

DATA STRUCTURES DAY 2

T. Sai krishna(192211870)

22) Write a C programme to implement array using stack operations.

The screenshot shows the Dev-C++ IDE interface with the code for a C program to implement a stack using an array. The code includes functions for Push, Pop, and Display operations. A terminal window shows the execution of the program, demonstrating the stack's behavior with various inputs like 'PUSH', 'POP', and 'DISPLAY'.

```
#include<stdio.h>
#define max 5
int main()
{
    int stack[max];
    int Top = -1, element,i;
    char ch;
    printf("Enter the choice (1)> PUSH, (2)> POP, (3)> DISPLAY: ");
    do
    {
        switch(ch)
        {
            case 1:
                printf("Enter element :");
                scanf("%d",&element);
                if(Top==max-1)
                    printf("You can't push since the stack is full");
                else
                    Top = Top + 1;
                    stack[Top] = element;
                break;
            case 2:
                if(Top==0)
                    printf("You can't pop since the stack is empty");
                else
                    printf("popped element = %d",stack[Top]);
                    Top = Top - 1;
                break;
            case 3:
                printf("Elements in the stack is : ");
                if(Top==0)
                    printf("No elements are present in the stack!");
                else
                    for(i=0;i<Top;i++)
                        printf("%d ",stack[i]);
                break;
            default:
                printf("INVALID CHOICE!");
        }
        printf("\nEnter the choice (1)> PUSH, (2)> POP, (3)> DISPLAY: ");
        scanf("%d",&ch);
    }while(ch!=3);
}
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\C programme to implement array using stack operations..exe
- Output Size: 128.6591756875 KiB
- Compilation Time: 0.25s

Line: 52 Col: 2 Sel: 0 Lines: 52 Length: 1072 Insert Done parsing in 0.015 seconds

The screenshot shows the Dev-C++ IDE interface with the same C program for stack implementation. The code structure is identical to the first one, but the logic for displaying the stack elements is implemented differently using a loop. The terminal window shows the program's execution and output.

```
#include<stdio.h>
#define max 5
int main()
{
    int stack[max];
    int Top = -1, element,i;
    char ch;
    printf("Enter the choice (1)> PUSH, (2)> POP, (3)> DISPLAY: ");
    do
    {
        switch(ch)
        {
            case 1:
                printf("Enter element :");
                scanf("%d",&element);
                Top = Top + 1;
                stack[Top] = element;
                break;
            case 2:
                if(Top==0)
                    printf("No elements are present in the stack!");
                else
                    for(i=0;i<Top;i++)
                        printf("%d ",stack[i]);
                break;
            case 3:
                printf("Elements in the stack is : ");
                if(Top==0)
                    printf("No elements are present in the stack!");
                else
                    for(i=0;i<Top;i++)
                        printf("%d ",stack[i]);
                break;
            default:
                printf("INVALID CHOICE!");
        }
        printf("\nEnter the choice (1)> PUSH, (2)> POP, (3)> DISPLAY: ");
        scanf("%d",&ch);
    }while(ch!=3);
}
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\C programme to implement array using stack operations..exe
- Output Size: 128.6591756875 KiB
- Compilation Time: 0.25s

Line: 52 Col: 2 Sel: 0 Lines: 52 Length: 1072 Insert Done parsing in 0.015 seconds

23) Write a C programme to implement data structures.

The screenshot shows the Dev-C++ IDE interface. The main window displays a C program for a queue implemented using an array. The code includes logic for enqueueing elements (rear++), dequeuing elements (front++), and displaying the queue. A terminal window shows the execution of the program, with user inputs and corresponding outputs for enqueueing, dequeuing, and displaying the queue's contents.

```
#include<stdio.h>
#define max 5
int main()
{
    int front = -1, rear = -1, ch, element, i;
    char ch1;
    printf("\nEnter the choice (1)> ENQUEUE, (2)> DEQUEUE, (3)> DISPLAY: ");
    scanf("%d", &ch);
    do
    {
        switch(ch)
        {
            case 1:
                if(front == max)
                    printf("You can't enqueue since the queue is full");
                else if(rear == max - 1)
                    printf("You can't enqueue since the queue is full");
                else if(front == 0)
                    front = 0;
                rear = 0;
                queue[rear] = element;
            case 2:
                if(front == -1)
                    printf("You can't dequeue since the queue is empty");
                else
                    rear = rear + 1;
                    queue[rear] = element;
            case 3:
                printf("Elements in the queue are : ");
                if(front == -1)
                    printf("No elements are present in the queue!");
                else
                    for(i=front;i<=rear;i++)
                        printf("%d ",queue[i]);
                break;
            default:
                printf("INVALID CHOICE!");
        }
        printf("\nEnter the choice (1)> ENQUEUE, (2)> DEQUEUE, (3)> DISPLAY: ");
        scanf("%d", &ch);
    }while(ch<=3);
    return 0;
}
```

The screenshot shows the Dev-C++ IDE interface again. This time, the code has been modified to use a while loop instead of a do-while loop. The logic for enqueueing and dequeuing remains the same. The terminal window shows the execution of the program, with user inputs and corresponding outputs for enqueueing, dequeuing, and displaying the queue's contents.

```
#include<stdio.h>
#define max 5
int main()
{
    int front = -1, rear = -1, ch, element, i;
    char ch1;
    printf("\nEnter the choice (1)> ENQUEUE, (2)> DEQUEUE, (3)> DISPLAY: ");
    scanf("%d", &ch);
    while(ch<=3)
    {
        switch(ch)
        {
            case 1:
                if(front == max)
                    printf("You can't enqueue since the queue is full");
                else if(rear == max - 1)
                    printf("You can't enqueue since the queue is full");
                else if(front == 0)
                    front = 0;
                rear = 0;
                queue[rear] = element;
            case 2:
                if(front == -1)
                    printf("You can't dequeue since the queue is empty");
                else
                    rear = rear + 1;
                    queue[rear] = element;
            case 3:
                printf("Elements in the queue are : ");
                if(front == -1)
                    printf("No elements are present in the queue!");
                else
                    for(i=front;i<=rear;i++)
                        printf("%d ",queue[i]);
                break;
            default:
                printf("INVALID CHOICE!");
        }
        printf("\nEnter the choice (1)> ENQUEUE, (2)> DEQUEUE, (3)> DISPLAY: ");
        scanf("%d", &ch);
    }
    return 0;
}
```

24) Doubly linked List by using C programming.

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

- Project:** Doubly linked List by using C programming
- Code Editor:** Displays the source code for a doubly linked list implementation. The code includes functions for creating nodes, inserting elements at the beginning and end, searching for elements, and deleting elements.
- Compiler Log:** Shows compilation results with 0 errors and 0 warnings. The output file is C:\Users\saiKr\OneDrive\Desktop\ Doubly linked List by using C programming.exe, size 131.7294921875 KB, and compilation time 0.22s.
- Terminal:** Displays the execution of the program. It shows the menu options and user interactions for inserting elements 12 and 5 at the beginning of the list.
- System Tray:** Shows the date (09-05-2023), time (12:26), and weather (35°C, Mostly sunny).

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

- Project:** Doubly linked List by using C programming
- Code Editor:** Displays the source code for a more advanced doubly linked list implementation. It includes search functionality based on data values.
- Compiler Log:** Shows compilation results with 0 errors and 0 warnings. The output file is C:\Users\saiKr\OneDrive\Desktop\ Doubly linked List by using C programming.exe, size 131.7294921875 KB, and compilation time 0.22s.
- Terminal:** Displays the execution of the program. It shows the menu options and user interactions for inserting element 21 at the beginning, deleting element 21, and searching for element 21.
- System Tray:** Shows the date (09-05-2023), time (12:28), and weather (35°C, Mostly sunny).

C:\Users\saikr\OneDrive\Desktop>Doubly linked List by using C programming.cpp - [Executing] - Dev-C++ 5.11

```

1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 2
Enter the element to insert at the beginning: 21
Element 21 inserted at the beginning.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 5
Enter the element to delete: 21
Element 21 deleted.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 6
Doubly linked list: 5 12 19

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 7
Exiting program.

Process exited after 80.01 seconds with return value 0
Press any key to continue . .

```

35°C
Mostly sunny

C:\Users\saikr\OneDrive\Desktop>Doubly linked List by using C programming.cpp - [Executing] - Dev-C++ 5.11

```

1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 2
Enter the element to insert at the beginning: 21
Element 21 inserted at the beginning.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 5
Enter the element to delete: 21
Element 21 deleted.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 6
Doubly linked list: 5 12 19

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 7
Exiting program.

Process exited after 80.01 seconds with return value 0
Press any key to continue . .

35°C  
Mostly sunny
```

The screenshot shows the Dev-C++ IDE interface. The left pane displays the C code for a Doubly linked List. The right pane shows the terminal window where the program is executed. The terminal output includes the menu options for the Doubly linked List, user inputs, and the resulting list structure.

```

C:\Users\saikr\OneDrive\Desktop>Doubly linked List by using C programming.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools ASyntax Window Help
Project Classes Debug [*] binary search tree.cpp [*] Doubly linked List by using C programming.cpp Doubly linked List by using C programming.exe
File Node struct
File Node struct
createNode(int data)
deleteNode(Node** head)
displayList(Node** head)
insertAfter(Node** head, int data)
insertAtBeginning(Node** head, int data)
insertAtEnd(Node** head, int data)
main() int

132     break;
133 case 2:
134     printf("Enter the element to insert at the beginning: ");
135     scanf("%d", &data);
136     insertAtBeginning(&head, data);
137     break;
138 case 3:
139     printf("Enter the element to insert at the end: ");
140     scanf("%d", &data);
141     insertAtEnd(&head, data);
142     break;
143 case 4:
144     printf("Enter the element to insert after: ");
145     scanf("%d", &searchData);
146     printf("Enter the element to insert: ");
147     scanf("%d", &data);
148     insertAfter(head, searchData, data);
149     break;
150 case 5:
151     printf("Enter the element to delete: ");
152     scanf("%d", &data);
153     deleteNode(&head, data);
154     break;
155 case 6:
156     displayList(head);
157     break;
158 case 7:
159     printf("Exiting program.\n");
160     return 0;
161 default:
162     printf("Invalid choice! Please try again.\n");
163     break;
164 }
165 }
166
167 return 0;
168 }

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\Doubly linked List by using C programming.exe
- Output Size: 131.7294921875 KIB
- Compilation Time: 0.22s

C:\Users\saikr\OneDrive\Desktop>Doubly linked List by using C programming.exe
---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 2
Enter the element to insert at the beginning: 21
Element 21 inserted at the beginning.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 5
Enter the element to delete: 21
Element 21 deleted.

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 6
Doubly linked list: 5 12 19

---- Doubly Linked List Menu ----
1. Search Element
2. Insert at Beginning
3. Insert at End
4. Insert after Element
5. Delete Element
6. Display List
7. Exit

Enter your choice: 7
Exiting program.

Process exited after 80.01 seconds with return value 0
Press any key to continue . . .

```

25)Singly linked List by using C programming.

The screenshot shows the Dev-C++ IDE interface. The left pane displays the C code for a Singly linked List. The right pane shows the terminal window where the program is executed. The terminal output includes the menu options for the Singly linked List, user inputs, and the resulting list structure.

```

C:\Users\saikr\OneDrive\Desktop\singly linked.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools ASyntax Window Help
Project Classes Debug [*] C programme to find factorial of given numbers using functions with recursion.cpp singly linked.exe
File Node struct
File Node struct
deleteNode(struct Node**)
insertAtBeginning(struct Node**, int)
insertAtEnd(struct Node**, int)
insertAtPosition(struct Node**, int, int)
main() int
printList(struct Node*)
search(struct Node*)
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\singly linked.exe
- Output Size: 131.6975 KIB
- Compilation Time: 2.13s

C:\Users\saikr\OneDrive\Desktop\singly linked.exe
*** Singly Linked List Operations ***
1. Insert at the beginning
2. Insert at the end
3. Insert at a specified position
4. Delete a node
5. Search for an element
6. Print the list
7. Exit

Enter your choice: 1
Enter the element to be inserted at the beginning: 12

*** Singly Linked List Operations ***
1. Insert at the beginning
2. Insert at the end
3. Insert at a specified position
4. Delete a node
5. Search for an element
6. Print the list
7. Exit

Enter your choice: 2
Enter the element to be inserted at the end: 5

*** Singly Linked List Operations ***
1. Insert at the beginning
2. Insert at the end
3. Insert at a specified position
4. Delete a node
5. Search for an element
6. Print the list
7. Exit

Enter your choice: 7
The list is: 12 5

*** Singly Linked List Operations ***
1. Insert at the beginning
2. Insert at the end
3. Insert at a specified position
4. Delete a node
5. Search for an element
6. Print the list
7. Exit

Enter your choice: 7
The list is: 12 5

Process exited after 41.8 seconds with return value 0
Press any key to continue . . .

```

The screenshot shows the Dev-C++ IDE interface. The code editor displays a C program named singly linked.cpp. The code implements a singly linked list with functions for insertion at beginning, end, and position, deletion, search, and printing. The compiler log shows no errors or warnings, and the output file singly linked.exe is generated. The system tray indicates it's 30°C and mostly cloudy.

```
[1] C programme to find factorial of given numbers using functions with recursion.cpp singly linked.cpp
```

```
Project Classes Debug [1] C programme to find factorial of given numbers using functions with recursion.cpp singly linked.cpp
Node struct
└ deleteNode(struct Node** head, int data);
  insertAtBeginning(struct Node** head, int data);
  insertAtEnd(struct Node** head, int data);
  insertAtPosition(struct Node** head, int data, int position);
main() int
printList(struct Node* head);
search(struct Node* head, int data);
```

```
37     insertAtBeginning(head, data);
38     break;
39   case 2:
40     printf("Enter the element to be inserted at the end: ");
41     scanf("%d", &data);
42     insertAtEnd(head, data);
43     break;
44   case 3:
45     printf("Enter the element to be inserted: ");
46     scanf("%d", &data);
47     printf("Enter the position where the element should be inserted: ");
48     scanf("%d", &position);
49     insertAtPosition(head, data, position);
50     break;
51   case 4:
52     printf("Enter the element to be deleted: ");
53     scanf("%d", &data);
54     deleteNode(&head, data);
55     break;
56   case 5:
57     printf("Enter the element to be searched: ");
58     scanf("%d", &data);
59     if (search(head, data) != NULL) {
60       printf("Element found in the list.\n");
61     } else {
62       printf("Element not found in the list.\n");
63     }
64     break;
65   case 6:
66     printf("The list is: ");
67     printList(head);
68     printf("\n");
69     break;
70   case 7:
71     exit(0);
72   default:
73     printf("Invalid choice! Please enter a valid option.\n");
```

```
Compiler Resources Compile Log Debug Find Results Close
```

```
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\singly linked.exe
- Output Size: 131.6875 Kib
- Compilation Time: 2.13s
```

```
30°C Mostly cloudy
```

The screenshot shows the Dev-C++ IDE interface. The code editor displays the same C program as before, but with a bug in the search function. The compiler log shows no errors or warnings, and the output file singly linked.exe is generated. The system tray indicates it's 30°C and mostly cloudy.

```
[1] C programme to find factorial of given numbers using functions with recursion.cpp singly linked.cpp
```

```
Project Classes Debug [1] C programme to find factorial of given numbers using functions with recursion.cpp singly linked.cpp
Node struct
└ deleteNode(struct Node** head, int data);
  insertAtBeginning(struct Node** head, int data);
  insertAtEnd(struct Node** head, int data);
  insertAtPosition(struct Node** head, int data, int position);
main() int
printList(struct Node* head);
search(struct Node* head, int data);
```

```
71     printf("Invalid choice! Please enter a valid option.\n");
72   }
73   return 0;
74 }
```

```
75 }
```

```
76 }
```

```
77 }
```

```
78 }
```

```
79 }
```

```
80 void printList(struct Node* head) {
81   struct Node* temp = head;
82   while (temp != NULL) {
83     printf("%d ", temp->data);
84     Temp = temp->next;
85   }
86 }
```

```
87 }
```

```
88 void insertAtBeginning(struct Node** head, int data) {
89   struct Node* newnode = (struct Node*)malloc(sizeof(struct Node));
90   newnode->data = data;
91   newnode->next = *head;
92   *head = newnode;
93 }
```

```
94 }
```

```
95 void insertAtEnd(struct Node** head, int data) {
96   struct Node* newnode = (struct Node*)malloc(sizeof(struct Node));
97   newnode->data = data;
98   newnode->next = NULL;
99 }
```

```
100 if (*head == NULL) {
101   *head = newnode;
102   return;
103 }
104 struct Node* temp = *head;
105 while (temp->next != NULL) {
106   temp = temp->next;
107 }
```

```
108 }
```

```
109 }
```

```
110 }
```

```
111 }
```

```
112 }
```

```
113 }
```

```
114 }
```

```
115 }
```

```
116 }
```

```
117 }
```

```
118 }
```

```
119 }
```

```
120 }
```

```
121 }
```

```
122 }
```

```
123 }
```

```
124 }
```

```
125 }
```

```
126 }
```

```
127 }
```

```
128 }
```

```
129 }
```

```
130 }
```

```
131 }
```

```
132 }
```

```
133 }
```

```
134 }
```

```
135 }
```

```
136 }
```

```
137 }
```

```
138 }
```

```
139 }
```

```
140 }
```

```
141 }
```

```
142 }
```

```
143 }
```

```
144 }
```

```
145 }
```

```
146 }
```

```
147 }
```

```
148 }
```

```
149 }
```

```
150 }
```

```
151 }
```

```
152 }
```

```
153 }
```

```
154 }
```

```
155 }
```

```
156 }
```

```
157 }
```

```
158 }
```

```
159 }
```

```
160 }
```

```
161 }
```

```
162 }
```

```
163 }
```

```
164 }
```

```
165 }
```

```
166 }
```

```
167 }
```

```
168 }
```

```
169 }
```

```
170 }
```

```
171 }
```

```
172 }
```

```
173 }
```

```
174 }
```

```
175 }
```

```
176 }
```

```
177 }
```

```
178 }
```

```
179 }
```

```
180 }
```

```
181 }
```

```
182 }
```

```
183 }
```

```
184 }
```

```
185 }
```

```
186 }
```

```
187 }
```

```
188 }
```

```
189 }
```

```
190 }
```

```
191 }
```

```
192 }
```

```
193 }
```

```
194 }
```

```
195 }
```

```
196 }
```

```
197 }
```

```
198 }
```

```
199 }
```

```
200 }
```

```
201 }
```

```
202 }
```

```
203 }
```

```
204 }
```

```
205 }
```

```
206 }
```

```
207 }
```

```
208 }
```

```
209 }
```

```
210 }
```

```
211 }
```

```
212 }
```

```
213 }
```

```
214 }
```

```
215 }
```

```
216 }
```

```
217 }
```

```
218 }
```

```
219 }
```

```
220 }
```

```
221 }
```

```
222 }
```

```
223 }
```

```
224 }
```

```
225 }
```

```
226 }
```

```
227 }
```

```
228 }
```

```
229 }
```

```
230 }
```

```
231 }
```

```
232 }
```

```
233 }
```

```
234 }
```

```
235 }
```

```
236 }
```

```
237 }
```

```
238 }
```

```
239 }
```

```
240 }
```

```
241 }
```

```
242 }
```

```
243 }
```

```
244 }
```

```
245 }
```

```
246 }
```

```
247 }
```

```
248 }
```

```
249 }
```

```
250 }
```

```
251 }
```

```
252 }
```

```
253 }
```

```
254 }
```

```
255 }
```

```
256 }
```

```
257 }
```

```
258 }
```

```
259 }
```

```
260 }
```

```
261 }
```

```
262 }
```

```
263 }
```

```
264 }
```

```
265 }
```

```
266 }
```

```
267 }
```

```
268 }
```

```
269 }
```

```
270 }
```

```
271 }
```

```
272 }
```

```
273 }
```

```
274 }
```

```
275 }
```

```
276 }
```

```
277 }
```

```
278 }
```

```
279 }
```

```
280 }
```

```
281 }
```

```
282 }
```

```
283 }
```

```
284 }
```

```
285 }
```

```
286 }
```

```
287 }
```

```
288 }
```

```
289 }
```

```
290 }
```

```
291 }
```

```
292 }
```

```
293 }
```

```
294 }
```

```
295 }
```

```
296 }
```

```
297 }
```

```
298 }
```

```
299 }
```

```
300 }
```

```
301 }
```

```
302 }
```

```
303 }
```

```
304 }
```

```
305 }
```

```
306 }
```

```
307 }
```

```
308 }
```

```
309 }
```

```
310 }
```

```
311 }
```

```
312 }
```

```
313 }
```

```
314 }
```

```
315 }
```

```
316 }
```

```
317 }
```

```
318 }
```

```
319 }
```

```
320 }
```

```
321 }
```

```
322 }
```

```
323 }
```

```
324 }
```

```
325 }
```

```
326 }
```

```
327 }
```

```
328 }
```

```
329 }
```

```
330 }
```

```
331 }
```

```
332 }
```

```
333 }
```

```
334 }
```

```
335 }
```

```
336 }
```

```
337 }
```

```
338 }
```

```
339 }
```

```
340 }
```

```
341 }
```

```
342 }
```

```
343 }
```

```
344 }
```

```
345 }
```

```
346 }
```

```
347 }
```

```
348 }
```

```
349 }
```

```
350 }
```

```
351 }
```

```
352 }
```

```
353 }
```

```
354 }
```

```
355 }
```

```
356 }
```

```
357 }
```

```
358 }
```

```
359 }
```

```
360 }
```

```
361 }
```

```
362 }
```

```
363 }
```

```
364 }
```

```
365 }
```

```
366 }
```

```
367 }
```

```
368 }
```

```
369 }
```

```
370 }
```

```
371 }
```

```
372 }
```

```
373 }
```

```
374 }
```

```
375 }
```

```
376 }
```

```
377 }
```

```
378 }
```

```
379 }
```

```
380 }
```

```
381 }
```

```
382 }
```

```
383 }
```

```
384 }
```

```
385 }
```

```
386 }
```

```
387 }
```

```
388 }
```

```
389 }
```

```
390 }
```

```
391 }
```

```
392 }
```

```
393 }
```

```
394 }
```

```
395 }
```

```
396 }
```

```
397 }
```

```
398 }
```

```
399 }
```

```
400 }
```

```
401 }
```

```
402 }
```

```
403 }
```

```
404 }
```

```
405 }
```

```
406 }
```

```
407 }
```

```
408 }
```

```
409 }
```

```
410 }
```

```
411 }
```

```
412 }
```

```
413 }
```

```
414 }
```

```
415 }
```

```
416 }
```

```
417 }
```

```
418 }
```

```
419 }
```

```
420 }
```

```
421 }
```

```
422 }
```

```
423 }
```

```
424 }
```

```
425 }
```

```
426 }
```

```
427 }
```

```
428 }
```

```
429 }
```

```
430 }
```

```
431 }
```

```
432 }
```

```
433 }
```

```
434 }
```

```
435 }
```

```
436 }
```

```
437 }
```

```
438 }
```

```
439 }
```

```
440 }
```

```
441 }
```

```
442 }
```

```
443 }
```

```
444 }
```

```
445 }
```

```
446 }
```

```
447 }
```

```
448 }
```

```
449 }
```

```
450 }
```

```
451 }
```

```
452 }
```

```
453 }
```

```
454 }
```

```
455 }
```

```
456 }
```

```
457 }
```

```
458 }
```

```
459 }
```

```
460 }
```

```
461 }
```

```
462 }
```

```
463 }
```

```
464 }
```

```
465 }
```

```
466 }
```

```
467 }
```

```
468 }
```

```
469 }
```

```
470 }
```

```
471 }
```

```
472 }
```

```
473 }
```

```
474 }
```

```
475 }
```

```
476 }
```

```
477 }
```

```
478 }
```

```
479 }
```

```
480 }
```

```
481 }
```

```
482 }
```

```
483 }
```

```
484 }
```

```
485 }
```

```
486 }
```

```
487 }
```

```
488 }
```

```
489 }
```

```
490 }
```

```
491 }
```

```
492 }
```

```
493 }
```

```
494 }
```

```
495 }
```

```
496 }
```

```
497 }
```

```
498 }
```

```
499 }
```

```
500 }
```

```
501 }
```

```
502 }
```

```
503 }
```

```
504 }
```

```
505 }
```

```
506 }
```

```
507 }
```

```
508 }
```

```
509 }
```

```
510 }
```

```
511 }
```

```
512 }
```

```
513 }
```

```
514 }
```

```
515 }
```

```
516 }
```

```
517 }
```

```
518 }
```

```
519 }
```

```
520 }
```

```
521 }
```

```
522 }
```

```
523 }
```

```
524 }
```

```
525 }
```

```
526 }
```

```
527 }
```

```
528 }
```

```
529 }
```

```
530 }
```

```
531 }
```

```
532 }
```

```
533 }
```

```
534 }
```

```
535 }
```

```
536 }
```

```
537 }
```

```
538 }
```

```
539 }
```

```
540 }
```

```
541 }
```

```
542 }
```

```
543 }
```

```
544 }
```

```
545 }
```

```
546 }
```

```
547 }
```

```
548 }
```

```
549 }
```

```
550 }
```

```
551 }
```

```
552 }
```

```
553 }
```

```
554 }
```

```
555 }
```

```
556 }
```

```
557 }
```

```
558 }
```

```
559 }
```

```
560 }
```

```
561 }
```

```
562 }
```

```
563 }
```

```
564 }
```

```
565 }
```

```
566 }
```

```
567 }
```

```
568 }
```

```
569 }
```

```
570 }
```

```
571 }
```

```
572 }
```

```
573 }
```

```
574 }
```

```
575 }
```

```
576 }
```

```
577 }
```

```
578 }
```

```
579 }
```

```
580 }
```

```
581 }
```

```
582 }
```

```
583 }
```

```
584 }
```

```
585 }
```

```
586 }
```

```
587 }
```

```
588 }
```

```
589 }
```

```
590 }
```

```
591 }
```

```
592 }
```

```
593 }
```

```
594 }
```

```
595 }
```

```
596 }
```

```
597 }
```

```
598 }
```

```
599 }
```

```
600 }
```

```
601 }
```

```
602 }
```

```
603 }
```

```
604 }
```

```
605 }
```

```
606 }
```

```
607 }
```

```
608 }
```

```
609 }
```

```
610 }
```

```
611 }
```

```
612 }
```

```
613 }
```

```
614 }
```

```
615 }
```

```
616 }
```

```
617 }
```

```
618 }
```

```
619 }
```

```
620 }
```

```
621 }
```

```
622 }
```

```
623 }
```

```
624 }
```

```
625 }
```

```
626 }
```

```
627 }
```

```
628 }
```

```
629 }
```

```
630 }
```

```
631 }
```

```
632 }
```

```
633 }
```

```
634 }
```

```
635 }
```

```
636 }
```

```
637 }
```

```
638 }
```

```
639 }
```

```
640 }
```

```
641 }
```

```
642 }
```

```
643 }
```

```
644 }
```

```
645 }
```

```
646 }
```

```
647 }
```

```
648 }
```

```
649 }
```

```
650 }
```

```
651 }
```

```
652 }
```

```
653 }
```

```
654 }
```

```
655 }
```

```
656 }
```

```
657 }
```

```
658 }
```

```
659 }
```

```
660 }
```

```
661 }
```

```
662 }
```

```
663 }
```

```
664 }
```

```
665 }
```

```
666 }
```

```
667 }
```

```
668 }
```

```
669 }
```

```
670 }
```

```
671 }
```

```
672 }
```

```
673 }
```

```
674 }
```

```
675 }
```

```
676 }
```

```
677 }
```

```
678 }
```

```
679 }
```

```
680 }
```

```
681 }
```

```
682 }
```

```
683 }
```

```
684 }
```

```
685 }
```

```
686 }
```

```
687 }
```

```
688 }
```

```
689 }
```

```
690 }
```

```
691 }
```

```
692 }
```

```
693 }
```

```
694 }
```

```
695 }
```

```
696 }
```

```
697 }
```

```
698 }
```

```
699 }
```

```
700 }
```

```
701 }
```

```
702 }
```

```
703 }
```

```
704 }
```

```
705 }
```

```
706 }
```

```
707 }
```

```
708 }
```

```
709 }
```

```
710 }
```

```
711 }
```

```
712 }
```

```
713 }
```

```
714 }
```

```
715 }
```

```
716 }
```

```
717 }
```

```
718 }
```

```
719 }
```

```
720 }
```

```
721 }
```

```
722 }
```

```
723 }
```

```
724 }
```

```
725 }
```

```
726 }
```

```
727 }
```

```
728 }
```

```
729 }
```

```
730 }
```

```
731 }
```

```
732 }
```

```
733 }
```

```
734 }
```

```
735 }
```

```
736 }
```

```
737 }
```

```
738 }
```

```
739 }
```

```
740 }
```

```
741 }
```

```
742 }
```

```
743 }
```

```
744 }
```

```
745 }
```

```
746 }
```

```
747 }
```

```
748 }
```

```
749 }
```

```
750 }
```

```
751 }
```

```
752 }
```

```
753 }
```

```
754 }
```

```
755 }
```

```
756 }
```

```
757 }
```

```
758 }
```

```
759 }
```

```
760 }
```

```
761 }
```

```
762 }
```

```
763 }
```

```
764 }
```

```
765 }
```

```
766 }
```

```
767 }
```

```
768 }
```

```
769 }
```

```
770 }
```

```
771 }
```

```
772 }
```

```
773 }
```

```
774 }
```

```
775 }
```

```
776 }
```

```
777 }
```

```
778 }
```

```
779 }
```

```
780 }
```

```
781 }
```

```
782 }
```

```
783 }
```

```
784 }
```

```
785 }
```

```
786 }
```

```
787 }
```

```
788 }
```

```
789 }
```

```
790 }
```

```
791 }
```

```
792 }
```

```
793 }
```

```
794 }
```

```
795 }
```

```
796 }
```

```
797 }
```

```
798 }
```

```
799 }
```

```
800 }
```

```
801 }
```

```
802 }
```

```
803 }
```

```
804 }
```

```
805 }
```

```
806 }
```

```
807 }
```

```
808 }
```

```
809 }
```

```
810 }
```

```
811 }
```

```
812 }
```

```
813 }
```

```
814 }
```

```
815 }
```

```
816 }
```

```
817 }
```

```
818 }
```

```
819 }
```

```
820 }
```

```
821 }
```

```
822 }
```

```
823 }
```

```
824 }
```

```
825 }
```

```
826 }
```

```
827 }
```

```
828 }
```

```
829 }
```

```
830 }
```

```
831 }
```

```
832 }
```

```
833 }
```

```
834 }
```

```
835 }
```

```
836 }
```

```
837 }
```

```
838 }
```

```
839 }
```

```
840 }
```

```
841 }
```

```
842 }
```

```
843 }
```

```
844 }
```

```
845 }
```

```
846 }
```

```
847 }
```

```
848 }
```

```
849 }
```

```
850 }
```

```
851 }
```

```
852 }
```

```
853 }
```

```
854 }
```

```
855 }
```

```
856 }
```

```
857 }
```

```
858 }
```

```
859 }
```

```
860 }
```

```
861 }
```

```
862 }
```

```
863 }
```

```
864 }
```

```
865 }
```

```
866 }
```

```
867 }
```

```
868 }
```

```
869 }
```

```
870 }
```

```
871 }
```

```
872 }
```

```
873 }
```

```
874 }
```

```
875 }
```

```
876 }
```

```
877 }
```

```
878 }
```

```
879 }
```

```
880 }
```

```
881 }
```

```
882 }
```

```
883 }
```

```
884 }
```

```
885 }
```

```
886 }
```

```
887 }
```

```
888 }
```

```
889 }
```

```
890 }
```

```
891 }
```

```
892 }
```

```
893 }
```

```
894 }
```

```
895 }
```

```
896 }
```

```
897 }
```

```
898 }
```

```
899 }
```

```
900 }
```

```
901 }
```

```
902 }
```

```
903 }
```

```
904 }
```

```
905 }
```

```
906 }
```

```
907 }
```

```
908 }
```

```
909 }
```

```
910 }
```

```
911 }
```

```
912 }
```

```
913 }
```

```
914 }
```

```
915 }
```

```
916 }
```

```
917 }
```

```
918 }
```

```
919 }
```

```
920 }
```

```
921 }
```

```
922 }
```

```
923 }
```

```
924 }
```

```
925 }
```

```
926 }
```

```
927 }
```

```
928 }
```

```
929 }
```

```
930 }
```

```
931 }
```

```
932 }
```

```
933 }
```

```
934 }
```

```
935 }
```

```
936 }
```

```
937 }
```

```
938 }
```

```
939 }
```

```
940 }
```

```
941 }
```

```
942 }
```

```
943 }
```

```
944 }
```

```
945 }
```

```
946 }
```

```
947 }
```

```
948 }
```

```
949 }
```

```
950 }
```

```
951 }
```

```
952 }
```

```
953 }
```

```
954 }
```

```
955 }
```

```
956 }
```

```
957 }
```

```
958 }
```

```
959 }
```

```
960 }
```

```
961 }
```

```
962 }
```

```
963 }
```

```
964 }
```

```
965 }
```

```
966 }
```

```
967 }
```

```
968 }
```

```
969 }
```

```
970 }
```

```
971 }
```

```
972 }
```

```
973 }
```

```
974 }
```

```
975 }
```

```
976 }
```

```
977 }
```

```
978 }
```

```
979 }
```

```
980 }
```

```
981 }
```

```
982 }
```

```
983 }
```

```
984 }
```

```
985 }
```

```
986 }
```

```
987 }
```

```
988 }
```

```
989 }
```

```
990 }
```

```
991 }
```

```
992 }
```

```
993 }
```

```
994 }
```

```
995 }
```

```
996 }
```

```
997 }
```

```
998 }
```

```
999 }
```

```
1000 }
```

```
1001 }
```

```
1002 }
```

```
1003 }
```

```
1004 }
```

```
1005 }
```

```
1006 }
```

```
1007 }
```

```
1008 }
```

```
1009 }
```

```
1010 }
```

```
1011 }
```

```
1012 }
```

```
1013 }
```

```
1014 }
```

```
1015 }
```

```
1016 }
```

```
1017 }
```

```
1018 }
```

```
1019 }
```

```
1020 }
```

```
1021 }
```

```
1022 }
```

```
1023 }
```

```
1024 }
```

```
1025 }
```

```
1026 }
```

```
1027 }
```

```
1028 }
```

```
1029 }
```

```
1030 }
```

```
1031 }
```

```
1032 }
```

```
1033 }
```

```
1034 }
```

```
1035 }
```

```
1036 }
```

```
1037 }
```

```
1038 }
```

```
1039 }
```

```
1040 }
```

```
1041 }
```

```
1042 }
```

```
1043 }
```

```
1044 }
```

```
1045 }
```

```
1046 }
```

```
1047 }
```

```
1048 }
```

```
1049 }
```

```
1050 }
```

```
1051 }
```

```
1052 }
```

```
1053 }
```

```
1054 }
```

```
1055 }
```

```
1056 }
```

```
1057 }
```

```
1058 }
```

```
1059 }
```

```
1060 }
```

```
1061 }
```

```
1062 }
```

```
1063 }
```

```
1064 }
```

```
1065 }
```

```
1066 }
```

```
1067 }
```

```
1068 }
```

```
1069 }
```

```
1070 }
```

```
1071 }
```

```
1072 }
```

```
1073 }
```

```
1074 }
```

```
1075 }
```

```
1076 }
```

```
1077 }
```

```
1078 }
```

```
1079 }
```

```
1080 }
```

```
1081 }
```

```
1082 }
```

```
1083 }
```

```
1084 }
```

```
1085 }
```

```
1086 }
```

```
1087 }
```

```
1088 }
```

```
1089 }
```

```
1090 }
```

```
1091 }
```

```
1092 }
```

```
1093 }
```

```
1094 }
```

```
1095 }
```

```
1096 }
```

```
1097 }
```

```
1098 }
```

```
1099 }
```

```
1100 }
```

```
1101 }
```

```
1102 }
```

```
1103 }
```

```
1104 }
```

```
1105 }
```

```
1106 }
```

```
1107 }
```

```
1108 }
```

```
1109 }
```

```
1110 }
```

```
1111 }
```

```
1112 }
```

```
1113 }
```

```
1114 }
```

```
1115 }
```

```
1116 }
```

```
1117 }
```

```
1118 }
```

```
1119 }
```

```
1120 }
```

```
1121 }
```

```
1122 }
```

```
1123 }
```

```
1124 }
```

```
1125 }
```

```
1126 }
```

```
1127 }
```

```
1128 }
```

```
1129 }
```

```
1130 }
```

```
1131 }
```

```
1132 }
```

```
1133 }
```

```
1134 }
```

```
1135 }
```

```
1136 }
```

```
1137 }
```

```
1138 }
```

```
1139 }
```

```
1140 }
```

```
1141 }
```

```
1142 }
```

```
1143 }
```

```
1144 }
```

```
1145 }
```

```
1146 }
```

```
1147 }
```

```
1148 }
```

```
1149 }
```

```
1150 }
```

```
1151 }
```

```
1152 }
```

```
1153 }
```

```
1154 }
```

```
1155 }
```

```
1156 }
```

```
1157 }
```

```
1158 }
```

```
1159 }
```

```
1160 }
```

```
1161 }
```

```
1162 }
```

```
1163 }
```

```
1164 }
```

```
1165 }
```

```
1166 }
```

```
1167 }
```

```
1168 }
```

```
1169 }
```

```
1170 }
```

```
1171 }
```

```
1172 }
```

```
1173 }
```

```
1174 }
```

```
1175 }
```

```
1176 }
```

```
1177 }
```

```
1178 }
```

```
1179 }
```

```
1180 }
```

```
1181 }
```

```
1182 }
```

```
1183 }
```

```
1184 }
```

```
1185 }
```

```
1186 }
```

```
1187 }
```

```
1188 }
```

```
1189 }
```

```
1190 }
```

```
1191 }
```

```
1192 }
```

```
1193 }
```

```
1194 }
```

```
1195 }
```

```
1196 }
```

```
1197 }
```

```
1198 }
```

```
1199 }
```

```
1200 }
```

```
1201 }
```

```
1202 }
```

```
1203 }
```

```
1204 }
```

```
1205 }
```

```
1206 }
```

```
1207 }
```

```
1208 }
```

```
1209 }
```

```
1210 }
```

```
1211 }
```

```
1212 }
```

```
1213 }
```

```
1214 }
```

```
1215 }
```

```
1216 }
```

```
1217 }
```

```
1218 }
```

```
1219 }
```

```
1220 }
```

```
1221 }
```

```
1222 }
```

```
1223 }
```

```
1224 }
```

```
1225 }
```

```
1226 }
```

```
1227 }
```

```
1228 }
```

```
1229 }
```

```
1230 }
```

```
1231 }
```

```
1232 }
```

```
1233 }
```

```
1234 }
```

```
1235 }
```

```
1236 }
```

```
1237 }
```

```
1238 }
```

```
1239 }
```

```
1240 }
```

```
1241 }
```

```
1242 }
```

```
1243 }
```

```
1244 }
```

```
1245 }
```

```
1246 }
```

```
1247 }
```

```
1248 }
```

```
1249 }
```

```
1250 }
```

```
1251 }
```

```
1252 }
```

```
1253 }
```

```
1254 }
```

```
1255 }
```

```
1256 }
```

```
1257 }
```

```
1258 }
```

```
1259 }
```

```
1260 }
```

```
1261 }
```

```
1262 }
```

```
1263 }
```

```
1264 }
```

```
1265 }
```

```
1266 }
```

```
1267 }
```

```
1268 }
```

```
1269 }
```

```
1270 }
```

```
1271 }
```

```
1272 }
```

```
1273 }
```

```
1274 }
```

```
1275 }
```

```
1276 }
```

```
1277 }
```

```
1278 }
```

```
1279 }
```

```
1280 }
```

```
1281 }
```

```
1282 }
```

```
1283 }
```

```
1284 }
```

```
1285 }
```

```
1286 }
```

```
1287 }
```

```
1288 }
```

```
1289 }
```

```
1290 }
```

```
1291 }
```

```
1292 }
```

```
1293 }
```

```
1294 }
```

```
1295 }
```

```
1296 }
```

```
1297 }
```

```
1298 }
```

```
1299 }
```

```
1300 }
```

```
1301 }
```

```
1302 }
```

```
1303 }
```

```
1304 }</pre
```

C:\Users\saikr\OneDrive\Desktop\singly linked.cpp - Dev-C++ 5.11

```
File Edit Search View Project Execute Tools ASyle Window Help
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug [C] C programme to find factorial of given numbers using functions with recursion...cpp singly linked.cpp
Node struct
  deleteNode(struct Node** head, int data)
  insertAtBeginning(struct Node** head, int data)
  insertAtEnd(struct Node** head, int data)
  main()
  printList(struct Node*)
  search(struct Node*)
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\singly linked.exe
- Output Size: 131.6875 Kib
- Compilation Time: 2.13s
30°C Humid
ENG IN 21/53 10-05-2023
```

C:\Users\saikr\OneDrive\Desktop\singly linked.cpp - Dev-C++ 5.11

```
File Edit Search View Project Execute Tools ASyle Window Help
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug [C] C programme to find factorial of given numbers using functions with recursion...cpp singly linked.cpp
Node struct
  void deleteNode(struct Node** head, int data)
  if (*head == NULL) {
    printf("List is empty. No node to delete.\n");
    return;
  }
  struct Node* temp = *head;
  struct Node* prev = NULL;
  if (temp != NULL && temp->data == data) {
    head = temp->next;
    free(temp);
    printf("Node with data %d deleted successfully.\n", data);
    return;
  }
  while (temp != NULL && temp->data != data) {
    prev = temp;
    temp = temp->next;
  }
  if (temp == NULL) {
    printf("Node with data %d not found in the list.\n", data);
    return;
  }
  prev->next = temp->next;
  free(temp);
  printf("Node with data %d deleted successfully.\n", data);
}
struct Node* search(struct Node* head, int data) {
  struct Node* temp = head;
  while (temp != NULL) {
    if (temp->data == data) {
      return temp;
    }
  }
}
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\singly linked.exe
- Output Size: 131.6875 Kib
- Compilation Time: 2.13s
30°C Humid
ENG IN 21/53 10-05-2023
```

The screenshot shows the Code::Blocks IDE interface with the following details:

- File Path:** C:\Users\saikr\Desktop\singly linked.cpp - Dev-C++ 5.11
- Project:** singly linked.cpp
- Code Editor:** Displays C code for singly linked list operations. The code includes functions for inserting at beginning, end, and position; deleting from beginning, end, and position; printing the list; and searching for a node.
- Compiler Output:** Shows compilation results with 0 errors and 0 warnings. The output file is C:\Users\saikr\Desktop\singly linked.exe, with an output size of 131.6875 KB and a compilation time of 2.13s.
- System Status:** Shows the system temperature as 30°C and humidity as Humid. The taskbar includes icons for File Explorer, Search, Task View, Edge, File History, Photos, OneDrive, Mail, News, Instagram, WhatsApp, and Microsoft Defender.

26) Binary search tree by using C programming.

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

- File Menu:** File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, Help.
- Compiler:** ION-GCC 4.9.2 64-bit Release.
- Project Explorer:** Shows the file structure for "binary search tree.cpp".
- Code Editor:** The main code for a binary search tree implementation. It includes functions for creating nodes, inserting data, searching for data, and performing in-order traversal. The code uses dynamic memory allocation for nodes.
- Output Window:** Displays the execution of the program. It shows the menu "Binary Search Tree Operations" and the user's choice of insertion (choice 1). It then prompts for an element to insert (12) and confirms the insertion was successful. The menu is shown again with choice 5 selected, followed by a message "Exiting the program...".
- Compiler Output:** Shows the command "Process exited after 10.18 seconds with return value 0" and "Press any key to continue . . .".
- Status Bar:** Shows the system temperature (35°C), battery status, network connection, and the date/time (10-05-2023 12:21).

C:\Users\saikr\OneDrive\Desktop\binary search tree.cpp - [Executing] - Dev-C++ 5.11

```
File Edit Search View Project Execute Tools ASyle Window Help
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug binary search tree.cpp
node struct
└ createNode(int data)
  deleteNode(struct node*)
  inorderTraversal(struct node*)
  insert(struct node* rc)
  main() int
  minValueNode(struct node*)
  search(struct node*)

37     }
38     if (data < root->data) {
39         return search(root->left, data);
40     }
41     return search(root->right, data);
42 }
43
44 struct node* minValueNode(struct node* node) {
45     struct node* current = node;
46
47     while (current && current->left != NULL) {
48         current = current->left;
49     }
50
51     return current;
52 }
53
54 struct node* deleteNode(struct node* root, int data) {
55     if (root == NULL) {
56         return root;
57     }
58
59     if (data < root->data) {
60         root->left = deleteNode(root->left, data);
61     } else if (data > root->data) {
62         root->right = deleteNode(root->right, data);
63     } else {
64         if (root->left == NULL) {
65             struct node* temp = root->right;
66             free(root);
67             return temp;
68         } else if (root->right == NULL) {
69             struct node* temp = root->left;
70             free(root);
71             return temp;
72         }
73     }
74 }
75
76 struct node* temp = minValueNode(root->right);
77 root->data = temp->data;
78 root->right = deleteNode(root->right, temp->data);
79
80 return root;
81
82 void inorderTraversal(struct node* root) {
83     if (root != NULL) {
84         inorderTraversal(root->left);
85         printf("%d ", root->data);
86         inorderTraversal(root->right);
87     }
88 }
89
90 int main() {
91     struct node* root = NULL;
92     int choice, data;
93
94     do {
95         printf("\nBinary Search Tree Operations\n");
96         printf("1. Insertion\n");
97         printf("2. Deletion\n");
98         printf("3. Searching\n");
99         printf("4. Inorder Traversal\n");
100        printf("5. Exit\n");
101        printf("Enter your choice: ");
102        scanf("%d", &choice);
103
104        switch (choice) {
105            case 1:
```

Compiler (2) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw\bin\gcc.exe cannot open output file C:\Users\saikr\OneDrive\Desktop\binary search tree.exe: Permission denied

C:\Users\saikr\OneDrive\Desktop\collect2.exe [Error] Id returned 1 exit status

C:\Users\saikr\OneDrive\Desktop\binary search tree.cpp - [Executing] - Dev-C++ 5.11

```
File Edit Search View Project Execute Tools ASyle Window Help
TDM-GCC 4.9.2 64-bit Release
Project Classes Debug binary search tree.cpp
node struct
└ createNode(int data)
  deleteNode(struct node*)
  inorderTraversal(struct node*)
  insert(struct node* rc)
  main() int
  minValueNode(struct node*)
  search(struct node*)

71     free(root);
72     return temp;
73 }
74
75 struct node* temp = minValueNode(root->right);
76 root->data = temp->data;
77 root->right = deleteNode(root->right, temp->data);
78
79 return root;
80
81 void inorderTraversal(struct node* root) {
82     if (root != NULL) {
83         inorderTraversal(root->left);
84         printf("%d ", root->data);
85         inorderTraversal(root->right);
86     }
87 }
88
89 int main() {
90     struct node* root = NULL;
91     int choice, data;
92
93     do {
94         printf("\nBinary Search Tree Operations\n");
95         printf("1. Insertion\n");
96         printf("2. Deletion\n");
97         printf("3. Searching\n");
98         printf("4. Inorder Traversal\n");
99         printf("5. Exit\n");
100        printf("Enter your choice: ");
101        scanf("%d", &choice);
102
103        switch (choice) {
104            case 1:
```

Compiler (2) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw\bin\gcc.exe cannot open output file C:\Users\saikr\OneDrive\Desktop\binary search tree.exe: Permission denied

C:\Users\saikr\OneDrive\Desktop\collect2.exe [Error] Id returned 1 exit status



C:\Users\saikr\OneDrive\Desktop\binary search tree.cpp - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

IDE-GCC 4.9.2 64-bit Release

Project Classes Debug binary search tree.cpp

node : struct

```
187     {
188         int data;
189         struct node* left;
190         struct node* right;
191     };
192
193     void createNode(int data);
194     void deleteNode(struct node* &root, int data);
195     void inorderTraversal(struct node* root);
196     void insert(struct node*& root, int data);
197     void main();
198     void minValueNode(struct node* root);
199     void search(struct node*& root, int data);
```

switch (choice) {

case 1:

```
199         printf("Enter the element to insert: ");
200         scanf("%d", &data);
201         insert(root, data);
202         printf("Element inserted successfully.\n");
```

break;

case 2:

```
203         printf("Enter the element to delete: ");
204         scanf("%d", &data);
205         root = deleteNode(root, data);
206         printf("Element deleted successfully.\n");
```

break;

case 3:

```
207         printf("Enter the element to search: ");
208         scanf("%d", &data);
209         if (search(root, data) != NULL) {
210             printf("Element found in the BST.\n");
211         } else {
212             printf("Element not found in the BST.\n");
213         }
214     }
```

break;

case 4:

```
215         printf("Inorder Traversal: ");
216         inorderTraversal(root);
217         printf("\n");
```

break;

case 5:

```
218         printf("Exiting the program...\n");
```

break;

default:

```
220         printf("Invalid choice! Please enter a valid option.\n");
```

}

while (choice != 5);

return 0;

Compiler (2) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw32\bin\gcc.exe: cannot open output file C:\Users\saikr\OneDrive\Desktop\binary search tree.exe: Permission denied
[Error] Id returned 1 exit status

35°C Haze

Search

12:22 10-05-2023

27)Binary-Tree Traversal using C programming.

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

- File Menu:** File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, Help.
- Compiler:** TDM-GCC 4.9.2 64-bit Release.
- Project Tree:** Untitled1.cpp, Node struct, createNode(int data), inorderTraversal(str), insertNode(struct Node*), main() : int, postorderTraversal(), preorderTraversal().
- Code Editor:** C++ code for a binary tree. It includes functions for creating nodes, inserting data, and performing in-order, pre-order, and post-order traversals. The code uses standard C++ headers like stdio.h and stdlib.h.
- Output Window:** Shows the execution of the program. It asks for the number of elements (6), then lists the inserted elements (12, 5, 19, 7, 15, 21) and their traversal paths (in-order: 5 7 12 15 19 21; pre-order: 12 5 7 19 15 21; post-order: 7 15 21 19 12). Finally, it exits after 22.71 seconds.
- Compiler Tab:** Shows compilation results with 0 errors and 0 warnings, and a compilation time of 0.34s.

C:\Users\saikr\OneDrive\Desktop\Untitled1.cpp - [Executing] - Dev-C++ 5.11

```
Project Classes Debug Untitled1.cpp
Node struct
createNode(int data)
inorderTraversal()
insertNode()
main()
postorderTraversal()
preorderTraversal()

void inorderTraversal(struct Node* root) {
    if (root != NULL) {
        inorderTraversal(root->left);
        printf("%d ", root->data);
        inorderTraversal(root->right);
    }
}

void preorderTraversal(struct Node* root) {
    if (root != NULL) {
        printf("%d ", root->data);
        preorderTraversal(root->left);
        preorderTraversal(root->right);
    }
}

void postorderTraversal(struct Node* root) {
    if (root != NULL) {
        postorderTraversal(root->left);
        postorderTraversal(root->right);
        printf("%d ", root->data);
    }
}

int main() {
    struct Node* root = NULL;
    int numElements, element;
    printf("Enter the number of elements to insert in the binary tree: ");
    scanf("%d", &numElements);
    for (int i = 0; i < numElements; i++) {
        printf("Enter the elements:\n");
        scanf("%d", &element);
        insertNode(element);
    }
}
```

Compiler Resources Compile Log Debug Find Results Close

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\Untitled1.exe
- Output Size: 130.6494140625 KiB
- Compilation Time: 0.34s

C:\Users\saikr\OneDrive\Desktop\Untitled1.exe

Enter the number of elements to insert in the binary tree: 6
Enter the elements:
12 5 19 7 15 21
Inorder traversal: 5 7 12 15 19 21
Preorder traversal: 12 5 7 19 15 21
Postorder traversal: 7 15 21 19 12

Process exited after 22.71 seconds with return value 0
Press any key to continue . . .

10:45 11-05-2023

C:\Users\saikr\OneDrive\Desktop\Untitled1.cpp - [Executing] - Dev-C++ 5.11

```
Project Classes Debug Untitled1.cpp
Node struct
createNode(int data)
inorderTraversal()
insertNode()
main()
postorderTraversal()
preorderTraversal()

void postorderTraversal(struct Node* root) {
    if (root != NULL) {
        postorderTraversal(root->left);
        postorderTraversal(root->right);
        printf("%d ", root->data);
    }
}

int main() {
    struct Node* root = NULL;
    int numElements, element;
    printf("Enter the number of elements to insert in the binary tree: ");
    scanf("%d", &numElements);
    for (int i = 0; i < numElements; i++) {
        printf("Enter the elements:\n");
        scanf("%d", &element);
        insertNode(element);
    }
}

printf("Inorder traversal: ");
inorderTraversal(root);
printf("\n");

printf("Preorder traversal: ");
preorderTraversal(root);
printf("\n");

printf("Postorder traversal: ");
postorderTraversal(root);
printf("\n");

return 0;
}
```

Compiler Resources Compile Log Debug Find Results Close

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\saikr\OneDrive\Desktop\Untitled1.exe
- Output Size: 130.6494140625 KiB
- Compilation Time: 0.34s

C:\Users\saikr\OneDrive\Desktop\Untitled1.exe

Enter the number of elements to insert in the binary tree: 6
Enter the elements:
12 5 19 7 15 21
Inorder traversal: 5 7 12 15 19 21
Preorder traversal: 12 5 7 19 15 21
Postorder traversal: 7 15 21 19 12

Process exited after 22.71 seconds with return value 0
Press any key to continue . . .

10:46 11-05-2023

28) Write C programme to find factorial of given numbers using functions without recursion.

The screenshot shows the Dev-C++ IDE interface. The project window displays a file named "C programme to find factorial of given numbers using functions without recursion.cpp". The code defines a recursive function "factorial" and a main function that reads a number from the user and prints its factorial. The output window shows the execution of the program for n=12, resulting in 479001600. The status bar at the bottom indicates the date and time as 10-05-2023 and 12:41.

```
Project: Classes Debug [*] binary search tree.cpp [*] SUM OF DIGITS USING WHILE LOOP.cpp Binary-Tree Traversal using C programming.cpp C programme to find factorial of given numbers using functions without recursion.cpp
File Edit Search View Project Execute Tools AsStyle Window Help
File Project (globals)
factorial(int num) : int
main() : int
1 //Factorial of a given number using function without recursion
2
3 int factorial(int num)
4 {
5     int fact=1;
6     for(int i=1;i<=num;i++)
7     {
8         fact=fact*i;
9     }
10    return fact;
11 }
12
13 int main()
14 {
15     int number;
16     printf("Enter a number: ");
17     scanf("%d", &number);
18
19     int fact = factorial(number);
20     printf("Factorial of %d is %d\n", number, fact);
21
22     return 0;
23 }
```

```
C:\Users\saikr\OneDrive\Desktop\C programme to find factorial of given numbers using functions without recursion.exe
Enter a number: 12
Factorial of 12 is 479001600
Process exited after 3.113 seconds with return value 0
Press any key to continue . . .

Compiler Resources Compile Log Debug Find Results Close
Compilation results...
Abort Compilation
Shorten compiler paths
35°C Haze
```

29) Write C programme to find factorial of given numbers using functions with recursion.

The screenshot shows the Dev-C++ IDE interface. The project window displays a file named "C programme to find factorial of given numbers using functions with recursion.cpp". The code defines a recursive function "factorial" and a main function that reads a number from the user and prints its factorial. It includes a check for negative numbers. The output window shows the execution of the program for n=12, resulting in 479001600. The status bar at the bottom indicates the date and time as 10-05-2023 and 12:43.

```
Project: Classes Debug [*] binary search tree.cpp [*] SUM OF DIGITS USING WHILE LOOP.cpp Binary-Tree Traversal using C programming.cpp C programme to find factorial of given numbers using functions with recursion.cpp
File Edit Search View Project Execute Tools AsStyle Window Help
File Project (globals)
factorial(int n) : int
main() : int
1 //Factorial of a given number using function with recursion
2
3 #include <stdio.h>
4
5 int factorial(int n)
6 {
7     if (n == 0 || n == 1)
8         return 1;
9     else
10        return n * factorial(n - 1);
11 }
12
13 int main()
14 {
15     int number;
16     printf("Enter a positive integer: ");
17     scanf("%d", &number);
18
19     if (number < 0)
20         printf("Error: Factorial of a negative number is undefined.");
21     else
22         int result = factorial(number);
23         printf("Factorial of %d is %d\n", number, result);
24
25     return 0;
26 }
```

```
C:\Users\saikr\OneDrive\Desktop\C programme to find factorial of given numbers using functions with recursion.exe
Enter a positive integer: 12
Factorial of 12 is 479001600
Process exited after 3.606 seconds with return value 0
Press any key to continue . . .

Compiler Resources Compile Log Debug Find Results Close
Compilation results...
Abort Compilation
Shorten compiler paths
35°C Haze
```

30) Write a C programme to generate Fibonacci using function with recursion

The screenshot shows the Dev-C++ IDE interface. The code editor displays a file named Untitled1.cpp containing the following C code:

```
//Fibonacci with recursion
#include <stdio.h>
int fibonaci(int n);
int main()
{
    int n, i;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci Series: ");
    for (i = 0; i < n; i++)
    {
        printf("%d ",fibonaci(i));
    }
    return 0;
}
int fibonaci(int n)
{
    if (n == 0)
        return 0;
    else if (n == 1)
        return 1;
    else
        return fibonaci(n - 1) + fibonaci(n - 2);
}
```

The output window shows the execution of the program. It prompts the user for the number of terms (6), prints the Fibonacci series (0 1 1 2 3 5), and then exits.

The status bar at the bottom indicates the weather as 34°C Mostly cloudy, the date and time as 11-05-2023 11:42, and the system language as ENG IN.