

Online C Compiler - Programiz x ChatGPT x +

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

Paused

All Bookmarks

Programiz

C Online Compiler

Programiz PRO

Premium Coding Courses by Programiz

Learn More

Programiz PRO

main.c

Share Run

Clear

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 typedef struct Node {
4     int data;
5     struct Node* left;
6     struct Node* right;
7 } Node;
8 Node* createNode(int data) {
9     Node* newNode = (Node*)malloc(sizeof(Node));
10    newNode->data = data;
11    newNode->left = NULL;
12    newNode->right = NULL;
13    return newNode;
14 }
15 void inorderTraversal(Node* root) {
16     if (root == NULL) return;
17     inorderTraversal(root->left);
18     printf("%d ", root->data);
19     inorderTraversal(root->right);
20 }
21 void preorderTraversal(Node* root) {
22     if (root == NULL) return;
23     printf("%d ", root->data);
24     preorderTraversal(root->left);
25     preorderTraversal(root->right);
26 }
27 void postorderTraversal(Node* root) {
28     if (root == NULL) return;
29     postorderTraversal(root->left);
30     postorderTraversal(root->right);
31     printf("%d ", root->data);
32 }
33 int main() {
34     Node* root = createNode(1);
35     root->left = createNode(2);
36     root->right = createNode(3);
37     root->left->left = createNode(4);
38     root->left->right = createNode(5);
39     printf("In-order traversal: ");
40     inorderTraversal(root);
41     printf("\n");
42     printf("Pre-order traversal: ");
43     preorderTraversal(root);
44     printf("\n");
45     printf("Post-order traversal: ");
46     postorderTraversal(root);
47     printf("\n");
48     return 0;
49 }
```

Output

Clear

./src/postord.c
In-order traversal: 4 2 5 1 3
Pre-order traversal: 1 2 4 5 3
Post-order traversal: 4 5 2 3 1

=== Code Execution Successful ===

Online C Compiler - Programizcircular linked list in c

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


Paused

All Bookmarks

Programiz

C Online Compiler

Premium Coding Courses by Programiz






Programiz

PRO

Programiz PRO

main.c



Run

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 typedef struct Node {
4     char data;
5     struct Node* left;
6     struct Node* right;
7 } Node;
8 Node* createNode(char data) {
9     Node* newNode = (Node*)malloc(sizeof(Node));
10    newNode->data = data;
11    newNode->left = newNode->right = NULL;
12    return newNode;
13 }
14 void inorder(Node* root) {
15     if (root != NULL) {
16         inorder(root->left);
17         printf("%c ", root->data);
18         inorder(root->right);
19     }
20 }
21 int main() {
22     Node* root = createNode('-');
23     root->left = createNode('A');
24     root->right = createNode('*');
25     root->right->left = createNode('B');
26     root->right->right = createNode('C');
27     printf("Inorder traversal: ");
28     inorder(root);
29     return 0;
30 }
```

Output

Clear

/tmp/cMlpLmBQnz.o

Inorder traversal: A - B * C

=== Code Execution Successful ===

Online C Compiler - Programiz

circular linked list in c

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

Programiz
C Online Compiler

Premium Coding
Courses by Programiz

Programiz PRO

Programiz PRO

main.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 typedef struct Node {
4     int data;
5     struct Node* left;
6     struct Node* right;
7 } Node;
8 Node* createNode(int data) {
9     Node* newNode = (Node*)malloc(sizeof(Node));
10    newNode->data = data;
11    newNode->left = newNode->right = NULL;
12    return newNode;
13 }
14 Node* insert(Node* root, int data) {
15     if (root == NULL) {
16         return createNode(data);
17     }
18     if (data < root->data) {
19         root->left = insert(root->left, data);
20     } else if (data > root->data) {
21         root->right = insert(root->right, data);
22     }
23     return root;
24 }
25 Node* search(Node* root, int data) {
26     if (root == NULL || root->data == data) {
27         return root;
28     }
29     if (data < root->data) {
30         return search(root->left, data);
31     } else {
32         return search(root->right, data);
33     }
34 }
35 void inorder(Node* root) {
36     if (root != NULL) {
37         inorder(root->left);
38         printf("%d ", root->data);
39         inorder(root->right);
40     }
41 }
42 int main() {
43     Node* root = NULL;
44     root = insert(root, 50);
45     root = insert(root, 30);
46     root = insert(root, 20);
47     root = insert(root, 40);
48     root = insert(root, 70);
49     root = insert(root, 60);
50     root = insert(root, 80);
51     printf("Inorder traversal: ");
52     inorder(root);
```

Output

Clear

```
100 / 201618007.c
Inorder traversal: A + B * C
=== Code Execution Successful ===
```