19BIT0292

Bhaumik Tandan

LAB SHEET

DATA STRUCTURES AND ALGORITHMS LABORATORY

CSE2011

L57+L58

Q1) Implementation of stack using array
Push(), Pop(), Display()

stack.h

CODE

```
#include "./stack_header/varibles_decalred.h"//it also contains header files

#include "./stack_header/stack_functions.h"

#include "./stack_header/push_type.h"

#pragma once//restrict double import

#define push(st,a) _Generic(a, int: pushi__19BIT0292, char*:
pushs__19BIT0292,double:
pushf__19BIT0292,char:pushc__19BIT0292,float:pushf__19BIT0292)(st,a)//char and
int will be treated similarly



void s_in(stack *s)

{

    s->t__19BIT0292=-1;

    s->stack__19BIT0292=0;

    s->d_type__19BIT0292=0;

}


void menu(stack *st)

{

    void* (* fp[3])(stack *);

    //0 push

    //1 pop

    //2 top

    //3 display whole stack
```

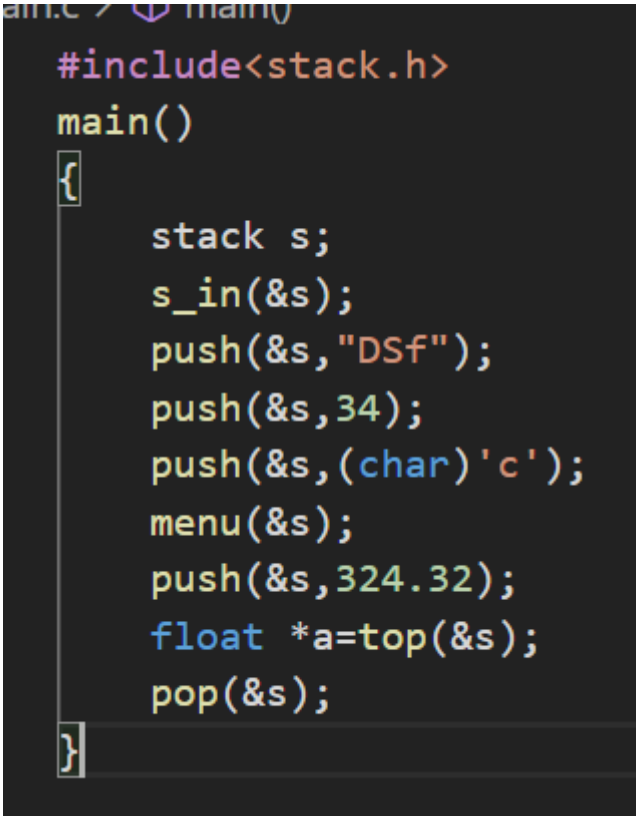
```
fp[0]=&pop;
fp[1]=&top;
fp[2]=&display;
printf("\n\n\n1)Push\n2)Pop\n3)Top\n4)Display\n5)Exit\n");
printf("\nEnter your choice: ");
int c;
scanf("%d",&c);
if(c==1)
{
    printf("\n\nEnter that you to push in the stack: ");
    char s[21];//this will get destroyed after function is finished it also has null
    scanf("%s",s);
    int a=atoi(s);//convert string to int
    float f=atof(s);
     if((a!=0 || strcmp("0",s)==0)&& f==a)
    {
        push(st,a);
        return menu(st);
    }
    if(f!=0)
    {
        push(st,f);
        return menu(st);
    }
    if(strlen(s)>1){
    push(st,s);
    }
    else
    push(st,s[0]);
```

```
        return menu(st);
    }
    else if(c==5)
    return;
    fp[c-2](st);
    return menu(st);
}
```

main.c

CODE

```
#include<stack.h>
main()
{
    stack s;
    s_in(&s);
    push(&s,"DSf");
    push(&s,34);
    push(&s,(char)'c');
    menu(&s);
    push(&s,324.32);
    float *a=top(&s);
    pop(&s);
}
```



OUTPUT

```
DSf pushed
34 pushed
c pushed


1)Push
2)Pop
3)Top
4)Display
5)Exit

Enter your choice: 1


Enter that you to push in the stack: 434.43

434.429993 pushed


1)Push
2)Pop
3)Top
4)Display
5)Exit

Enter your choice: 4

The whole stack is

|            434.4300|
|                   c|
|                  34|
|                 DSf|
|____________________|


1)Push
2)Pop
3)Top
4)Display
5)Exit

Enter your choice: 5

324.320007 pushed
Top Element is: 324.320007
324.320007 poped
```

CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE

Q2) Conversion of infix expressions to postfix expressions using stack A+B/C – D +(E^F)

CODE

```
#include <stdio.h>
#include <string.h>
char stack[50];
int top = -1;
void push(char c)
{
    top++;
    stack[top] = c;
}
char pop()
{
    char c;
    if (top == -1)
        return -1;
    else
    {
        c = stack[top];
        top--;
        return c;
    }
}
int priority(char x)
{
```

```
    if (x == '(')
        return 0;
    else if (x == '+' || x == '-')
        return 1;
    else if (x == '*' || x == '/')
        return 2;
    else if (x == '^')
        return 3;
}
main()
{
    char exp[100];
    char *p, x;
    printf("Enter the expression :: ");
    scanf("%s", exp);
    p = exp;
    while (*p != '\0')
    {
        if (isalnum(*p))
            printf("%c", *p);

        else if (*p == '(')
            push(*p);
        else if (*p == ')')
        {
            while ((x = pop()) != '(')
                printf("%c", x);
        }
```

```
        else
        {
            while (priority(stack[top]) >= priority(*p))
                printf("%c", pop());
            push(*p);
        }
        p++;
    }
    while (top != -1)
    {
        printf("%c", pop());
    }
}
```

OUTPUT

```
Enter the expression :: A+B/C-D+(E^F)
ABC/+D-EF^+
```

```
Enter the expression :: 2*4+(4*5+2)^(8*2-8)-8
24*45*2+82*8-^+8-
```

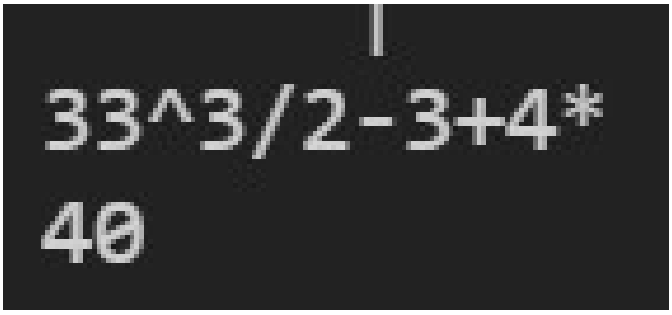
CLICK HERE FOR GITHUB LINK

Q3) Evaluation of postfix expression using stack

CODE

```
#include <stdio.h>
#include <math.h>
int s[100];
int top = -1;
push(int e)
{
    s[++top] = e;
}
int pop()
{
    return (s[top--]);
}
main()
{
    int i = 0, v1, v2;
    char c[100];
    scanf("%s", c);
    while (c[i] != '\0')
    {
        if (isdigit(c[i]))
        {
            push(c[i] - 48);
```

```
        }
        else
        {
            v1 = pop();
            v2 = pop();
            switch (c[i])
            {
            case '+':
                push(v2 + v1);
                break;
            case '-':
                push(v2 - v1);
                break;
            case '*':
                push(v2 * v1);
                break;
            case '/':
                push(v2 / v1);
                break;
            }
        }
        i++;
    }
    printf("%d", pop());
}
```



CLICK HERE FOR GITHUB LINK

Q4) Implement Queue and realize various operations to be carried out on it.

queue.h

CODE

```
#include "./queue_header/varibles_decalred.h"//it also contains header files

#include "./queue_header/queue_functions.h"

#include "./queue_header/enqueue_type.h"

#pragma once//restrict double import

#define enqueue(s,a) _Generic(a, int: enqueuei__19BIT0292, char*:
enqueues__19BIT0292,double:
enqueuef__19BIT0292,char:enqueuec__19BIT0292,float:enqueuef__19BIT0292)(s,a)

void q_in(queue *q)

{

    q->r__19BIT0292=-1;

    q->queue__19BIT0292=0;

    q->d_type__19BIT0292=0;

}

void menu(queue *q)

{

    void* (* fp[4])(queue*);

    fp[0]=&denqueue;

    fp[1]=&front;

    fp[2]=&rear;

    fp[3]=&display;

    printf("\n\n\n1)Enqueue\n2)Dequeue\n3)Front\n4)Rear\n5)Display\n6)Exit\n");

    printf("\nEnter your choice: ");
```

```
int c;
scanf("%d",&c);
if(c==1)
{
    printf("\n\nEnter that you to enqueue in the stack: ");
    char s[21];
    scanf("%s",s);
    int a=atoi(s);//convert string to int
    float f=atof(s);
     if((a!=0 || strcmp("0",s)==0)&& f==a)
    {
        enqueue(q,a);
        return menu(q);
    }
    if(f!=0)
    {
        enqueue(q,f);
        return menu(q);
    }
    if(strlen(s)>1)
    enqueue(q,s);
    else
    enqueue(q,s[0]);
    return menu(q);
}
else if(c==6)
return;
fp[c-2](q);
return menu(q);
```

```
}
```

main.c

CODE

```
#include<queue.h>
main()
{
    queue q;
    q_in(&q);
    enqueue(&q,'c');
    front(&q);
    rear(&q);
    menu(&q);
    enqueue(&q,(char)'c');
    rear(&q);
}
```

```c
main.c > main()
#include<queue.h>
main()
{
    queue q;
    q_in(&q);
    enqueue(&q,'c');
    front(&q);
    rear(&q);
    menu(&q);
    enqueue(&q,(char)'c');
    rear(&q);
}
```

OUTPUT

```
99 queued
Front Element is: 99
Rear Element is: 99


1)Enqueue
2)Dequeue
3)Front
4)Rear
5)Display
6)Exit

Enter your choice: 1
```

```
Enter that you to enqueue in the stack: 34.32

34.320000 queued


1)Enqueue
2)Dequeue
3)Front
4)Rear
5)Display
6)Exit

Enter your choice: 5

The whole queue is

            |
            |rear
            |
            v
|              34.3200|
|                   99|
           |
           |front
           |
           v


1)Enqueue
2)Dequeue
3)Front
4)Rear
5)Display
6)Exit

Enter your choice: 6

c queued
Rear Element is: c
```

CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE

Q5) Implementation of circular queue.

CODE

```
#include<stdio.h>
#define s 5
int ar[s],f=0,r=-1,e;
add(int a)
{
    if(e==s){
    printf("\nQueue Full");
    return 1;
    }
    r=(r+1)%s;
    ar[r]=a;
    e++;
    printf("\n%d added",a);
}
del()
{
    if(e==0)
    {
        printf("\nQueue Empty");
        return;
    }
    e--;
    int t=f;
    f=(f+1)%s;
    return ar[t];
}
```

```
disp()
{
    if(e==0)
    {
        printf("\nQueue Empty");
        return;
    }
    printf("\nFront ---> ");
    int i=f;
    while (i!=r)
    {
        printf("%d ",ar[i]);
        i=(i+1)%s;
    }
    printf("%d ---> Rear",ar[i]);
}
main()
{
    add(3);
    add(23);
     add(94);
    add(232);
     add(4);
    add(231);
    disp();
    printf("\n %d deleted",del());
    add(299);
    disp();
    printf("\n %d deleted",del());
    printf("\n %d deleted",del());
```

```
    printf("\n %d deleted",del());
    printf("\n %d deleted",del());
    printf("\n %d deleted",del());
    disp();
    add(3);
    add(23);
     add(4);
    add(231);
    disp();
}
```

OUTPUT

```
3 added
23 added
94 added
232 added
4 added
Queue Full
Front ---> 3 23 94 232 4 ---> Rear
 3 deleted
299 added
Front ---> 23 94 232 4 299 ---> Rear
 23 deleted
 94 deleted
 232 deleted
 4 deleted
 299 deleted
Queue Empty
3 added
23 added
4 added
231 added
Front ---> 3 23 4 231 ---> Rear
PS C:\Users\bhaum\OneDrive\Desktop\DSA_CODES\queue>
```

CLICK HERE FOR GITHUB LINK

Q6) . Implementation of singly linked list

main.c

CODE

```
#include <ll.h>

main()

{

    node *h;

    l_in(&h);

    ins(&h,77,0);

    menu(&h);

    del(&h,2);

}
```


```
#include <ll.h>
main()
{
    node *h;
    l_in(&h);
    ins(&h,77,0);
    menu(&h);
    del(&h,2);
}
```


OUTPUT

```
77 inserted

1)Insert
2)Display
3)Delete
4)Lenght
5)Exit
Enter your choice: 1


Enter the position you want to insert(-1 for last element and 0 for head): 0

Enter the element to be inserted: jlj

jlj inserted

1)Insert
2)Display
3)Delete
4)Lenght
5)Exit
Enter your choice: 1


Enter the position you want to insert(-1 for last element and 0 for head): 89

Enter the element to be inserted: 89.89
```

```
89.89 inserted

1)Insert
2)Display
3)Delete
4)Lenght
5)Exit
Enter your choice: 2

1)Head
2)Tail
3)Whole
4)Custom
Enter Display choice: 3

The whole linked list is

         _
        ___
       _____
         |
         v
```

```
#######################
#                 jlj #
#######################
         |
         v
#######################
#                  77 #
#######################
         |
         v
#######################
#               89.89 #
#######################


1)Insert
2)Display
3)Delete
4)Lenght
5)Exit
Enter your choice: 5

77 deleted
```

CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE

Q7) Implementation of stack using linked list

CODE

```
#include <stdio.h>
struct node
{
    int d;
    struct node *n;
} * h;
typedef struct node node;

push(int d)
{
    node *t = malloc(sizeof(node));
    t->d = d;
    t->n = h;
    h=t;
    printf("\n%d pushed",d);
}

pop()
{
    if (!h)
    {
        printf("\nStack Empty");
        return;
    }
    int a = h->d;
```

```
    h = h->n;
    return a;
}

disp()
{
    if (!h)
    {
        printf("\nStack Empty");
        return;
    }
    printf("\nStack is: ");
    node *p = h;
    while (p)
    {
        printf("%d ", p->d);
        p = p->n;
    }
}
main()
{
    push(3);
    push(23);
     push(94);
    push(232);
     push(4);
    push(231);
    disp();
    printf("\n %d poped",pop());
    push(299);
```

```
    disp();
    printf("\n %d poped",pop());
    printf("\n %d poped",pop());
    printf("\n %d poped",pop());
    printf("\n %d poped",pop());
    printf("\n %d poped",pop());
    disp();
    push(3);
    push(23);
     push(4);
    push(231);
    disp();
}
```

OUTPUT

```
3 pushed
23 pushed
94 pushed
232 pushed
4 pushed
231 pushed
Stack is: 231 4 232 94 23 3
 231 poped
299 pushed
Stack is: 299 4 232 94 23 3
 299 poped
 4 poped
 232 poped
 94 poped
 23 poped
Stack is: 3
3 pushed
23 pushed
4 pushed
231 pushed
Stack is: 231 4 23 3 3
PS C:\Users\bhaum\OneDrive\Desktop\DSA_CODES\stack>
```

CLICK HERE FOR GITHUB LINK

Q8) Implementation of queue using linked list

CODE

```
#include <stdio.h>
struct node
{
    int d;
    struct node *n;
} * h;
typedef struct node node;

enqueue(int d)
{
    node *t = malloc(sizeof(node));
    t->d = d;
    t->n = 0;
    printf("\n%d enqueueed",d);
    if (!h)
    {
        h = t;
        return;
    }
    node *p = h;
    while (p->n)
        p = p->n;
    p->n = t;
}
```

```
dequeue()
{
    if (!h)
    {
        printf("\nQueue Empty");
        return;
    }
    int a = h->d;
    h = h->n;
    return a;
}

disp()
{
    if (!h)
    {
        printf("\nQueue Empty");
        return;
    }
    printf("\nQueue is: ");
    node *p = h;
    while (p)
    {
        printf("%d ", p->d);
        p = p->n;
    }
}
main()
{
    enqueue(3);
```

```
    enqueue(23);
     enqueue(94);
    enqueue(232);
     enqueue(4);
    enqueue(231);
    disp();
    printf("\n %d dequeued",dequeue());
    enqueue(299);
    disp();
    printf("\n %d dequeued",dequeue());
    printf("\n %d dequeued",dequeue());
    printf("\n %d dequeued",dequeue());
    printf("\n %d dequeued",dequeue());
    printf("\n %d dequeued",dequeue());
    disp();
    enqueue(3);
    enqueue(23);
     enqueue(4);
    enqueue(231);
    disp();
}
```

OUTPUT

```
3 enqueueed
23 enqueueed
94 enqueueed
232 enqueueed
4 enqueueed
231 enqueueed
Queue is: 3 23 94 232 4 231
 3 dequeued
299 enqueueed
Queue is: 23 94 232 4 231 299
 23 dequeued
 94 dequeued
 232 dequeued
 4 dequeued
 231 dequeued
Queue is: 299
3 enqueueed
23 enqueueed
4 enqueueed
231 enqueueed
Queue is: 299 3 23 4 231
```

CLICK HERE FOR GITHUB LINK