



BHARATI VIDYAPEETH'S COLLEGE OF ENGINEERING

A-4, PASCHIM VIHAR, NEW DELHI - 110063

CERTIFICATE

This is to Certify that Mr./Miss..... *Vishun Verma*

Roll No. *02211504919* Branch: *EEE*

has satisfactorily completed his / her terms work in

Data Structures LAB subject

Head of the Department

Date

Professor
Incharge

INDEX

NAME : Varun Verma

Roll.No.

0	2	2	1	1	5	0	4	9	1	9
---	---	---	---	---	---	---	---	---	---	---

EXPT. NO.	NAME OF THE EXPERIMENT	DATE OF PERFORMANCE	DATE OF SUBMISSION	REMARKS
1	Write a menu driven program to implement linear and binary search			
2	Write a menu driven program to insert and delete element from array			
3	Write a program to insert, delete and display a singly linked list			
4	Write a program to implement A doubly linked list			
5	Write a program to implement a circular linked list			
6	Write a program to implement stacks as (i) Arrays (ii) Linked List			
7	Write a program to implement queue as (i) Arrays (ii) linked list			
8	Program to convert Infix expression to postfix expressions			
9				
10				
11				
12				

Any Additional individual work done by students.

1	Write a program to reverse elements in stack using Recursion			
2	Write a program to count the no. of nodes in a Binary search tree			
3				

Varun Verma

02211504919

EEE

EXPERIMENT – 10

Content Beyond Syllabus

AIM: Write a C program to Count the number of nodes in a binary search tree

SOURCE CODE:

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
    struct node *lchild;
    int info;
    struct node *rchild;
};

struct node *insert(struct node *ptr, int ikey);
void display(struct node *ptr,int level);
int NodesAtLevel(struct node *ptr, int level) ;

int main()
{
    struct node *root=NULL,*root1=NULL,*ptr;
    int choice,k,item,level;

    while(1)
    {
        printf("\nSelect operation to be done on tree: \n");
        printf(" 1.Insert Element in tree \n");
        printf(" 2.Display Tree \n");
        printf(" 3.Count Number of Nodes \n");
        printf(" 4.Exit\n");
        printf("\n Enter Your Choice : ");
        scanf("%d",&choice);

        switch(choice)
        {

            case 1:
```

```

        printf("\n Enter the value to be inserted : ");
        scanf("%d",&k);
        root = insert(root, k);
        break;

case 2:
    printf("\n");
    display(root,0);
    printf("\n");
    break;

case 3:
    printf("\n");
    printf(" Enter any level : ");
    scanf("%d",&level);
    printf("\n Number of nodes at [ %d ] Level : %d\n",level,NodesAtLevel(root,level));
    break;

case 4:
    exit(1);

default:
    printf("\n Wrong Choice\n");

}/*End of switch */
}/*End of while */

return 0;

}/*End of main( )*/



struct node *insert(struct node *ptr, int ikey )
{
    if(ptr==NULL)
    {
        ptr = (struct node *) malloc(sizeof(struct node));
        ptr->info = ikey;
        ptr->lchild = NULL;
        ptr->rchild = NULL;
    }
    else if(ikey < ptr->info) /*Insertion in left subtree*/
        ptr->lchild = insert(ptr->lchild, ikey);
    else if(ikey > ptr->info) /*Insertion in right subtree */
        ptr->rchild = insert(ptr->rchild, ikey);
    else
        printf("\n Duplicate Key\n");
    return(ptr);
}/*End of insert( )*/

```

```
void display(struct node *ptr,int level)
{
    int i;
    if(ptr == NULL )/*Base Case*/
        return;
    else
    {
        display(ptr->rchild, level+1);
        printf("\n");
        for (i=0; i<level; i++)
            printf("    ");
        printf("%d", ptr->info);
        display(ptr->lchild, level+1);
    }
}/*End of display()*/



int NodesAtLevel(struct node *ptr, int level)
{
    if(ptr==NULL)
        return 0;
    if(level==0)
        return 1;
    return NodesAtLevel(ptr->lchild,level-1) + NodesAtLevel(ptr->rchild,level-1);
}/*End of NodesAtLevel()*/
```

OUTPUT:

i) Menu and binary search tree creation

```
Select operation to be done on tree:  
1.Insert Element in tree  
2.Display Tree  
3.Count Number of Nodes  
4.Exit  
  
Enter Your Choice : 1  
Enter the value to be inserted : 5
```

```
Enter Your Choice : 1  
Enter the key to be inserted : 3  
  
Enter Your Choice : 1  
Enter the key to be inserted : 6  
  
Enter Your Choice : 1  
Enter the key to be inserted : 9
```

ii) Displaying Tree

```
Enter Your Choice : 2  
  
      9  
    6     5  
  3
```

iii) Counting Nodes

```
Enter Your Choice : 3  
Enter any level : 1  
Number of nodes at [ 1 ] Level : 2
```

```
Enter Your Choice : 3  
Enter any level : 2  
Number of nodes at [ 2 ] Level : 1
```

VARUN VERMA
02211504919
EEE

CONTENT BEYOND SYLLABUS

Aim: Write a C program to reverse elements in the stack using recursion.

```
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
struct node{
    int data;
    struct node *next;
};

void push(struct node **head,int value) {
    struct node *temp=(struct node*)malloc(sizeof(struct node));
    if(temp==NULL) {
        printf("\t\tInsufficient memory\n");
        return ;
    }
    temp->data=value;
    temp->next=*head;
    *head = temp;
    return;
}

int pop(struct node **head) {
    int element;
    struct node *temp=*head;
    if (*head == NULL) {
        printf("\t\tStack empty\n");
        return;
    }
    else {
        element=temp->data;
        *head=temp->next;
        free(temp);
        return element;
    }
}
int isEmpty(struct node *head) {
```

```

if(head==NULL)
    return 1;
else
    return 0;
}

void reverse(struct node **head) {
    if(!isEmpty(*head)) {
        int temp=pop(head);
        reverse(head);
        insertAtBottom(head,temp);
    }
}

void insertAtBottom(struct node **head,int value)
{
    if(isEmpty(*head)) {
        //printf("Empty stack: %d\n",value);
        push(head,value);
    }
    else {
        int temp=pop(head);
        insertAtBottom(head,value);
        push(head,temp);
    }
}

void display(struct node *head)
{
    struct node *temp=head;
    while(temp!=NULL) {
        printf("\t%d ", temp->data);
        temp=temp->next;
    }
    printf("\n");
}

// -----PARTITION_FOR_LESS_CONFUSION-----

int main() {
    struct node *stack=NULL;
    int value,test,answer,choice;
    printf("\n _____REVERSE_STACK_ELEMENTS_____"
          "\n");

    printf(" Press 1 to Push \n Enter 2 to Reverse\n Enter 3 to Display\n Enter 4 to Exit\n");
    while(1) {
        printf("\n      Enter your choice: ");
        scanf("%d",&choice);
        switch(choice) {
            case 1:
                printf("\tEnter number to insert: ");

```

```
        scanf("%d",&value);
        push(&stack, value);
        break;
    case 2:
        reverse(&stack);
        printf("\tReversed.\n");
        break;
    case 3:
        display(stack);
        break;
    case 4:
        exit(0);
        break;
    default:
        printf("\tWrong choice\n");
        break;
    }
}
}
```

OUTPUT

REVERSE_STACK_ELEMENTS

Press 1 to Push

Enter 2 to Reverse

Enter 3 to Display

Enter 4 to Exit

Enter your choice: 1

Enter number to insert: 1

Enter your choice: 1

Enter number to insert: 34

Enter your choice: 1

Enter number to insert: 4373563

Enter your choice: 33

Wrong choice

Enter your choice: 3

4373563 34 1

Enter your choice: 2

Reversed.

Enter your choice: 3

1 34 4373563

Enter your choice: 4

...Program finished with exit code 0

Press ENTER to exit console.█

Experiment 8

Aim: -Write a program to convert Infix to postfix

Name: - Varun Verma

Branch: -EEE

Roll number: - 02211504919

Source code: -

```
#include<stdio.h>
#include<ctype.h>
```

```
char stack[100];
int top = -1;
```

```
void push(char x)
{
    stack[++top] = x;
}
```

```
char pop()
{
    if(top == -1)
        return -1;
    else
        return stack[top--];
}
```

```
int priority(char x)
{
    if(x == '(')
        return 0;
    if(x == '+' || x == '-')
        return 1;
    if(x == '*' || x == '/')
        return 2;
    return 0;
}
```

```
int main()
{
    char exp[100];
    char *e, x;
```

```

printf("Enter the expression : ");
scanf("%s",exp);
printf("\n");
e = exp;

while(*e != '\0')
{
    if(isalnum(*e))
        printf("%c ",*e);
    else if(*e == '(')
        push(*e);
    else if(*e == ')')
    {
        while((x = pop()) != '(')
            printf("%c ", x);
    }
    else
    {
        while(priority(stack[top]) >= priority(*e))
            printf("%c ",pop());
        push(*e);
    }
    e++;
}

while(top != -1)
{
    printf("%c ",pop());
}
return 0;
}

```

Screen shot: -

The screenshot shows a Linux desktop environment with a dark theme. A Firefox browser window is open, displaying the URL <https://www.programiz.com/c-programming/online-compiler/>. The page title is "C Online Compiler". The main content area shows a code editor with a file named "main.c" containing C code for a reverse polish calculator. The code uses a stack to handle operators and parentheses. The "Run" button is highlighted in blue. To the right of the code editor is an "Output" pane showing the result of the compilation and execution of the code. The output shows the command "/tmp/VTGReQm7si.o" and the expression "(6-7)*(8-9)" followed by the result "6 7 - 8 9 - *". Below the code editor, there is a vertical toolbar with icons for file operations like Open, Save, and Print, as well as other application icons.

```
if(isalnum(*e))
    printf("%c ",*e);
else if(*e == '(')
    push(*e);
else if(*e == ')')
{
    while((x = pop()) != '(')
        printf("%c ", x);
}
else
{
    while(priority(stack[top]) >= priority(*e))
        printf("%c ",pop());
    push(*e);
}
e++;
}
while(top != -1)
{
    printf("%c ",pop());
}return 0;
}
```

Activities Firefox Web Browser ▾ Wed 07:55

Bharati Program Google Untitled vbbots Inbox (2) Learning Examples script.g C Prog Online F Online X +

https://www.programiz.com/c-programming/online-compiler/

Programiz C Online Compiler

POWER YOUR IDEAS AutoCAD for less than ₹270*/day When you subscribe for 1-year

AUTODESK AUTOCAD BUY NOW > *T&Cs apply

Learn Python App

main.c

```
42 if(isalnum(*e))
43     printf("%c ", *e);
44 else if(*e == '(')
45     push(*e);
46 else if(*e == ')')
47 {
48     while((x = pop()) != '(')
49         printf("%c ", x);
50 }
51 else
52 {
53     while(priority(stack[top]) >= priority(*e))
54         printf("%c ",pop());
55     push(*e);
56 }
57 e++;
58 }
59
60 while(top != -1)
61 {
62     printf("%c ",pop());
63 }return 0;
64 }
```

/tmp/VTGReQm7si.o
Enter the expression : (a-b)*(d+g)
a b - d g + *

Clear

https://adclick.g.doubleclick.net/pcs/click?xai=AKAOjssnxES3Hv_Ozi93R8qOQnK9uxa43O9r.../autocad/subscribe?mktvar002=3360537|DSP|23011047|283267396|138316165&dclid=%edclid!

EXPERIMENT:-7

NAME:- VARUN VERMA
ROLL NUMBER:- 02211504919

Aim :- Write a program to implement queue

Using array:

```
#include
<stdio.h>

#define MAX 50

void insert();
void delete();
void display();
int queue_array[MAX];
int rear = - 1;
int front = - 1;
main()
{
    int choice;
    while (1)
    {
        printf("1.Insert element to queue \n");
        printf("2.Delete element from queue \n");
        printf("3.Display all elements of queue \n");
        printf("4.Quit \n");
        printf("Enter your choice : ");
        scanf("%d", &choice);
        switch (choice)
        {
            case 1:
                insert();
                break;
            case 2:
                delete();
                break;
```

```

        case 3:
            display();
            break;
        case 4:
            exit(1);
        default:
            printf("Wrong choice \n");
    }
}

void insert()
{
    int add_item;
    if (rear == MAX - 1)
        printf("Queue Overflow \n");
    else
    {
        if (front == - 1)

            front = 0;
        printf("Inset the element in queue : ");
        scanf("%d", &add_item);
        rear = rear + 1;
        queue_array[rear] = add_item;
    }
}

void delete()
{
    if (front == - 1 || front > rear)
    {
        printf("Queue Underflow \n");
        return ;
    }
    else
    {
        printf("Element deleted from queue is : %d\n", queue_array[front]);
        front = front + 1;
    }
}

void display()

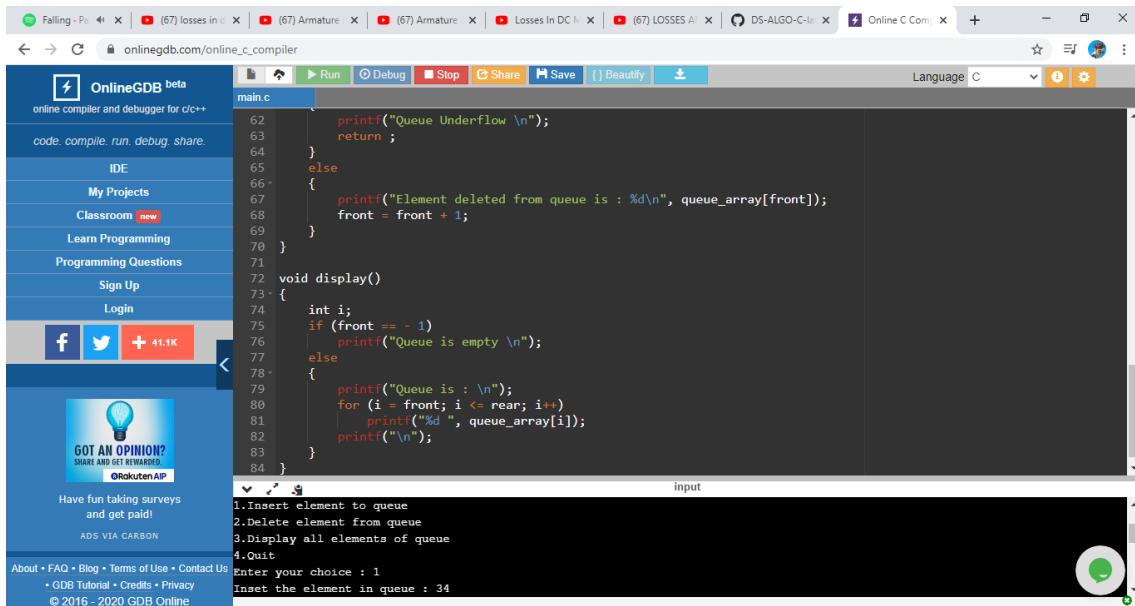
```

```

{
    int i;
    if (front == - 1)
        printf("Queue is empty \n");
    else
    {
        printf("Queue is : \n");
        for (i = front; i <= rear; i++)
            printf("%d ", queue_array[i]);
        printf("\n");
    }
}

```

SCREEN SHOT:-



The screenshot shows the OnlineGDB beta IDE interface. The main window displays the C code for a queue implementation. The code includes functions for inserting elements into the queue, deleting elements from the queue, and displaying the queue's contents. The IDE has a toolbar with options like Run, Debug, Stop, Share, Save, and Beautify. The language is set to C. The bottom panel shows the input and output fields, where the user has entered '1' and 'Inset the element in queue : 34'. There are also social sharing icons and a feedback survey banner.

```

main.c
62     printf("Queue Underflow \n");
63     return ;
64 }
65 else
66 {
67     printf("Element deleted from queue is : %d\n", queue_array[front]);
68     front = front + 1;
69 }
70 }
71 void display()
72 {
73     int i;
74     if (front == - 1)
75         printf("Queue is empty \n");
76     else
77     {
78         printf("Queue is : \n");
79         for (i = front; i <= rear; i++)
80             printf("%d ", queue_array[i]);
81         printf("\n");
82     }
83 }
84 }

1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice : 1
Inset the element in queue : 34

```

The screenshot shows the OnlineGDB beta IDE interface. The main window displays a C program named 'main.c' with the following code:

```
63     return ;
64 }
65 else
66 {
67     printf("Element deleted from queue is : %d\n", queue_array[front]);
68     front = front + 1;
69 }
70 }

72 void display()
73 {
74     int i;
75     if (front == -1)
76         printf("Queue is empty \n");
77     else
78     {
79         printf("Queue is : \n");
80         for (i = front; i <= rear; i++)
81             printf("%d ", queue_array[i]);
82         printf("\n");
83     }
84 }
85
```

Below the code editor, there is a terminal window with the following input and output:

```
input
34
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice :
```

A green speech bubble icon is visible in the bottom right corner of the terminal window.

The screenshot shows the OnlineGDB beta IDE interface. The main window displays the same C program 'main.c' as the previous screenshot. The terminal window now shows the user has selected option 2 (Delete element from queue) and the output is:

```
input
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice : 2
Element deleted from queue is : 34
```

A green speech bubble icon is visible in the bottom right corner of the terminal window.

Using linked list:-

```
#include
<stdio.h>
>
#include <stdlib.h>

struct node
{
    int info;
    struct node *ptr;
}*front,*rear,*temp,*front1;

int frontelement();
void enq(int data);
void deq();
void empty();
void display();
void create();
void queuesize();

int count = 0;

void main()
{
    int no, ch, e;

    printf("\n 1 - Enque");
    printf("\n 2 - Deque");
    printf("\n 3 - Front element");
    printf("\n 4 - Empty");
    printf("\n 5 - Exit");
    printf("\n 6 - Display");
    printf("\n 7 - Queue size");
    create();
    while (1)
    {
        printf("\n Enter choice : ");
        scanf("%d", &ch);
        switch (ch)
```

```

{
    case 1:
        printf("Enter data : ");
        scanf("%d", &no);
        enq(no);
        break;
    case 2:
        deq();
        break;
    case 3:
        e = frontelement();
        if (e != 0)
            printf("Front element : %d", e);
        else
            printf("\n No front element in Queue as queue is empty");
        break;
    case 4:
        empty();
        break;
    case 5:
        exit(0);
    case 6:
        display();
        break;
    case 7:
        queuesize();
        break;
    default:
        printf("Wrong choice, Please enter correct choice  ");
        break;
    }
}
}

/* Create an empty queue */
void create()
{
    front = rear = NULL;
}

/* Returns queue size */
void queuesize()
{

```

```

        printf("\n Queue size : %d", count);
    }

/* Enqueing the queue */
void enq(int data)
{
    if (rear == NULL)
    {
        rear = (struct node *)malloc(1*sizeof(struct node));
        rear->ptr = NULL;
        rear->info = data;
        front = rear;
    }
    else
    {
        temp=(struct node *)malloc(1*sizeof(struct node));
        rear->ptr = temp;
        temp->info = data;
        temp->ptr = NULL;

        rear = temp;
    }
    count++;
}

/* Displaying the queue elements */
void display()
{
    front1 = front;

    if ((front1 == NULL) && (rear == NULL))
    {
        printf("Queue is empty");
        return;
    }
    while (front1 != rear)
    {
        printf("%d ", front1->info);
        front1 = front1->ptr;
    }
    if (front1 == rear)
        printf("%d", front1->info);
}

```

```

/* Dequeing the queue */
void deq()
{
    front1 = front;

    if (front1 == NULL)
    {
        printf("\n Error: Trying to display elements from empty queue");
        return;
    }
    else
        if (front1->ptr != NULL)
    {
        front1 = front1->ptr;
        printf("\n Dequeued value : %d", front->info);
        free(front);
        front = front1;
    }
    else
    {
        printf("\n Dequeued value : %d", front->info);
        free(front);
        front = NULL;
        rear = NULL;
    }
    count--;
}

/* Returns the front element of queue */
int frontelement()
{
    if ((front != NULL) && (rear != NULL))
        return(front->info);
    else
        return 0;
}

/* Display if queue is empty or not */
void empty()
{
    if ((front == NULL) && (rear == NULL))
        printf("\n Queue empty");
}

```

```
else
    printf("Queue not empty");
}
```

Screen shot:

The screenshot shows the OnlineGDB beta IDE interface. The main window displays a C program named `main.c`. The code implements a queue using pointers `front` and `rear`, and a counter `count`. It includes functions to add elements to the queue (`enqueue`), remove elements from the front (`dequeue`), and check if the queue is empty (`empty`). The terminal window below shows the execution of the program, with input commands like "Enter choice : 1" and "Enter data : 78" followed by the output "99 45 78". The sidebar on the left contains links for "My Projects", "Classroom", "Learn Programming", "Programming Questions", "Sign Up", and "Login". A social media sharing bar is visible at the bottom of the sidebar.

```
main.c
147     rear = NULL;
148 }
149 count--;
150 }
151 /* Returns the front element of queue */
152 int frontelement()
153 {
154     if ((front != NULL) && (rear != NULL))
155         return(front->info);
156     else
157         return 0;
158 }
159
160 /* Display if queue is empty or not */
161 void empty()
162 {
163     if ((front == NULL) && (rear == NULL))
164         printf("\n Queue empty");
165     else
166         printf("Queue not empty");
167 }
168
169
```

input

```
Enter choice : 1
Enter data : 78
Enter choice : 6
99 45 78
Enter choice :
```

The screenshot shows the OnlineGDB beta IDE interface. The code editor displays a C program named main.c. The code implements a queue using pointers to a struct. It includes functions for enqueueing, dequeuing, displaying the front element, and checking if the queue is empty. The IDE has a toolbar with Run, Debug, Stop, Share, Save, and Beautify buttons. The language is set to C. The status bar at the bottom shows the input prompt 'input'.

```
main.c
147     rear = NULL;
148 }
149 count--;
150 }
151 /* Returns the front element of queue */
152 int frontelement()
153 {
154     if ((front != NULL) && (rear != NULL))
155         return(front->info);
156     else
157         return 0;
158 }
159 }
160 /* Display if queue is empty or not */
161 void empty()
162 {
163     if ((front == NULL) && (rear == NULL))
164         printf("\n Queue empty");
165     else
166         printf("Queue not empty");
167 }
168 }
169 }
```

input

The screenshot shows the execution results of the C program. The user enters data values (99, 45, 78) and choice values (6, 1, 2). The program processes these inputs and outputs the dequeued value (99) and the choice value (2).

```
Enter data : 99
Enter choice : 6
99
Enter choice : 1
Enter data : 45
Enter choice : 1
Enter choice : 2
Dequeued value : 99
Enter choice : 2
```

S Never Ch | (67) losses in c | (67) Armature | (67) Armature | Losses In DC N | (67) LOSSES A | DS-ALGO-C-In | Online C Comp | + - ⌂ X

onlinegdb.com/online_c_compiler

OnlineGDB beta
online compiler and debugger for c/c++

code, compile, run, debug, share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

f 41.1K SHARE AND GET REWARDED Rakuten AIP

GOT AN OPINION?

Have fun taking surveys and get paid!

ADS VIA CARBON

About • FAQ • Blog • Terms of Use • Contact Us
• GDB Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online

main.c

```
147     rear = NULL;
148 }
149 count--;
150 }
151 /* Returns the front element of queue */
152 int frontelement()
153 {
154     if ((front != NULL) && (rear != NULL))
155         return(front->info);
156     else
157         return 0;
158 }
159
160 /* Display if queue is empty or not */
161 void empty()
162 {
163     if ((front == NULL) && (rear == NULL))
164         printf("\n Queue empty");
165     else
166         printf("Queue not empty");
167 }
168
169
```

input

```
99 45 78
Enter choice : 2

Dequeued value : 99
Enter choice : 3
front element : 45
```

chat icon

NAME :- Varun Verma

Branch: - EEE

Roll no. : - 02211504919

AIM:- Write a program to implement Circular linked list

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
    int data;
    struct node *next;
};

struct node *head;

void begininsert ();
void lastinsert ();
void randominsert();
void begin_delete();
void last_delete();
void random_delete();
void display();
void search();
void main ()
{
```

```
int choice =0;
while(choice != 7)
{
    printf("\n*****Main Menu*****\n");
    printf("\nChoose one option from the following list ... \n");
    printf("\n===== \n");
    printf("\n1.Insert in begining\n2.Insert at last\n3.Delete from Beginning\n4.Delete from
last\n5.Search for an element\n6.Show\n7.Exit\n");
    printf("\nEnter your choice?\n");
    scanf("\n%d",&choice);
    switch(choice)
    {
        case 1:
            begininsert();
            break;
        case 2:
            lastinsert();
            break;
        case 3:
            begin_delete();
            break;
        case 4:
            last_delete();
            break;
        case 5:
            search();
            break;
        case 6:
            display();
            break;
        case 7:
    }
}
```

```
    exit(0);
    break;
default:
printf("Please enter valid choice..");
}
}

void begininsert()
{
    struct node *ptr,*temp;
    int item;
    ptr = (struct node *)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter the node data?");
        scanf("%d",&item);
        ptr -> data = item;
        if(head == NULL)
        {
            head = ptr;
            ptr -> next = head;
        }
        else
        {
            temp = head;
            while(temp->next != head)
                temp = temp->next;
        }
    }
}
```

```

ptr->next = head;
temp -> next = ptr;
head = ptr;
}

printf("\nnnode inserted\n");

}

void lastinsert()
{
    struct node *ptr,*temp;
    int item;
    ptr = (struct node *)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW\n");
    }
    else
    {
        printf("\nEnter Data?");
        scanf("%d",&item);
        ptr->data = item;
        if(head == NULL)
        {
            head = ptr;
            ptr -> next = head;
        }
        else
        {
            temp = head;
            while(temp -> next != head)

```

```
{  
    temp = temp -> next;  
}  
  
temp -> next = ptr;  
ptr -> next = head;  
}  
  
printf("\nnnode inserted\n");  
}
```

```
}  
  
void begin_delete()  
{  
    struct node *ptr;  
    if(head == NULL)  
    {  
        printf("\nUNDERFLOW");  
    }  
    else if(head->next == head)  
    {  
        head = NULL;  
        free(head);  
        printf("\nnnode deleted\n");  
    }  
  
    else  
    {  
        ptr = head;  
        while(ptr -> next != head)  
        {  
            ptr = ptr -> next;  
            ptr->next = head->next;  
        }  
    }  
}
```

```

        free(head);

        head = ptr->next;

        printf("\nnnode deleted\n");

    }

}

void last_delete()

{

    struct node *ptr, *preptr;

    if(head==NULL)

    {

        printf("\nUNDERFLOW");

    }

    else if (head ->next == head)

    {

        head = NULL;

        free(head);

        printf("\nnnode deleted\n");

    }

    else

    {

        ptr = head;

        while(ptr ->next != head)

        {

            preptr=ptr;

            ptr = ptr->next;

        }

        preptr->next = ptr -> next;

        free(ptr);

        printf("\nnnode deleted\n");

    }

}

```

```
}

}

void search()
{
    struct node *ptr;
    int item,i=0,flag=1;
    ptr = head;
    if(ptr == NULL)
    {
        printf("\nEmpty List\n");
    }
    else
    {
        printf("\nEnter item which you want to search?\n");
        scanf("%d",&item);
        if(head ->data == item)
        {
            printf("item found at location %d",i+1);
            flag=0;
        }
        else
        {
            while (ptr->next != head)
            {
                if(ptr->data == item)
                {
                    printf("item found at location %d ",i+1);
                    flag=0;
                    break;
                }
            }
        }
    }
}
```

```
        }

    else

    {

        flag=1;

    }

    i++;

    ptr = ptr -> next;

}

}

if(flag != 0)

{

    printf("Item not found\n");

}

}

}

void display()

{

    struct node *ptr;

    ptr=head;

    if(head == NULL)

    {

        printf("\nnothing to print");

    }

    else

    {

        printf("\n printing values ... \n");

        while(ptr -> next != head)

        {
```

```

    printf("%d\n", ptr -> data);

    ptr = ptr -> next;

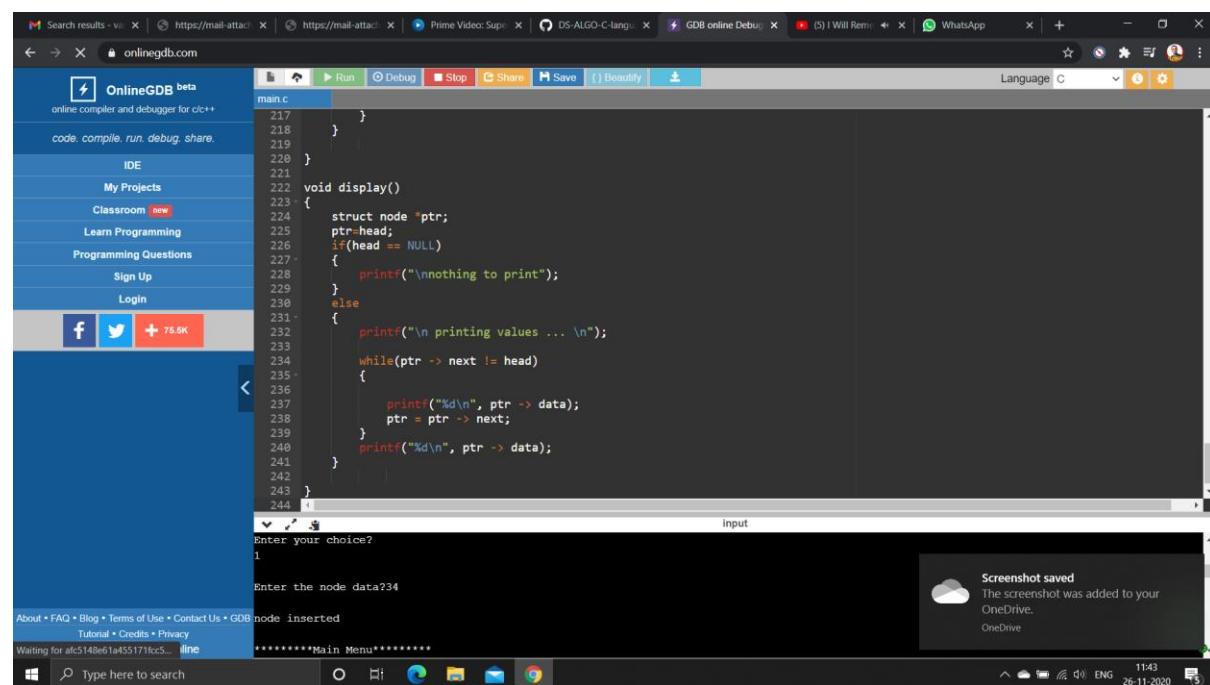
}

printf("%d\n", ptr -> data);

}

}

```



The screenshot shows a Windows desktop environment. A browser window is open on the OnlineGDB beta website, displaying a C program for printing linked list values. The code is as follows:

```
main.c
217     }
218 }
219
220 }
221 void display()
222 {
223     struct node *ptr;
224     ptr = head;
225     if(head == NULL)
226     {
227         printf("\nnothing to print");
228     }
229     else
230     {
231         printf("\n printing values ... \n");
232     }
233     while(ptr -> next != head)
234     {
235         printf("%d\n", ptr -> data);
236         ptr = ptr -> next;
237     }
238     printf("%d\n", ptr -> data);
239 }
240 }
```

The browser interface includes tabs for various websites like WhatsApp, GDB online Debug, and Prime Video. The OnlineGDB sidebar shows links for IDE, My Projects, Classroom, Learn Programming, Programming Questions, Sign Up, and Login. Social sharing icons for Facebook, Twitter, and LinkedIn are present, along with a 76.5K follower count.

Below the code editor, there's an input field with placeholder text "Enter your choice?". The user has typed "1". Below that, another input field says "Enter the node data?". The user has typed "34". At the bottom of the code editor, there's a footer with links to About, FAQ, Blog, Terms of Use, Contact Us, GDB Tutorial, Credits, and Privacy. It also mentions "Waiting for afc5148e61a455171cc5... line".

The desktop taskbar at the bottom shows icons for File Explorer, Task View, Edge, Mail, and Google Chrome. A OneDrive notification bubble is visible in the top right corner, stating "Screenshot saved. The screenshot was added to your OneDrive. OneDrive". The system tray shows the date and time as 26-11-2020 11:43.

Name: - Varun Verma

EEE

02211504919

Aim:- Write a program to print stack as array
and linked list

Source
code:-

```
#include <stdio.h>
#define MAX 3 // Altering this value changes size of stack created
int st[MAX], top=-1;
void push(int st[], int val);
int pop(int st[]);
int peek(int st[]);
void display(int st[]);
int main(int argc, char *argv[]) {
    int val, option;
    do
    {
        printf("\n *****MAIN MENU*****");
        printf("\n 1. PUSH");
        printf("\n 2. POP");
        printf("\n 3. PEEK");
        printf("\n 4. DISPLAY");
        printf("\n 5. EXIT");
        printf("\n Enter your option: ");
        scanf("%d", &option);
        switch(option)
        {
            case 1:
                printf("\n Enter the number to be pushed on stack: ");
                scanf("%d", &val);
                push(st, val);
                break;
            case 2:
```

```

val = pop(st);
if(val != -1)
printf("\n The value deleted from stack is: %d", val);
break;
case 3:
val = peek(st);
if(val != -1)
printf("\n The value stored at top of stack is: %d", val);
break;
case 4:
display(st);
break;
}
}while(option != 5);
return 0;
}
void push(int st[], int val)
{
if(top == MAX-1)
{
printf("\n STACK OVERFLOW");
}
else
{
top++;
st[top] = val;
}
}
int pop(int st[])
{
int val;
if(top == -1)
{
printf("\n STACK UNDERFLOW");
return -1;
}
else
{
val = st[top];
top--;
return val;
}
}
void display(int st[])
{
int i;

```

```

if(top == -1)
printf("\n STACK IS EMPTY");
else
{
for(i=top;i>=0;i--)
printf("\n %d",st[i]);
printf("\n"); // Added for formatting purposes
}
}
int peek(int st[])
{
if(top == -1)
{
printf("\n STACK IS EMPTY");
return -1;
}
else
return (st[top]);
}

```

Stack as a linked list

```

#include <stdio.h>
#include <stdlib.h>
void push();
void pop();
void display();
struct node
{
int val;
struct node *next;
};
struct node *head;
void main ()
{
int choice=0;

```

```
printf("\n*****Stack operations using linked list*****\n");
printf("\n-----\n");
while(choice != 4)
{
    printf("\n\nChose one from the below options...\n");
    printf("\n1.Push\n2.Pop\n3.Show\n4.Exit");
    printf("\n Enter your choice \n");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1:
        {
            push();
            break;
        }
        case 2:
        {
            pop();
            break;
        }
        case 3:
        {
            display();
            break;
        }
        case 4:
        {
            printf("Exiting....");
            break;
        }
        default:
        {
            printf("Please Enter valid choice ");
        }
    };
}
}

void push ()
{
    int val;
    struct node *ptr = (struct node*)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("not able to push the element");
    }
    else
```

```
{  
printf("Enter the value");  
scanf("%d",&val);  
if(head==NULL)  
{  
ptr->val = val;  
ptr -> next = NULL;  
head=ptr;  
}  
else  
{  
ptr->val = val;  
ptr->next = head;  
head=ptr;  
}  
printf("Item pushed");  
}  
}  
}  
void pop()  
{  
int item;  
struct node *ptr;  
if (head == NULL)  
{  
printf("Underflow");  
}  
else  
{  
item = head->val;  
ptr = head;  
head = head->next;  
free(ptr);  
printf("Item popped");  
}  
}  
}  
void display()  
{  
int i;  
struct node *ptr;  
ptr=head;  
if(ptr == NULL)  
{  
printf("Stack is empty\n");  
}  
else  
{
```

```

printf("Printing Stack elements \n");
while(ptr!=NULL)
{
printf("%d\n",ptr->val);
ptr = ptr->next;
}
}
}

```

OUTPUT

STACK AS AN ARRAY: -

The screenshot shows the OnlineGDB beta interface with a C program for a stack. The code defines a stack structure, initializes it, and provides functions for pushing, popping, peeking, and displaying its elements. The OneDrive interface in the background shows a screenshot was saved to OneDrive.

```

main.c
66 val = st[top];
67 top--;
68 return val;
69 }
70 }
71 void display(int st[])
72 {
73 int i;
74 if(top == -1)
75 printf("\n STACK IS EMPTY");
76 else
77 {
78 for(i=top;i>=0;i--)
79 printf("\n %d",st[i]);
80 printf("\n"); // Added for formatting purposes
81 }
82 }
83 int peek(int st[])
84 {
85 if(top == -1)
86 {
87 printf("\n STACK IS EMPTY");
88 return -1;
89 }
90 else
91 return (st[top]);
92 }
93

```

input

2. POP
3. PEEK
4. DISPLAY
5. EXIT

Enter your option: 1

Enter the number to be pushed on stack: 45

Screenshot saved
The screenshot was added to your OneDrive.
OneDrive

```

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.

IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
Facebook Twitter + 76.9K
Red Hat
Massive demand calls for new approaches. Embrace cloud-native dev with Red Hat.
ADS VIA CARBON
About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online

main.c
66 val = st[top];
67 top--;
68 return val;
69 }
70 }
71 void display(int st[])
72 {
73 int i;
74 if(top == -1)
75 printf("\n STACK IS EMPTY");
76 else
77 {
78 for(i=top;i>=0;i--)
79 printf("\n %d",st[i]);
80 printf("\n"); // Added for formatting purposes
81 }
82 }
83 int peek(int st[])
84 {
85 if(top == -1)
86 {
87 printf("\n STACK IS EMPTY");
88 return -1;
89 }
90 else
91 return (st[top]);
92 }
93 }

2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1
Enter the number to be pushed on stack: 12

```

Input

```

main.c
66 val = st[top];
67 top--;
68 return val;
69 }
70 }
71 void display(int st[])
72 {
73 int i;
74 if(top == -1)
75 printf("\n STACK IS EMPTY");
76 else
77 {
78 for(i=top;i>=0;i--)
79 printf("\n %d",st[i]);
80 printf("\n"); // Added for formatting purposes
81 }
82 }
83 int peek(int st[])
84 {
85 if(top == -1)
86 {
87 printf("\n STACK IS EMPTY");
88 return -1;
89 }
90 else
91 return (st[top]);
92 }
93 }

4. DISPLAY
5. EXIT
Enter your option: 4
45
12
25

```

Input

Screenshot saved
The screenshot was added to your OneDrive.
OneDrive

PEEK: -

```

main.c
68     return val;
69 }
70 }
71 void display(int st[])
72 {
73     int i;
74     if(top == -1)
75         printf("\n STACK IS EMPTY");
76     else
77     {
78         for(i=top;i>=0;i--)
79             printf("\n %d",st[i]);
80         printf("\n"); // Added for formatting purposes
81     }
82 }
83 int peek(int st[])
84 {
85     if(top == -1)
86     {
87         printf("\n STACK IS EMPTY");
88         return -1;
89     }
90     else
91         return (st[top]);
92 }
93
94
95
5. EXIT
Enter your option: 3

The value stored at top of stack is: 25
*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK

```

Screenshot saved
The screenshot was added to your OneDrive.
OneDrive

POP: -

```

main.c
68     return val;
69 }
70 }
71 void display(int st[])
72 {
73     int i;
74     if(top == -1)
75         printf("\n STACK IS EMPTY");
76     else
77     {
78         for(i=top;i>=0;i--)
79             printf("\n %d",st[i]);
80         printf("\n"); // Added for formatting purposes
81     }
82 }
83 int peek(int st[])
84 {
85     if(top == -1)
86     {
87         printf("\n STACK IS EMPTY");
88         return -1;
89     }
90     else
91         return (st[top]);
92 }
93
94
95
4. DISPLAY
5. EXIT
Enter your option: 2

The value deleted from stack is: 25
*****MAIN MENU*****
1. PUSH
2. POP

```

WhatsApp icon in the bottom right corner.

The screenshot shows a web-based IDE interface for OnlineGDB. The main window displays a C program for a stack implementation. The code includes functions for push, pop, display, and peek operations. A sidebar on the left provides navigation links for IDE, My Projects, Classroom, Learn Programming, and Programming Questions. A Red Hat advertisement is visible on the left side of the interface. The bottom status bar shows system information like battery level, network, and date/time.

```
main.c
68     return val;
69 }
70 }
71 void display(int st[])
72 {
73     int i;
74     if(top == -1)
75         printf("\n STACK IS EMPTY");
76     else
77     {
78         for(i=top;i>=0;i--)
79             printf("\n %d",st[i]);
80         printf("\n"); // Added for formatting purposes
81     }
82 }
83 int peek(int st[])
84 {
85     if(top == -1)
86     {
87         printf("\n STACK IS EMPTY");
88     }
89     else
90         return (st[top]);
91 }
92
93
94
95
5. EXIT
Enter your option: 4
12
45
*****MAIN MENU*****
1. PUSH
```

STACK AS A LINKED LIST: -

The image displays two side-by-side screenshots of a Windows desktop environment. Both screenshots show the OnlineGDB IDE interface.

Top Screenshot:

- The title bar shows multiple open tabs: (37) Ed Sheeran - I See Fire, Inbox (1,470) - varunverma71@..., DS-ALGO-C-language/Experiments, GDB online Debugger | Compiler, and WhatsApp.
- The OnlineGDB interface has a toolbar with Run, Debug, Stop, Share, Save, and Beautify buttons.
- The code editor window contains the following C code:

```
main.c
93 item = head->val;
94 ptr = head;
95 head = head->next;
96 free(ptr);
97 printf("Item popped");
98 }
99 }
100 }
101 void display()
102 {
103 int i;
104 struct node *ptr;
105 ptr=&head;
106 if(ptr == NULL)
107 {
108 printf("Stack is empty\n");
109 }
110 else
111 {
112 printf("Printing Stack elements \n");
113 while(ptr!=NULL)
114 {
115 printf("%d\n",ptr->val);
116 ptr = ptr->next;
117 }
118 }
119 }
120 }
```

Bottom Screenshot:

- The title bar shows the same tabs as the top screenshot.
- The OnlineGDB interface has a toolbar with Run, Debug, Stop, Share, Save, and Beautify buttons.
- The code editor window contains the same C code as the top screenshot.
- The terminal window below the code editor shows the following interaction:

```
1.Push
2.Pop
3.Show
4.Exit
Enter your choice
Enter the value12
Item pushed
```

A message box in the bottom right corner says "Screenshot saved" with the subtext "The screenshot was added to your OneDrive. OneDrive".

OnlineGDB beta

online compiler and debugger for c/c++

code. compile. run. debug. share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

Facebook Twitter + 76.9K

Red Hat

Massive demand calls for new approaches. Embrace cloud-native dev with Red Hat.

ADS VIA CARBON

About • FAQ • Blog • Terms of Use • Contact Us • GDB

Tutorial • Credits • Privacy

© 2016 - 2020 GDB Online

main.c

```
93 item = head->val;
94 ptr = head;
95 head = head->next;
96 free(ptr);
97 printf("Item popped");
98 }
99 }
100 }
101 void display()
102 {
103 int i;
104 struct node *ptr;
105 ptr=head;
106 if(ptr == NULL)
107 {
108 printf("Stack is empty\n");
109 }
110 else
111 {
112 printf("Printing Stack elements \n");
113 while(ptr!=NULL)
114 {
115 printf("%d\n",ptr->val);
116 ptr = ptr->next;
117 }
118 }
119 }
120 }
```

2.Pop
3.Show
4.Exit
Enter your choice:

1

Enter the value12

Item pushed

Input

11:45 09-12-2020

Simple Man - YouTube

Inbox (1,469) - varunverma71@... | DS-ALGO-C-language/Experiment | GDB online Debugger | Compiler | WhatsApp

Type here to search

OnlineGDB beta

online compiler and debugger for c/c++

code. compile. run. debug. share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

Facebook Twitter + 76.9K

Red Hat

Massive demand calls for new approaches. Embrace cloud-native dev with Red Hat.

ADS VIA CARBON

About • FAQ • Blog • Terms of Use • Contact Us • GDB

Tutorial • Credits • Privacy

© 2016 - 2020 GDB Online

main.c

```
219 }
220 }
221 void display()
222 {
223 struct node *ptr;
224 ptr=head;
225 if(head == NULL)
226 {
227 printf("\nnothing to print");
228 }
229 else
230 {
231 printf("\n printing values ... \n");
232 while(ptr -> next != head)
233 {
234 printf("%d\n", ptr -> data);
235 ptr = ptr -> next;
236 }
237 printf("%d\n", ptr -> data);
238 }
239 }
240 }
```

1

Enter your choice?

7.Exit

Enter the node data?22

Input

11:14 09-12-2020

Experiment 4(mo...docx

Experiment-3(mo...docx

EXPERIMENT-2docx

Type here to search

The screenshot shows a web-based IDE interface for OnlineGDB. The main area displays a C program named 'main.c' with line numbers. The code implements a stack using a linked list structure. It includes functions for pushing items onto the stack, popping items off the stack, and displaying the stack elements. The IDE has a toolbar with buttons for Run, Debug, Stop, Share, Save, and Beautify. The status bar at the bottom shows the date and time (11:46 09-12-2020) and system icons.

```
item = head->val;
ptr = head;
head = head->next;
free(ptr);
printf("Item popped");
}
}
void display()
{
int i;
struct node *ptr;
ptr=head;
if(ptr == NULL)
{
printf("Stack is empty\n");
}
else
{
printf("Printing Stack elements \n");
while(ptr!=NULL)
{
printf("%d\n",ptr->val);
ptr = ptr->next;
}
}
}
int main()
{
3
Printing Stack elements
12
45
}
```

ADS VIA CARBON

About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online

Experiment 4

Aim: - To implement doubly linked list

Name: - Varun Verma

Branch: - EEE

Source Code: -

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
    struct node *prev;
    int n;
    struct node *next;
}*h,*temp,*temp1,*temp2,*temp4;

void insert1();
void insert2();
void insert3();
void traversebeg();
void traverseend(int);
void sort();
void search();
void update();
void delete();

int count = 0;

void main()
{
    int ch;

    h = NULL;
    temp = temp1 = NULL;

    printf("\n 1 - Insert at beginning");
    printf("\n 2 - Insert at end");
    printf("\n 3 - Insert at position i");
    printf("\n 4 - Delete at i");
    printf("\n 5 - Display from beginning");
    printf("\n 6 - Display from end");
    printf("\n 7 - Search for element");
    printf("\n 8 - Sort the list");
    printf("\n 9 - Update an element");
    printf("\n 10 - Exit");
```

```

while (1)
{
    printf("\n Enter choice : ");
    scanf("%d", &ch);
    switch (ch)
    {
        case 1:
            insert1();
            break;
        case 2:
            insert2();
            break;
        case 3:
            insert3();
            break;
        case 4:
            delete();
            break;
        case 5:
            traversebeg();
            break;
        case 6:
            temp2 = h;
            if (temp2 == NULL)
                printf("\n Error : List empty to display ");
            else
            {
                printf("\n Reverse order of linked list is : ");
                traverseend(temp2->n);
            }
            break;
        case 7:
            search();
            break;
        case 8:
            sort();
            break;
        case 9:
            update();
            break;
        case 10:
            exit(0);
        default:
            printf("\n Wrong choice menu");
    }
}

/*
 * TO create an empty node */
void create()
{
    int data;

    temp =(struct node *)malloc(1*sizeof(struct node));
    temp->prev = NULL;
    temp->next = NULL;
    printf("\n Enter value to node : ");
    scanf("%d", &data);
    temp->n = data;
    count++;
}

```

```

/* TO insert at beginning */
void insert1()
{
    if (h == NULL)
    {
        create();
        h = temp;
        templ = h;
    }
    else
    {
        create();
        temp->next = h;
        h->prev = temp;
        h = temp;
    }
}

/* To insert at end */
void insert2()
{
    if (h == NULL)
    {
        create();
        h = temp;
        templ = h;
    }
    else
    {
        create();
        templ->next = temp;
        temp->prev = templ;
        templ = temp;
    }
}

/* To insert at any position */
void insert3()
{
    int pos, i = 2;

    printf("\n Enter position to be inserted : ");
    scanf("%d", &pos);
    temp2 = h;

    if ((pos < 1) || (pos >= count + 1))
    {
        printf("\n Position out of range to insert");
        return;
    }
    if ((h == NULL) && (pos != 1))
    {
        printf("\n Empty list cannot insert other than 1st position");
        return;
    }
    if ((h == NULL) && (pos == 1))
    {
        create();
        h = temp;
        templ = h;
    }
}

```

```

        return;
    }
else
{
    while (i < pos)
    {
        temp2 = temp2->next;
        i++;
    }
    create();
    temp->prev = temp2;
    temp->next = temp2->next;
    temp2->next->prev = temp;
    temp2->next = temp;
}
}

/* To delete an element */
void delete()
{
    int i = 1, pos;

    printf("\n Enter position to be deleted : ");
    scanf("%d", &pos);
    temp2 = h;

    if ((pos < 1) || (pos >= count + 1))
    {
        printf("\n Error : Position out of range to delete");
        return;
    }
    if (h == NULL)
    {
        printf("\n Error : Empty list no elements to delete");
        return;
    }
    else
    {
        while (i < pos)
        {
            temp2 = temp2->next;
            i++;
        }
        if (i == 1)
        {
            if (temp2->next == NULL)
            {
                printf("Node deleted from list");
                free(temp2);
                temp2 = h = NULL;
                return;
            }
        }
        if (temp2->next == NULL)
        {
            temp2->prev->next = NULL;
            free(temp2);
            printf("Node deleted from list");
            return;
        }
        temp2->next->prev = temp2->prev;
    }
}

```

```

        if (i != 1)
            temp2->prev->next = temp2->next;           /* Might not need this
statement if i == 1 check */
        if (i == 1)
            h = temp2->next;
        printf("\n Node deleted");
        free(temp2);
    }
    count--;
}

/* Traverse from beginning */
void traversebeg()
{
    temp2 = h;

    if (temp2 == NULL)
    {
        printf("List empty to display \n");
        return;
    }
    printf("\n Linked list elements from begining : ");

    while (temp2->next != NULL)
    {
        printf(" %d ", temp2->n);
        temp2 = temp2->next;
    }
    printf(" %d ", temp2->n);
}

/* To traverse from end recursively */
void traverseend(int i)
{
    if (temp2 != NULL)
    {
        i = temp2->n;
        temp2 = temp2->next;
        traverseend(i);
        printf(" %d ", i);
    }
}

/* To search for an element in the list */
void search()
{
    int data, count = 0;
    temp2 = h;

    if (temp2 == NULL)
    {
        printf("\n Error : List empty to search for data");
        return;
    }
    printf("\n Enter value to search : ");
    scanf("%d", &data);
    while (temp2 != NULL)
    {
        if (temp2->n == data)
        {
            printf("\n Data found in %d position", count + 1);
        }
    }
}

```

```

        return;
    }
    else
        temp2 = temp2->next;
        count++;
}
printf("\n Error : %d not found in list", data);
}

/* To update a node value in the list */
void update()
{
    int data, data1;

    printf("\nEnter node data to be updated : ");
    scanf("%d", &data);
    printf("\nEnter new data : ");
    scanf("%d", &data1);
    temp2 = h;
    if (temp2 == NULL)
    {
        printf("\n Error : List empty no node to update");
        return;
    }
    while (temp2 != NULL)
    {
        if (temp2->n == data)
        {

            temp2->n = data1;
            traversebeg();
            return;
        }
        else
            temp2 = temp2->next;
    }

    printf("\n Error : %d not found in list to update", data);
}

/* To sort the linked list */
void sort()
{
    int i, j, x;

    temp2 = h;
    temp4 = h;

    if (temp2 == NULL)
    {
        printf("\n List empty to sort");
        return;
    }

    for (temp2 = h; temp2 != NULL; temp2 = temp2->next)
    {
        for (temp4 = temp2->next; temp4 != NULL; temp4 = temp4->next)
        {
            if (temp2->n > temp4->n)
            {
                x = temp2->n;

```

```
        temp2->n = temp4->n;
        temp4->n = x;
    }
}
traversebeg();
}
```

Screen shot :

The screenshot shows a code editor with a syntax highlighter. The code is a C program with the following structure:

```
345     temp2->n = temp4->n;
346     temp4->n = x;
347 }
348 }
349 }
350 traversebeg();
351 }
```

Below the code, there are tabs for "input" and "Output". To the right, there are buttons for "clear the output" and "syntax highlight". A red message at the bottom left says "Runtime error #stdin #stdout 0s 4244KB". At the bottom right, there are "save" and "ideone it!" buttons.

Runtime error #stdin #stdout 0s 4456KB comments (0)

stdin

copy

stdout

copy

List (First to Last):

Output

```
gcc -o /tmp/5XSumhB0Qo.o /tmp/5XSumhB0Qo.c -lm
/tmp/5XSumhB0Qo.o
1 - Insert at beginning
2 - Insert at end
3 - Insert at position i
4 - Delete at i
5 - Display from beginning
6 - Display from end
7 - Search for element
8 - Sort the list
9 - Update an element
10 - Exit
Enter choice : 1
Enter value to node : 34

Enter choice : 1

Enter value to node : 34

Enter choice : 5

Linked list elements from begining :  34  34
Enter choice : |
```

```
gcc -o /tmp/GSOhb615dZ.o /tmp/GSOhb615dZ.c -lm
/tmp/GSOhb615dZ.o
1 - Insert at beginning
2 - Insert at end
3 - Insert at position i
4 - Delete at i
5 - Display from beginning
6 - Display from end
7 - Search for element
8 - Sort the list
9 - Update an element
10 - Exit
Enter choice : 1
Enter value to node : 34

Enter choice : 1
Enter value to node : 34
Enter choice : 3
Enter position to be inserted : 2

Enter value to node : 45

Enter choice : 4

Enter position to be deleted : 2
Node deleted
Enter choice :
```

Experiment : -3

Aim: - write a program to implement singly linked list

Name: -Varun verma

Roll No.: - 02211504919

Branch: - EEE

Source Code: -

```
1. #include <stdio.h>
2. #include <malloc.h>
3. #include <stdlib.h>
4.
5. struct node {
6.     int value;
7.     struct node *next;
8. };
9.
10.    void insert();
11.    void display();
12.    void delete();
13.    int count();
14.
15.    typedef struct node DATA_NODE;
16.
17.    DATA_NODE *head_node, *first_node, *temp_node = 0,
18.             *prev_node, next_node;
19.    int data;
20.
21.    int main() {
22.        int option = 0;
23.
24.        printf("Singly Linked List Example - All
25.               Operations\n");
26.
```

```
27.     printf("\nOptions\n");
28.     printf("1 : Insert into Linked List \n");
29.     printf("2 : Delete from Linked List \n");
30.     printf("3 : Display Linked List\n");
31.     printf("4 : Count Linked List\n");
32.     printf("Others : Exit()\n");
33.     printf("Enter your option:");
34.     scanf("%d", &option);
35.     switch (option) {
36.     case 1:
37.         insert();
38.         break;
39.     case 2:
40.         delete();
41.         break;
42.     case 3:
43.         display();
44.         break;
45.     case 4:
46.         count();
47.         break;
48.     default:
49.         break;
50.     }
51. }
52.
53. return 0;
54. }
```

55.

```
56. void insert() {
57.     printf("\nEnter Element for Insert Linked List :
      \n");
58.     scanf("%d", &data);
59.
60.     temp_node = (DATA_NODE *) malloc(sizeof
      (DATA_NODE));
61.
62.     temp_node->value = data;
63.
64.     if (first_node == 0) {
65.         first_node = temp_node;
66.     } else {
67.         head_node->next = temp_node;
68.     }
69.     temp_node->next = 0;
70.     head_node = temp_node;
```

```
71.     fflush(stdin);
72. }
73.
74. void delete() {
75.     int countvalue, pos, i = 0;
76.     countvalue = count();
77.     temp_node = first_node;
78.     printf("\nDisplay Linked List : \n");
79.
80.     printf("\nEnter Position for Delete Element : \n");
81.     scanf("%d", &pos);
82.
83.     if (pos > 0 && pos <= countvalue) {
84.         if (pos == 1) {
85.             temp_node = temp_node -> next;
86.             first_node = temp_node;
87.             printf("\nDeleted Successfully \n\n");
88.         } else {
89.             while (temp_node != 0) {
90.                 if (i == (pos - 1)) {
91.                     prev_node->next = temp_node->next;
92.                     if(i == (countvalue - 1))
93.                     {
94.                         head_node = prev_node;
95.                     }
96.                     printf("\nDeleted Successfully \n\n");
97.                     break;
98.                 } else {
99.                     i++;
100.                    prev_node = temp_node;
101.                    temp_node = temp_node -> next;
102.                }
103.            }
104.        }
105.    } else
106.    printf("\nInvalid Position \n\n");
107. }
108.
109. void display() {
110.     int count = 0;
111.     temp_node = first_node;
112.     printf("\nDisplay Linked List : \n");
113.     while (temp_node != 0) {
114.         printf("# %d # ", temp_node->value);
115.         count++;
116.         temp_node = temp_node -> next;
```

```
117.      }
118.      printf("\nNo Of Items In Linked List : %d\n",
   count);
119.  }
120.
121.  int count() {
122.  int count = 0;
123.  temp_node = first_node;
124.  while (temp_node != 0) {
125.    count++;
126.    temp_node = temp_node -> next;
127.  }
128.  printf("\nNo Of Items In Linked List : %d\n",
   count);
129.  return count;
130. }
```

OUTPUT: -

```
117.     }
118.     printf("\nNo Of Items In Linked List : %d\n", count);
119. }
120.
121. int count() {
122.     int count = 0;
123.     temp_node = first_node;
124.     while (temp_node != 0) {
125.         count++;
126.         temp_node = temp_node->next;
127.     }
128.     printf("\nNo Of Items In Linked List : %d\n", count);
129.     return count;
130. }
131.
```

Feedback

Runtime error #stdin #stdout 0s 4320KB

comments (0)

stdin

copy

Standard input is empty

stdout

copy

Singly Linked List Example - All Operations

Options

1 : Insert into Linked List

2 : Delete from Linked List

3 : Display Linked List

4 : Count Linked List

Others : Exit()

Enter your option:

Options

1 : Insert into Linked List

2 : Delete from Linked List

3 : Display Linked List

4 : Count Linked List

Others : Exit()

EXPERIMENT 1

NAME:- Varun Verma

BRANCH:- EEE

ROLL NUMBER:- 02211504919

AIM: - Write a menu driven program to implement Linear Search and Binary Search

Program Code:-

```
/*write a menu driven program to do the searching
```

```
I) Linear Search
```

```
ii) Binary Search*/
```

```
#include<stdio.h>
```

```
void lsearch(int [],int);
```

```
void bsearch(int [],int);
```

```
int main()
```

```
{
```

```
int ch,ch1,m,n;
```

```
int a[10],b[10],i;
```

```
do
```

```
{
```

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

```
printf("\n1. linear search");
```

```
printf("\n2. binary search");
```

```
scanf("\n%d",&ch);
```

```
switch(ch)
```

```
{
```

```
case 1:printf("\n Enter Array size : ");

scanf("%d",&m);

for(i=0;i<m;i++)

{

    printf("\n Enter array element : ");

    scanf("%d",&a[i]);

}

lsearch(a,m);

break;

case 2: printf("\nEnter Array size : ");

scanf("%d",&n);

printf("\n Enter array element in sorted order : ");

for(i=0;i<n;i++)

{

    scanf("%d",&b[i]);

}

bsearch(b,n);

break;

default: printf("\n Wrong input !!! ");

}

printf("\n Want to be continue press non zero no : ");

scanf("%d",&ch1);

}

while(ch1!=0);
```

```
}

void lsearch(int a[],int m)
{
    int num,i,f=0;
    printf("\n Enter no to be search : ");
    scanf("%d",&num);
    for(i=0;i<m;i++)
    {
        if(a[i]==num)
        {
            printf("\n item is found at index %d !!!",i);
            f=1;
            break;
        }
    }
    if(f==0)
        printf("\n item is not found !!!");
}

void bsearch(int b[],int n)
{
    int beg=0,num,mid,last;
    last=n-1;
    printf("\n Enter no to be search : ");
    scanf("%d",&num);
    mid=(beg+last)/2;
```

```

while(num != b[mid] && beg <= last)

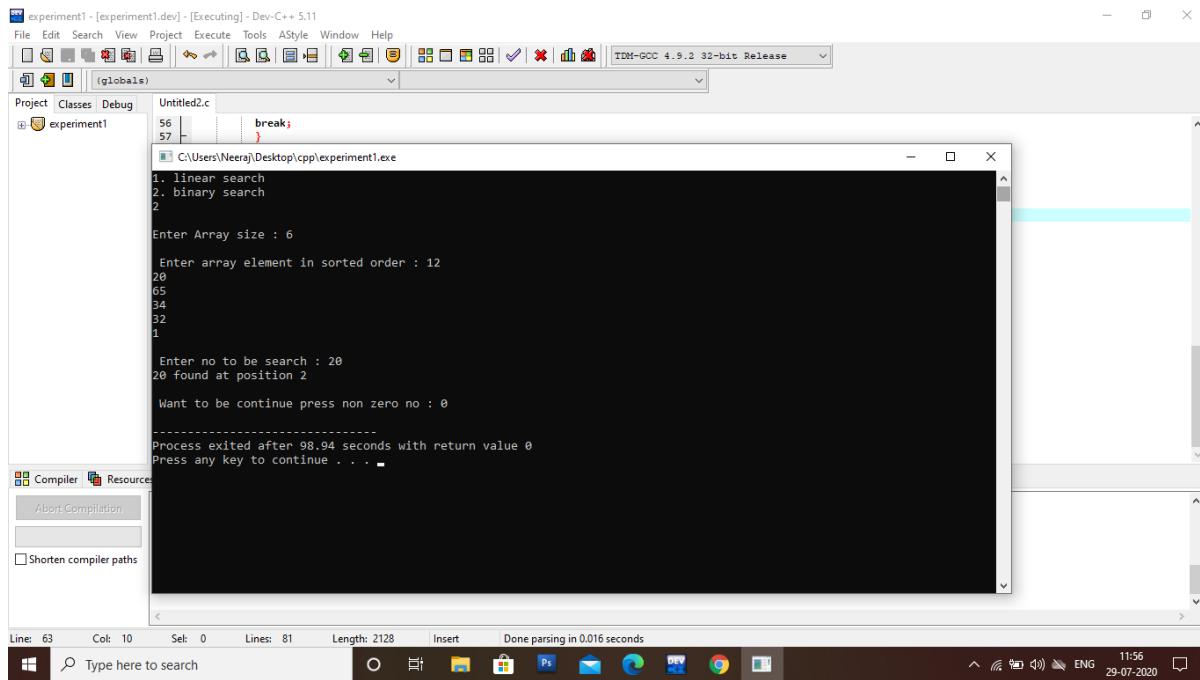
{
    if(num > b[mid])
        beg=mid+1;
    else
        last=mid-1;
    mid=(beg+last)/2;
}

if(num==b[mid])
    printf("%d found at position %d\n",num,mid+1);

if(beg>last)
    printf("%d not found in array\n",num);
}

```

OUTPUT:-



The screenshot shows the Dev-C++ IDE interface with the following details:

- Title Bar:** experiment1 - [experiment1.dev] - [Executing] - Dev-C++ 5.11
- Toolbar:** Standard Dev-C++ toolbar with various icons.
- Menu Bar:** File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, Help.
- Toolbox:** Standard Dev-C++ toolbox with icons for project, classes, debug, etc.
- Project Explorer:** Shows the project "experiment1" with file "Untitled2.c".
- Code Editor:** Displays the C code for binary search, with line 57 containing a closing brace `}` highlighted.
- Output Window:** Shows the execution of the program "C:\Users\Neeraj\Desktop\cpp\experiment1.exe". The output is:


```

1. linear search
2. binary search
2

Enter Array size : 6
Enter array element in sorted order : 12
20
65
34
32
1

Enter no to be search : 20
20 found at position 2

Want to be continue press non zero no : 0

Process exited after 98.94 seconds with return value 0
Press any key to continue . . .

```
- Compiler/Resource Window:** Shows compiler status with options like "Abort Compilation" and "Shorten compiler paths".
- System Taskbar:** Shows the Windows taskbar with various pinned icons (File Explorer, Edge, Mail, etc.) and system status (Network, Battery, Language, Date/Time).

experiment1 - [experiment1.dev] - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDN-GCC 4.9.2 32-bit Release

(globals)

Project Classes C:\Users\Neeraj\Desktop\cpp\experiment1.exe

experiment1

```
Enter your choice :
1. linear search
2. binary search
1

Enter Array size : 6
Enter array element : 20
Enter array element : 10
Enter array element : 31
Enter array element : 9
Enter array element : 7
Enter array element : 18
Enter no to be search : 18
item is found at index 5 !!!
Want to be continue press non zero no : 1
```

Compiler Abort Compiler

Shorten compiler paths

- Output Filename: C:\Users\Neeraj\Desktop\cpp\experiment1.exe
- Output Size: 102.3125 KiB
- Compilation Time: 0.25s

Line: 81 Col: 12 Sel: 0 Lines: 81 Length: 2128 Insert Done parsing in 0.016 seconds

11:54 29-07-2020

EXPERIMENT-2

Name :- Varun Verma

Roll no :- 02211504919

Branch :- EEE

Aim:-Write a menu driven program to:-

- (a) To insert an element in the array.**
- (b) To delete an element in the array.**

Answer (a):-

Program of insert of an element in the array:-

```
#include<stdio.h>
int main(){
    int arr[100]={0};
    int pos, n = 10, x, I;
    // initial array of size 10
    for(i = 0; i < 10; i++);
        arr[i]=i+1;
    //print the original array

    for (i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    // element to be inserted
    x = 50;

    // position at which element
    // is to be inserted
    pos = 5;

    // increase the size by 1
    n++;
```

```

// shift elements forward
for (i = n; i >= pos; i--)
    arr[i] = arr[i - 1];

// insert x at pos
arr[pos - 1] = x;

// print the updated array
for (i = 0; i < n; i++)
    printf("%d ", arr[i]);
printf("\n");

return 0;
}

```

(b)Program to delete an element from array

```

#include <stdio.h>

int main()
{
    int array[100], position, c, n;

    printf("Enter number of elements in array\n");
    scanf("%d", &n);

    printf("Enter %d elements\n", n);

    for ( c = 0 ; c < n ; c++ )
        scanf("%d", &array[c]);

    printf("Enter the location where you wish to delete element\n");
    scanf("%d", &position);

    if ( position >= n+1 )
        printf("Deletion not possible.\n");

```

```

else
{
    for ( c = position - 1 ; c < n - 1 ; c++ )
        array[c] = array[c+1];

    printf("Resultant array is\n");

    for( c = 0 ; c < n - 1 ; c++ )
        printf("%d\n", array[c]);
}
return 0;
}

```

OUTPUT

- (a) Output of insert an element in the array**
(b) Output of delete an element in the array

The screenshot shows a browser window with multiple tabs open. The active tab is titled "C program to Insert" and contains the following C code:

```

20.     for ( i = n; i >= pos; i-- )
21.         arr[i] = arr[i - 1];
22.
23.     // insert x at pos
24.     arr[pos - 1] = x;
25.
26.     // print the updated array
27.     for ( i = 0; i < n; i++)
28.         printf("%d ", arr[i]);
29.     printf("\n");
30.
31.     return 0;
32. }

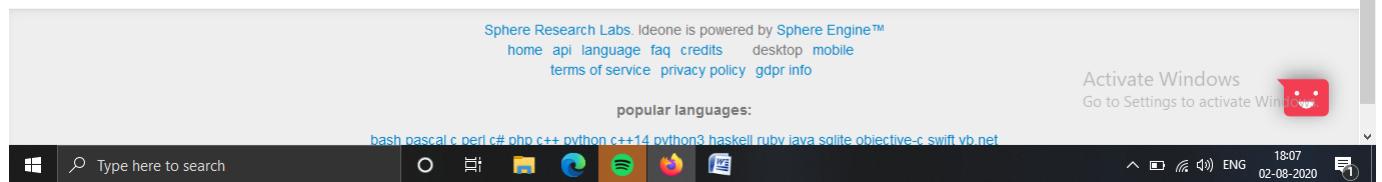
```

Below the code, it says "Success #stdin #stdout 0s 4296KB". The standard input is empty, and the standard output is:

```

1 2 3 4 5 6 7 8 9 10
1 2 3 4 50 5 6 7 8 9 10

```



The screenshot shows a web browser window with multiple tabs open. The active tab is titled "mze91A - Online C Compiler" and contains a C program to delete an element from an array. The code is as follows:

```
31 }  
32 }  
input Output  
Success #stdin #stdout 0s 4356KB  
Enter number of elements in array  
Enter 6 elements  
Enter the location where you wish to delete element  
Resultant array is  
1  
2  
4
```

Below the code, there are buttons for "save" and "ideone it!". To the right, there is a sidebar with options to "clear the output" and "syntax highlight". A tooltip provides information about visibility settings. Below the code area, there is a "Share or Embed source code" section with social sharing icons (Facebook, Twitter, Google+, VK, Email) and a link.

Under the code area, there is a message: "Time limit exceeded #stdin #stdout 5s 4408KB". To the right, there are links to "Sphere online judge" and "Learn How to Code". Further down, there are links to "Discover > Sphere Engine API" and "Discover > IDE Widget". At the bottom right, there is a Windows taskbar with the message "Activate Windows Go to Settings to activate Windows" and a red smiley face icon.