack is empty.\n");

    }

    else{

        char r\_number=stacks[top];

    top--;

    printf ("%c",r\_number);

    }

}

int main(int argc, char const \*argv[])

{

push('D');

push('A');

push('D');

push('D');

    printf ("Its reverse form is(Pop operation): \n");

pop();

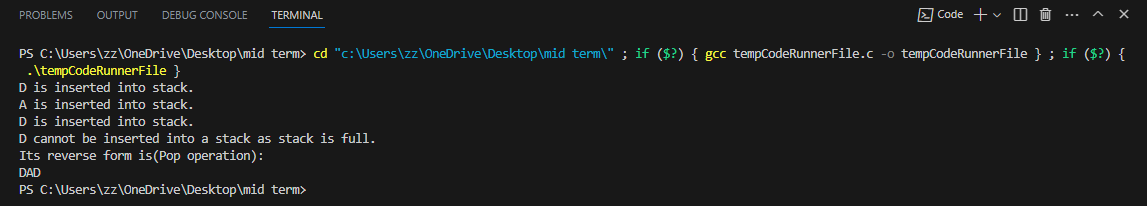
pop();

pop();

    return 0;

}

## Output of question no1:



# Question no 2

## Question no 2 code:

#include<stdio.h>

char \*queues[3];

int rear = -1;

int front = -1;

void enqueue(char element[]){

    if (rear==2){

        printf("%s cannot be inserted into the queue. The queue is full.\n",element);

    }else{

        rear++;

        queues[rear] = element;

        printf("%s \n",queues[rear]);

    }

}

void dequeue(){

    if(front == rear){

        printf("Dequeue operation cannot be performed because queue is empty.");

    }

    else{

        front++;

        printf("%s is displayed on the screen. \n",queues[front]);

    }

}

int main(int argc, char const \*argv[])

{

    printf("Nabil Bank bank counter queue \n");

    enqueue("Ticket No 1");

    enqueue("Ticket No 2");

    enqueue("Ticket No 3");

    printf("-----------------------\n");

    printf("Dequeue operation.....\n");

    dequeue();

    dequeue();

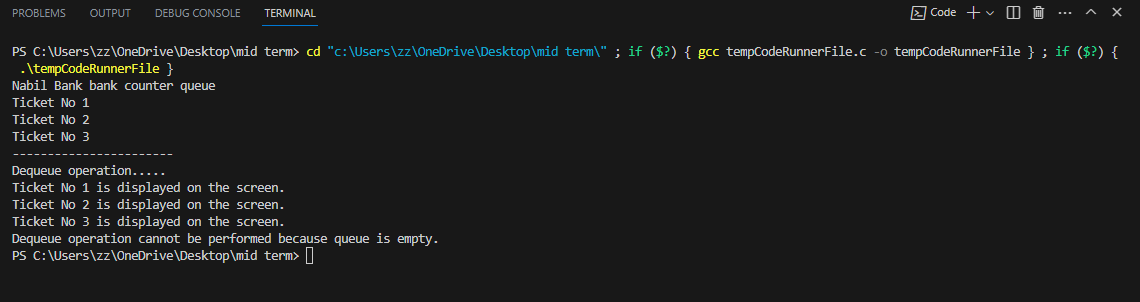
    dequeue();

    dequeue();

    return 0;

}

## Output of Question no 2:



# Question no 3:

## Code of question no 3:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct Node

{

    char data[50];

    struct Node \*nextNode;

}

node;

node \*firstNode;

node \*newNode;

void insertfromlast(char data[])

{

    newNode = malloc(sizeof(node));

    strcpy(newNode->data, data);

    newNode->nextNode = NULL;

    if (firstNode == NULL)

    {

        firstNode = newNode;

    }

    else{

        node \*uNode = firstNode;

        while (uNode->nextNode != NULL)

        {

            uNode = uNode->nextNode;

        }

            uNode->nextNode = newNode;

    }

    printf("%s\n", newNode->data);

}

void traverse(){

    node \*node = firstNode;

    while (node->nextNode != NULL)

    {

        printf("%s =>", node->data);

        node = node->nextNode;

    }

    printf("%s\n", node->data);

}

void removeFromLast(){

    if (firstNode == NULL)

    {

        printf("List is empty\n");

        return;

    }

    node \*nodetodelete = firstNode;

    firstNode = firstNode->nextNode;

    nodetodelete->nextNode = NULL;

}

int main(int argc, char const \*argv[]){

    printf("Insertion Operation in the linkedlist.....\n");

    insertfromlast("Wash Face");

    insertfromlast("Drink Coffee");

    insertfromlast("Code");

    printf("Traversing operation in the linkedlist....\n");

    traverse();

    printf("Deletion operation from the linkedlist....\n");

    removeFromLast();

    traverse();

    return 0;

}

## Output Of Question no 3:

