DS LAB FILE MAIN

1. Array and sum and average

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>

int main() {
    int marks[10],
     sum=0, avg,i,n;
    printf("Enter the value of n: ");
    scanf("%d" ,&n);
    printf("Enter the marks: ");
    for( i=0; i<n; i++)
    scanf("%d", &marks[i]);

    for (i = 0; i<n; i++)
    sum = sum + marks[i];
    printf("sum is %d\n" ,sum);
    avg = sum/n;
    printf("avg is %d\n" ,avg);



    return 0;
}
```

Output

```
Enter the value of n: 5
Enter the marks: 65
69
50
59
100
sum is 343
avg is 68


=== Code Execution Successful ===
```

2. trasversing

Programiz
C Online Compiler

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>

int main() {
    int n;
    int arr[10];
    printf("Enter the value of elements: ");
    scanf("%d", &n);

    printf("Enter %d elements: ", n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
        printf("Transversing a elements: ");
        for (int i = 0; i < n; i++)
            printf("%d\n" ,arr[i]);
      return 0;
}
```

Output

```
Enter the value of elements: 5
Enter 5 elements: 1
2
3
4
5
Transversing a elements: 1
2
3
4
5


=== Code Execution Successful ===
```

2. insertion

main.c

```c
int main() {
    int arr[10],n,value,K;
    printf("Enter the value of elements: ");
    scanf("%d", &n);

    printf("Enter %d elements: ", n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
        printf("Enter the value to insert: ");
        scanf("%d", &value);
    printf("Enter the position (0 to %d): ", n);
    scanf("%d", &K);
        if (K >= 0 && K<= n) {
        for (int i = n; i > K; i--) {
            arr[i] = arr[i - 1];
        }
        arr[K] = value;
        n++;
    }
     printf("new array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

      return 0;
}
```

Output

```
Enter the value of elements: 5
Enter 5 elements: 1
2
3
4
5
Enter the value to insert: 6
Enter the position (0 to 5): 3
new array: 1 2 3 6 4 5

=== Code Execution Successful ===
```

Programiz C Online Compiler

main.c

```c
#include <stdio.h>
int main() {
   int arr[10],n,value,K;
    printf("ENTER THE VALUE OF ELEMENT: ");
    scanf("%d", &n);
    printf("ENTER %d elements :", n );
    for(int i = 0; i <n; i++)
     scanf("%d", &arr[i]);
      printf("ENTER THE VALUE TO INSERT: ");
      scanf("%d", &value);
       printf("ENTER THE position (0 to %d): ", n);
       scanf("%d", &K);

          for (int i = n; i >K; i--) {
              arr[i] = arr[i-1];

      }
      arr[K] = value;

      printf("new array: ");
      for (int i = 0; i< n+1; i++) {
          printf("%d" , arr[i]);
      }
      return 0;

}
```

Output

```
ENTER THE VALUE OF ELEMENT: 5
ENTER 5 elements :1
2
3
4
5
ENTER THE VALUE TO INSERT: 100
ENTER THE position (0 to 5): 2
new array: 12100345

=== Code Execution Successful ===
```

DELETION

programiz.com/c-programming/online-compiler/

Programiz C Online Compiler

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>
int main() {
    int arr[10], N, VALUE, i;

    printf("Enter the number of elements: ");
    scanf("%d", &N);
printf("Enter %d elements: ", N);
    for (i = 0; i < N; i++) {
        scanf("%d", &arr[i]);
    }

    printf("\nVALUE to delete (0 to %d): ", N - 1);
    scanf("%d", &VALUE);

        for (i = VALUE; i < N - 1; i++) {
            arr[i] = arr[i + 1];
        }

        printf("NEW ARRAY: ");
        for (i = 0; i < N-1; i++) {
            printf("%d ", arr[i]);
        }


    return 0;
}
```

Output

```
Enter the number of elements: 5
Enter 5 elements: 1
2
3
4
5

VALUE to delete (0 to 4): 3
NEW ARRAY: 1 2 3 5

=== Code Execution Successful ===
```

binary to decimal

Programiz C Online Compiler

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>

void main(int num) {
    int binary[100];
    int index = 0;
printf("Enter a decimal number: ");
    scanf("%d", &num);

   while (num > 0) {
        binary[index] = num % 2;
        num = num / 2;
        index++;
    }

    printf("Binary: ");
    for (int j = index - 1; j >= 0; j--) {
        printf("%d", binary[j]);
    }
    printf("\n");
}
```

Output

```
Enter a decimal number: 7
Binary: 111


=== Code Exited With Errors ===
```

2D MULTIPLICATION

Programiz C Online Compiler

main.c

Share Run

```c
// Online C compiler to run C program online
#include <stdio.h>
int main() {
    int A[2][2] = { {1, 2}, {3, 4} };
    int B[2][2] = { {5, 6}, {7, 8} };
    int result[2][2];
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            result[i][j] = 0;
        }
    }
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            for (int k = 0; k < 2; k++) {
                result[i][j] += A[i][k] * B[k][j];
            }
        }
    }
    printf("Matrix result:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
}
```

Output

```
Matrix result:
19 22
43 50


=== Code Execution Successful ===
```

transposing

programiz.com/c-programming/online-compiler/

Programiz C Online Compiler

Programiz PRO >

main.c

Share Run

```c
#include <stdio.h>
int main() {
    int matrix[10][10], transpose[10][10];
    int row, col,i,j;
     // rows and columns
    printf("Enter the number of rows and columns: ");
    scanf("%d %d", &row, &col);
    printf("Enter elements of the matrix:\n");
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
    printf("Transpose of the matrix:\n");
    for (int i = 0; i < col; i++) {
        for (int j = 0; j < row; j++) {
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }
}
```

Output Clear

```
Enter the number of rows and columns: 2  2
Enter elements of the matrix:
1 2
3  4
Transpose of the matrix:
1 3
2 4


=== Code Execution Successful ===
```

stack push and pop

Programiz C Online Compiler

Programiz PRO ›

main.c

Share Run

```c
// Online C compiler to run C program online
#include <stdio.h>
#define MAX 5
int stack[MAX], top = -1;

void push() {
    int value;
    if (top == MAX - 1) {
        printf("Stack Overflow!\n");
        return;
    }
    printf("Enter the value to push: ");
    scanf("%d", &value);
    stack[++top] = value;
    printf("%d pushed onto the stack.\n", value);
}

void pop() {
    if (top == -1) {
        printf("Stack Underflow!\n");
        return;
    }
    printf("%d popped from the stack.\n", stack[top--]);
}
void display() {
    if (top == -1) {
```

Output

Clear

```
Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 1
Enter the value to push: 2
2 pushed onto the stack.

Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 1
Enter the value to push: 3
3 pushed onto the stack.

Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 2
3 popped from the stack.

Stack Operations:
 1.Push
```

Programiz C Online Compiler

Programiz PRO ›

main.c

Share Run

```c
        printf("Stack is empty.\n");
        return;
    }
    printf("Stack elements: ");
    for (int i = top; i >= 0; i--) {
        printf("%d ", stack[i]);
    }
    printf("\n");
}

int main() {
    int choice;
    while (1) {
        printf("\nStack Operations:\n");
        printf(" 1.Push\n 2.Pop\n 3.Display\n ");
        printf("Enter to print your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: push(); break;
            case 2: pop(); break;
            case 3: display(); break;
        }
    }
    return 0;
}
```

Output

Clear

```
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 1
Enter the value to push: 3
3 pushed onto the stack.

Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 2
3 popped from the stack.

Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice: 3
Stack elements: 2

Stack Operations:
 1.Push
 2.Pop
 3.Display
 Enter to print your choice:
```

queue

Programiz C Online Compiler

Programiz PRO ›

main.c

Share Run

```c
// Online C compiler to run C program online
#include <stdio.h>
#define SIZE 2
int queue[SIZE], front = -1, rear = -1;
void insert(int value) {
    if ((rear + 1) % SIZE == front) { // check full
        printf("Queue is full\n");
        return;
    }
    if (front == -1) front = 0;
    rear = (rear + 1) % SIZE;
    queue[rear] = value;
    printf("Inserted: %d\n", value);
}

void deletion() {
    if (front == -1 ) {
        printf("Queue is empty\n");
    } else {
        printf("Deleted %d\n", queue[front]);
        front++;
        if (front > rear) front = rear = -1;
    }
}

void display() {
```

Output

Clear

```
1. insert
2. deletion
3. Display.
Enter choice: 1
Enter value: 1
Inserted: 1

1. insert
2. deletion
3. Display.
Enter choice: 1
Enter value: 1
Inserted: 1

1. insert
2. deletion
3. Display.
Enter choice: 2
Deleted 1

1. insert
2. deletion
3. Display.
Enter choice: 1
Enter value: 2
```

Programiz C Online Compiler

Programiz PRO ›

main.c

Share Run

```c
    if (front == -1) {
        printf("Queue is empty!\n");
        return;
    }
   printf("Queue: ");
    int i = front;
    while (1) {
        printf("%d ", queue[i]);

        if (i == rear)
            break;

        i = (i + 1) % SIZE;
    }
    printf("\n");
}
int main() {
    int choice, value;
    while (1) {
        printf("\n1. insert\n2. deletion\n3. Display. \nEnter choice
            : ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter value: ");
                scanf("%d", &value);
```

Output

Clear

```
Enter value: 1
Inserted: 1

1. insert
2. deletion
3. Display.
Enter choice: 2
Deleted 1

1. insert
2. deletion
3. Display.
Enter choice: 1
Enter value: 2
Inserted: 2

1. insert
2. deletion
3. Display.
Enter choice: 3
Queue: 1 2

1. insert
2. deletion
3. Display.
Enter choice:
```

26°C Haze Search ENG IN 21:08 01-04-2025


```c
            : ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter value: ");
                scanf("%d", &value);
```


```c
               insert(value);
                break;
            case 2: deletion(); break;
            case 3: display(); break;

                return 0;

        }
    }
}
```


```
2. deletion
3. Display.
Enter choice: 1
Enter value: 2
Inserted: 2

1. insert
2. deletion
3. Display.
Enter choice: 3
Queue: 1 2

1. insert
2. deletion
3. Display.
Enter choice:
```


26°C Haze Search

Linked list operation

programiz.com/c-programming/online-compiler/

Programiz C Online Compiler

Programiz PRO >

main.c

Share Run

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

void insert();
void del();
void display();

struct linklist {
    int num;
    struct linklist *next;
};

struct linklist *ptr, *head, *start, *ptr1, *temp;

void main() {
    int choice;

    head = (struct linklist *)malloc(sizeof(struct linklist));


    printf("Enter value n: ");
    scanf("%d", &head->num);
    head->next = NULL;

    do {
        printf("\n1. Insert");
```

Output

Clear

```
Enter value n: 10

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 1

Enter value: 20

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 1

Enter value: 30

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 2
Deleted node value: 30

1. Insert
```

Programiz C Online Compiler

Programiz PRO >

main.c

Share Run

```c
        printf("\n2. Del");
        printf("\n3. Display");
        printf("\n4. Exit");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: insert(); break;
            case 2: del(); break;
            case 3: display(); break;
            case 4: exit(0);

        }
    } while (choice != 4);
}

void insert() {
    temp = (struct linklist *)malloc(sizeof(struct linklist));
    printf("\nEnter value: ");
    scanf("%d", &temp->num);
    temp->next = head;
    head = temp;
}

void del() {
    ptr = head;
```

Output

Clear

```
Enter your choice: 2
Deleted node value: 30

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 1

Enter value: 40

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 3
Linked List:
40
20
10

1. Insert
2. Del
3. Display
4. Exit
Enter your choice:
```

Finance headline
Oil tumbles over...

Search

ENG IN 21:19 03-04-2025

```c
    temp->next = head;
    head = temp;
}

void del() {
    ptr = head;
    head = head->next;
    printf("Deleted node value: %d\n", ptr->num);
    free(ptr);
}

void display() {
    ptr = head;
    printf("Linked List: \n");
    while (ptr != NULL) {
        printf("%d\n", ptr->num);
        ptr = ptr->next;
    }
}
```

```
Enter your choice: 1

Enter value: 40

1. Insert
2. Del
3. Display
4. Exit
Enter your choice: 3
Linked List:
40
20
10

1. Insert
2. Del
3. Display
4. Exit
Enter your choice:
```

stack through linked list

Programiz
C Online Compiler

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>
#include <stdlib.h>
void push();
void pop();
void display();

struct Stack {
    int data;
    struct Stack* next;
};
struct Stack *top = NULL, *temp;
void main() {
    int choice;

    do {
        printf("\n1. Push");
        printf("\n2. Pop");
        printf("\n3. Display");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: push(); break;
            case 2: pop(); break;
            case 3: display(); break;
```

Output

```
1. Push
2. Pop
3. Display
Enter your choice: 1

Enter value: 20
20 pushed

1. Push
2. Pop
3. Display
Enter your choice: 3
Stack elements:
20
10

1. Push
2. Pop
3. Display
Enter your choice: 2
Pop value: 20

1. Push
2. Pop
3. Display
Enter your choice:
```

Programiz C Online Compiler

main.c

```c
        }
    } while (choice != 4);
}
void push() {
    temp = (struct Stack*)malloc(sizeof(struct Stack));

    if (temp == NULL) {
        printf("Memory allocation failed!\n");
        return;
    }

    printf("\nEnter value: ");
    scanf("%d", &temp->data);

    temp->next = top;
    top = temp;

    printf("%d pushed\n", temp->data);
}
void pop() {
    if (top == NULL) {
        printf("Stack Underflow \n");
        return;
    }

    struct Stack* temp = top;
```

Output

```
1. Push
2. Pop
3. Display
Enter your choice: 1

Enter value: 20
20 pushed

1. Push
2. Pop
3. Display
Enter your choice: 3
Stack elements:
20
10

1. Push
2. Pop
3. Display
Enter your choice: 2
Pop value: 20

1. Push
2. Pop
3. Display
Enter your choice:
```

Programiz C Online Compiler

main.c

```c
        printf("Stack Underflow \n");
        return;
    }

    struct Stack* temp = top;
    printf("Pop value: %d\n", temp->data);
    top = top->next;
    free(temp);
}
void display() {
    struct Stack* temp = top;

    if (top == NULL) {
        printf("empty.\n");
        return;
    }

    printf("Stack elements:\n");
    while (temp != NULL) {
        printf("%d\n", temp->data);
        temp = temp->next;
    }
}
```

Output

```
1. Push
2. Pop
3. Display
Enter your choice: 1

Enter value: 20
20 pushed

1. Push
2. Pop
3. Display
Enter your choice: 3
Stack elements:
20
10

1. Push
2. Pop
3. Display
Enter your choice: 2
Pop value: 20

1. Push
2. Pop
3. Display
Enter your choice:
```

queue through link list

Programiz C Online Compiler

main.c

```c
// Online C compiler to run C program online
#include <stdio.h>
#include <stdlib.h>
void insert();
void del();
void display();

struct Queue {
    int data;
    struct Queue* next;
};
struct Queue *front = NULL, *rear = NULL, *temp;
void main() {
    int choice;

    do {
        printf("\n1. insert");
        printf("\n2. delete");
        printf("\n3. Display");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: insert(); break;
            case 2: del(); break;
            case 3: display(); break;
```

Output

```
1. insert
2. delete
3. Display
Enter your choice: 1

Enter value: 20
20 inset

1. insert
2. delete
3. Display
Enter your choice: 3
Queue elements:
10
20

1. insert
2. delete
3. Display
Enter your choice: 2
Deleted value: 10

1. insert
2. delete
3. Display
Enter your choice:
```

```c
        }
    } while (choice != 4);
}
    void insert () {
    temp = (struct Queue*)malloc(sizeof(struct Queue));

    if (temp == NULL) {
        printf("Memory allocation failed!\n");
        return;
    }

  printf("\nEnter value: ");
    scanf("%d", &temp->data);
    temp->next = NULL;
    if (rear == NULL) {
        front = rear = temp;
    } else {
        rear->next = temp;
        rear = temp;
    }
    printf("%d insert\n", temp->data);
}
void del() {
    if (front == NULL) {
        printf("Queue Underflow\n");
        return;
```

```
1. insert
2. delete
3. Display
Enter your choice: 1

Enter value: 20
20 inset

1. insert
2. delete
3. Display
Enter your choice: 3
Queue elements:
10
20

1. insert
2. delete
3. Display
Enter your choice: 2
Deleted value: 10

1. insert
2. delete
3. Display
Enter your choice:
```

main.c Share Run Output

```c
        printf("Queue Underflow\n");
        return;
    }
    temp = front;
    printf("Deleted value: %d\n", temp->data);
    front = front->next;
    if (front == NULL) {
        rear = NULL;
    }
    free(temp);
}
void display() {
    temp = front;

    if (front == NULL) {
        printf("Queue is empty.\n");
        return;
    }

    printf("Queue elements:\n");
    while (temp != NULL) {
        printf("%d\n", temp->data);
        temp = temp->next;
    }
}

```

```
2. delete
3. Display
Enter your choice: 1

Enter value: 20
20 inset

1. insert
2. delete
3. Display
Enter your choice: 3
Queue elements:
10
20

1. insert
2. delete
3. Display
Enter your choice: 2
Deleted value: 10

1. insert
2. delete
3. Display
Enter your choice:
=== Session Ended. Please Run the code again ===
```

Concatenation

Programiz C Online Compiler

main.c

Share Run

```c
#include <stdio.h>
#include <stdlib.h>
struct Node {
    int data;
    struct Node* next;
};
struct Node* createNode(int data) {
struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}
void apNode(struct Node** head, int data) {
    struct Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }
    struct Node* temp = *head;
    while (temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = newNode;
}
void concatenate(struct Node** head1, struct Node* head2) {
    if (*head1 == NULL) {
```

Output

```
 elements in List 1: 1
Enter value 1: 2
 elements in List 2: 4
Enter value 1: 1
Enter value 2: 2
Enter value 3: 3
Enter value 4: 4

Concatenated List: 2 -> 1 -> 2 -> 3 -> 4 -> NULL


=== Code Execution Successful ===
```

Programiz C Online Compiler

main.c

Share Run

```c
        *head1 = head2;
        return;
    }
    struct Node* temp = *head1;
    while (temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = head2;
}
void printList(struct Node* head) {
    while (head != NULL) {
        printf("%d -> ", head->data);
        head = head->next;
    }
    printf("NULL\n");
}
void inputList(struct Node** head, int n) {
    int val;
    for (int i = 0; i < n; i++) {
        printf("Enter value %d: ", i + 1);
        scanf("%d", &val);
        apNode(head, val);
    }
}
int main() {
    struct Node* list1 = NULL;
```

```
        apNode(head, val);
    }
}
int main() {
    struct Node* list1 = NULL;
    struct Node* list2 = NULL;
    int n1, n2;
    printf(" elements in List 1: ");
    scanf("%d", &n1);
    inputList(&list1, n1);

    printf(" elements in List 2: ");
    scanf("%d", &n2);
    inputList(&list2, n2);

    concatenate(&list1, list2);

    printf("\nConcatenated List: ");
    printList(list1);

    return 0;
}

```

tree traversal

main.c

```c
#include <stdio.h>
#include <stdlib.h>
struct tree {
    int data;
    struct tree *lchild, *rchild;
};

struct tree *insert(struct tree *p, int n);
void inorder(struct tree *p);
void preorder(struct tree *p);
void postorder(struct tree *p);

int main() {
    int x, y, i;
    struct tree *root = NULL;

    printf(" number of nodes in the tree: ");
    scanf("%d", &x);

    while (x-- > 0) {
        printf(" data part of the node: ");
        scanf("%d", &y);
        root = insert(root, y);
    }

    printf("\nEnter the traversal you want\n");
```

Output

```
 number of nodes in the tree: 3
 data part of the node: 60
 data part of the node: 55
 data part of the node: 70

Enter the traversal you want
1. Inorder
2. Preorder
3. Postorder
1
 inorder traversal is:
55 60 70

=== Code Execution Successful ===
```

main.c

```c
    printf("1. Inorder\n2. Preorder\n3. Postorder\n");
    scanf("%d", &i);

    switch (i) {
        case 1:
            printf(" inorder traversal is:\n");
            inorder(root);
            break;
        case 2:
            printf("preorder traversal is:\n");
            preorder(root);
            break;
        case 3:
            printf(" postorder traversal is:\n");
            postorder(root);
            break;
    }
}

//  insert node
struct tree *insert(struct tree *p, int n) {
    if (p == NULL) {
        struct tree *newNode = (struct tree *)malloc(sizeof(struct
            tree));
        newNode->data = n;
```

Output

```
 number of nodes in the tree: 3
 data part of the node: 60
 data part of the node: 55
 data part of the node: 70

Enter the traversal you want
1. Inorder
2. Preorder
3. Postorder
1
 inorder traversal is:
55 60 70

=== Code Execution Successful ===
```

main.c

```c
        tree));
        newNode->data = n;
        newNode->lchild = newNode->rchild = NULL;
        return newNode;
    }

    if (n < p->data)
        p->lchild = insert(p->lchild, n);
    else
        p->rchild = insert(p->rchild, n);

    return p;
}

// Inorder
void inorder(struct tree *p) {
    if (p != NULL) {
        inorder(p->lchild);
        printf("%d ", p->data);
        inorder(p->rchild);
    }
}
// Preorder
void preorder(struct tree *p) {
    if (p != NULL) {
        printf("%d ", p->data);
```

```c
        printf("%d ", p->data);
        inorder(p->rchild);
    }
}
// Preorder
void preorder(struct tree *p) {
    if (p != NULL) {
        printf("%d ", p->data);
        preorder(p->lchild);
        preorder(p->rchild);
    }
}
// Postorder
void postorder(struct tree *p) {
    if (p != NULL) {
        postorder(p->lchild);
        postorder(p->rchild);
        printf("%d ", p->data);
    }
}
```

```
1. Inorder
2. Preorder
3. Postorder
1
 inorder traversal is:
55 60 70

=== Code Execution Successful ===
```

tree searching (sorted)

Programiz C Online Compiler

main.c

```c
#include <stdio.h>
int Search(int DATA[], int LB, int UB, int ITEM) {
    int BEG = LB, END = UB, MID;
    int LOC = -1;
    while (BEG <= END) {
        MID = (BEG + END) / 2;
        if (ITEM < DATA[MID]) {
            END = MID - 1;
        } else if (ITEM > DATA[MID]) {
            BEG = MID + 1;
        } else {
            LOC = MID; break;
        } }
    return LOC; }
int main() {
    int DATA[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91};
    int n = sizeof(DATA) / sizeof(DATA[0]);
    int ITEM, LOC;
    printf("Item to search: ");
    scanf("%d", &ITEM);
    LOC = Search(DATA, 0, n - 1, ITEM);
    if (LOC != -1)
        printf("Item found : %d\n", LOC + 1);
    else
        printf("Item not found\n");
    return 0;
```

Share Run Output

```
Item to search: 16
Item found : 5


=== Code Execution Successful ===
```

37°C Partly sunny Search