**דוח מעבדה 8**

**שמות מגישים:**

סער ויקטור – 312392822

אילון בן סימון – 312162951

**תרגיל 1**

קובץ header:

#ifndef Targil1

#define Targil1

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

//structures definition

typedef struct Item {

int num;

struct Item\* next;

}Item;

typedef struct Stack {

Item\* head;

int size; //a current number of items

}Stack, \*PStack;

//functions declaration

void Push(PStack s, int new\_elem); //add a new member to list of the stack

int Pop(PStack s, int \* del\_value); //delete member from the stack and return the deleted value using int \* del\_value

void Err\_Msg(char \*str);

#define NUM 5

#endif

קובץ main:

#include "Targil1.h"

int main()

{

Stack s;

int i;

int del\_val;

int num[NUM] = { 1,2,3,4,5 };

s.head = NULL;

s.size = 0;

for (i = 0; i < NUM; i++) //filling the stack

{

if (NUM == s.size)

{

printf("The stack is full!");

break;

}

Push(&s, num[i]);

}

while (Pop(&s, &del\_val)==1) //popping out from the stack

{

printf("The deleted value is: %d\n", del\_val);

}

printf("The stack is now empty!");

getch();

return 0;

}

קובץ מימוש:

#include "Targil1.h"

//add a new member to list of the stack

void Push(PStack s, int new\_elem)

{

Item \*temp;

s->size++; //counting how many nodes in the linked list

temp = (Item\*)malloc(sizeof(Item));

if (temp == NULL)

{

Pop(s, &new\_elem);

Err\_Msg("Memory!");

}

temp->num = new\_elem; //initializing the new node

temp->next = s->head; //linking

s->head = temp; //linking

}

//delete member from the stack and return the deleted value using int \* del\_value

int Pop(PStack s, int \* del\_value)

{

Item\* temp;

if (s->head != NULL) //delete the node

{

\*del\_value = s->head->num;

temp = s->head;

s->head = s->head->next;

free(temp);

return 1;

}

return 0;

}

void Err\_Msg(char \*str)

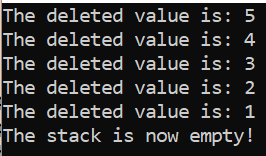
{

printf("%s\n", str);

exit(1);

}

פלט:



**תרגיל 2**

קובץ header:

#ifndef Targil2

#define Targil2

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

//structure defintion

typedef struct Stack {

int top; //index of the last member in array of a stack;

int\* Array; //pointer to array of members of a stack

int size; //size of the array of a stack

int count; //the current number of members

}Stack, \*PStack;

//functions declaration

void InitStack(PStack s, int size); //initialization of a new stack with capacity of size elements

void Push(PStack s, int new\_elem); //add a new member to array of the stack

int Pop(PStack s, int \* del\_value); //delete member from the array of the stack and return the deleted value using int \* del\_value

void Err\_Msg(char \*str);

#endi

קובץ main:

#include "Targil2.h"

int main()

{

Stack st;

int del\_value, flag;

InitStack(&st, 4);

Push(&st, 3);

Push(&st, 5);

Push(&st, 1);

Push(&st, 10);

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

Push(&st, 8);

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

Push(&st, 7);

Push(&st, 4);

Push(&st, 100);

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

while (st.count)

{

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

}

printf("\n");

flag = Pop(&st, &del\_value);

if (flag)

printf("\n%d was deleted", del\_value);

free(st.Array);

getch();

return 0;

}

קובץ מימוש:

#include "Targil2.h"

//the function initializing the stack

void InitStack(PStack s, int size)

{

s->Array = (int\*)malloc(sizeof(int)\*size);

if (s->Array == NULL)

Err\_Msg("Memory");

s->size = size;

s->count = 0;

s->top = -1;

}

//add a new member to the array

void Push(PStack s, int new\_elem)

{

if (s->count < s->size)

{

s->top++;

s->count++;

s->Array[s->top] = new\_elem;

}

else

printf("\nThe Stack is full");

}

//delete member from the array and return the deleted value using int \* del\_value

int Pop(PStack s, int \* del\_value)

{

\*del\_value = 0;

if (s->count > 0)

{

\*del\_value = s->Array[s->top];

s->top--;

s->count--;

}

else

printf("The Stack is empty\n");

return \*del\_value;

}

void Err\_Msg(char \*str)

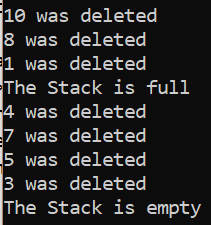
{

printf("%s\n", str);

exit(1);

}

פלט:



**תרגיל 3**

קובץ header:

#ifndef Targil3

#define Targil3

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

//structures definition

typedef struct Item {

int num;

struct Item\* next;

}Item;

typedef struct Que {

Item\* head, \*tail;

int size; //a current number of items

}Queue, \*PQue;

//functions declaration

void Enqueue(PQue Q, int new\_elem); //add a new member to list of the queue

int Dequeue(PQue Q, int \* del\_value); //delete member from the queue and return the deleted value using int \*del\_value

void Err\_Msg(char \*str);

#define NUM 5

#endif

קובץ main:

#include "Targil3.h"

int main()

{

Queue Q;

int i;

int del\_val;

int num[NUM] = { 1,2,3,4,5 };

Q.head = NULL;

Q.tail = NULL;

Q.size = 0;

for (i = 0; i < NUM; i++) //filling the queue

{

if (Q.size == NUM)

{

printf("The queue is full!\n");

break;

}

Enqueue(&Q, num[i]); //popping out from the queue

}

while (Dequeue(&Q, &del\_val) == 1)

{

printf("The deleted value is: %d\n", del\_val);

}

printf("The queue is now empty!");

getch();

return 0;

}

קובץ מימוש:

#include "Targil3.h"

//the function adding nodes to the tail of the linked list

void Enqueue(PQue Q, int new\_elem)

{

Item\* temp;

Q->size++;

temp = (Item\*)malloc(sizeof(Item));

if (temp == NULL)

{

Dequeue(&Q,&new\_elem);

Err\_Msg("Memory!");

}

temp->num = new\_elem;

temp->next = NULL;

if (Q->head == NULL) //in case the queue is empty

{

Q->head = temp;

Q->tail = temp;

}

else //in case the queue isn't empty

{

Q->tail->next = temp;

Q->tail = temp;

}

}

//the function deletes nodes from the head of the linked list

int Dequeue(PQue Q, int \* del\_value)

{

Item\* temp;

if (Q->head != NULL)

{

\*del\_value = Q->head->num;

temp = Q->head;

Q->head = Q->head->next;

free(temp);

return 1;

}

return 0;

}

void Err\_Msg(char \*str)

{

printf("%s\n", str);

exit(1);

}

פלט:

