Navlakhi®

Stacks ADT (Dynamic)

Methodology and Program

By Abhishek Navlakhi
Semester 3: Data Structures

```
#include <stdio.h>

#include <alloc.h>
#include <stdlib.h>

struct node
{
void *data;
struct node *link;
};

struct STACK
{
int count;
struct node *top;
}*stack;



void createStack()
{
stack=(struct STACK *)malloc(sizeof(struct STACK));
if (stack==NULL)
{
        printf("Insufficient Memory.... Exiting Program\n");
        exit(1);
}

/* Memory creation for head node successful */
stack ->top=NULL;
stack->count=0;
}
```

```
int pushStack(void *dataIn)
{
struct node *pNew;

pNew=(struct node *)malloc(sizeof(struct node));

if (pNew==NULL) return 0;

pNew->data=dataIn;
pNew->link=stack->top;
stack->top=pNew;
stack->count+=1;

return 1;
}

int popStack(void **dataPtr)
{
 struct node *pLoc;

 if(stack->count==0)  /* OR stack->top==NULL*/
 return 0;

 pLoc=stack->top;
 *dataPtr=pLoc->data;
 stack->top=pLoc->link; /* OR stack->top=stack->top->link*/
 free(pLoc);

 stack->count-=1;

 return 1;
}


int stackCount( )
{
 return stack->count;
}
```

```
int stackTop(void **dataIn)
{
if (stack->count!=0)
{
        *dataIn=stack->top->data;
        return 1;
}

else return 0;
}

int menu( )
{
int choice;

printf("\n\n");
printf("1. Push Data onto the Stack\n");
printf("2. Pop Data from the Stack\n");
printf("3. Check number of data items on Stack\n");
printf("4. Check the stack top element\n");
printf("5. Exit\n\n");

printf("Feed in your choice: ");
scanf("%d",&choice);

return choice;
}
```

```
/***************************
void main( )
{
int choice;
int *datain;
int **popedData,**topdata;
int success;

popedData=(int **)malloc(sizeof(int*));
topdata=(int **)malloc(sizeof(int*));

stack=createStack();

do
{
choice=menu();

switch (choice)
{
case 1: printf("feed in the data to insert onto the Stack: ");
        datain=(int *)malloc(sizeof(int));
        scanf("%d",datain);
        success=pushStack(datain);
        if (success) printf("Data Inserted Successfully\n");
        else printf("Memory Overflow\n");
        break;
case 2: success=popStack(popedData);
        if (success) printf("Data pop'ed of Stack is %d\n",**popedData);
        else printf("Underflow... no data to pop\n");
        break;
case 3:        printf("Stack count= %d\n",stackCount());
        break;
case 4: success= stackTop(topdata);
        if (success) printf("The stack top element is %d\n",**topdata);
        else printf("Satck is empty\n");
        break;
}
}while (choice !=5);
destroyStack();
}
*************************/
```

```
void main( )
{
int choice;
char *datain;
char **popedData,**topdata;
int success;

popedData=(char **)malloc(sizeof(char *));
topdata=(char **)malloc(sizeof(char *));

stack=createStack();

do
{
choice=menu();

switch (choice)
{
case 1: printf("feed in the data to insert onto the Stack: ");
        datain=(char *)malloc(sizeof(int));
        fflush(stdin);
        scanf("%c",datain);
        fflush(stdin);
        success=pushStack(datain);
        if (success) printf("Data Inserted Successfully\n");
        else printf("Memory Overflow\n");
        break;
case 2: success=popStack(popedData);
        if (success) printf("Data pop'ed of Stack is %c\n",**popedData);
        else printf("Underflow... no data to pop\n");
        break;
case 3:        printf("Stack count= %d\n",stackCount());
        break;
case 4: success= stackTop(topdata);
        if (success) printf("The stack top element is %c\n",**topdata);
        else printf("Satck is empty\n");
        break;
}
}while (choice !=5);
destroyStack();
}
```