Programiz
C Online Compiler

main.c

Share Run

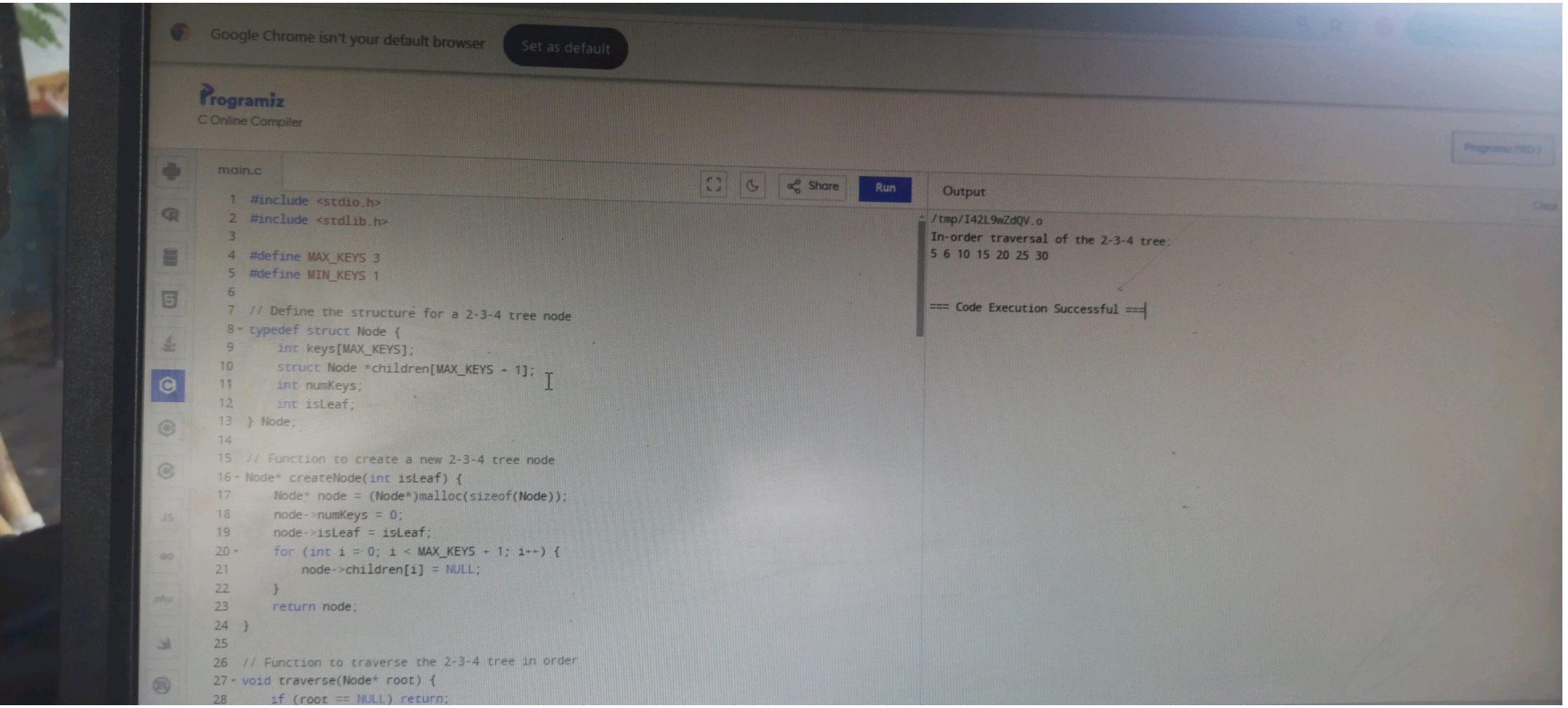
```c
// Searching a key on a B-tree in C

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

#define MAX 3
#define MIN 2

struct BTreeNode {
  int val[MAX + 1], count;
  struct BTreeNode *link[MAX + 1];
};

struct BTreeNode *root;

// Create a node
struct BTreeNode *createNode(int val, struct BTreeNode *child) {
  struct BTreeNode *newNode;
  newNode = (struct BTreeNode *)malloc(sizeof(struct BTreeNode));
  newNode->val[1] = val;
  newNode->count = 1;
  newNode->link[0] = root;
  newNode->link[1] = child;
  return newNode;
}

// Insert node
void insertNode(int val, int pos, struct BTreeNode *node,
```

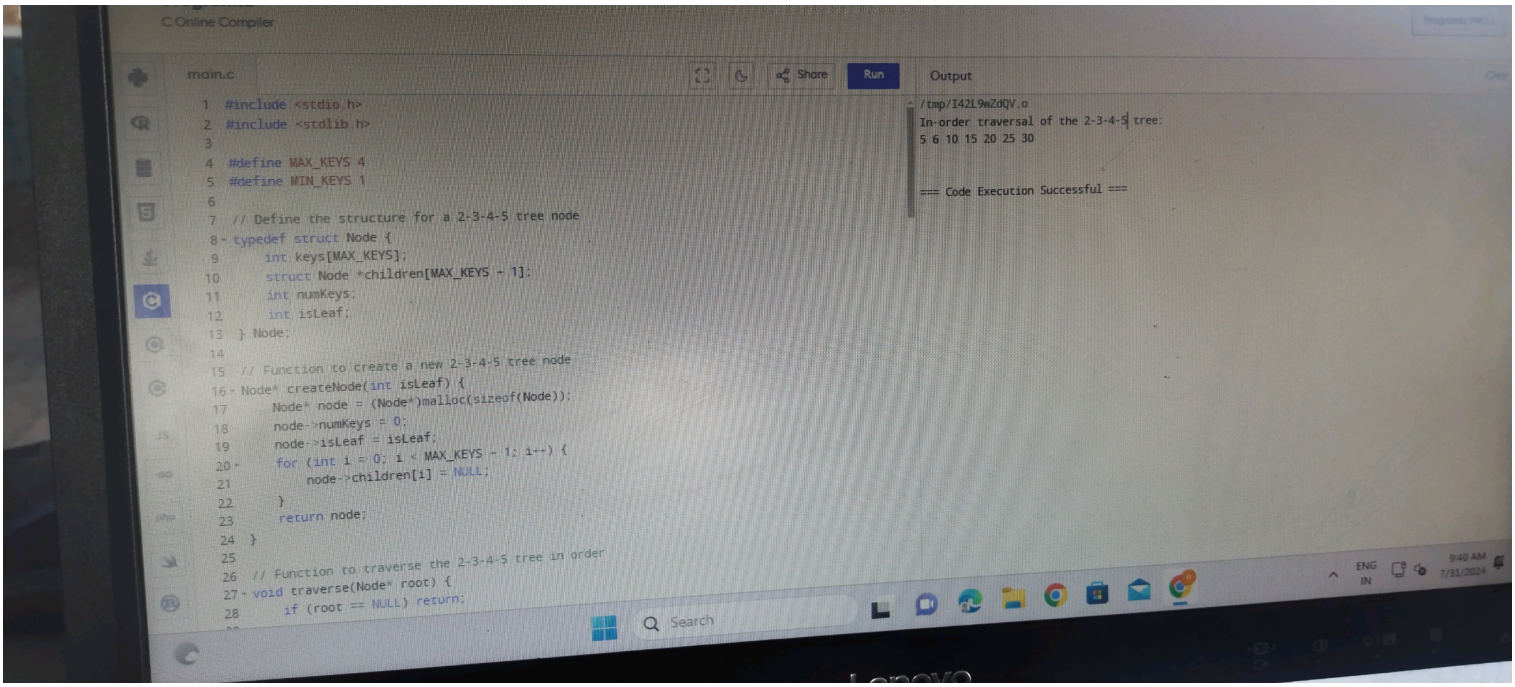
Output

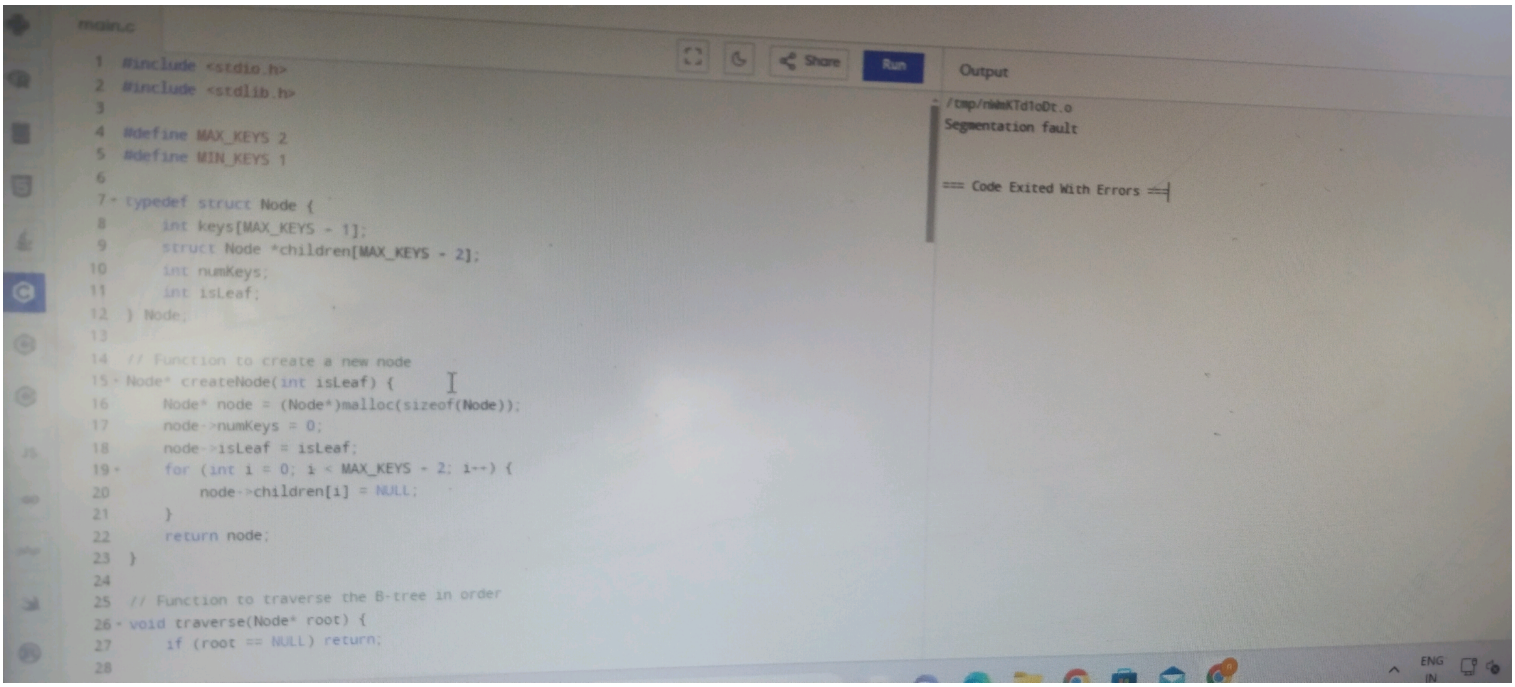
```
/tmp/oSfQ2dOVix.o
8 9 10 11 15 16 17 18 20 23
11 is found

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

Search

ENG


Google Chrome isn't your default browser
Set as default
Programiz
C Online Compiler
main.c
Share
Run
Output
/tmp/I42L9wZdQV.o
In-order traversal of the 2-3-4 tree:
5 6 10 15 20 25 30
=== Code Execution Successful ===
#include <stdio.h>
#include <stdlib.h>
#define MAX_KEYS 3
#define MIN_KEYS 1
// Define the structure for a 2-3-4 tree node
typedef struct Node {
int keys[MAX_KEYS];
struct Node *children[MAX_KEYS + 1];
int numKeys;
int isLeaf;
} Node;
// Function to create a new 2-3-4 tree node
Node* createNode(int isLeaf) {
Node* node = (Node*)malloc(sizeof(Node));
node->numKeys = 0;
node->isLeaf = isLeaf;
for (int i = 0; i < MAX_KEYS + 1; i++) {
node->children[i] = NULL;
}
return node;
}
// Function to traverse the 2-3-4 tree in order
void traverse(Node* root) {
if (root == NULL) return;


C Online Compiler
main.c
Share
Run
Output
#include <stdio.h>
#include <stdlib.h>
#define MAX_KEYS 4
#define MIN_KEYS 1
// Define the structure for a 2-3-4-5 tree node
typedef struct Node {
int keys[MAX_KEYS];
struct Node *children[MAX_KEYS + 1];
int numKeys;
int isLeaf;
} Node;
// Function to create a new 2-3-4-5 tree node
Node* createNode(int isLeaf) {
Node* node = (Node*)malloc(sizeof(Node));
node->numKeys = 0;
node->isLeaf = isLeaf;
for (int i = 0; i < MAX_KEYS + 1; i++) {
node->children[i] = NULL;
}
return node;
}
// Function to traverse the 2-3-4-5 tree in order
void traverse(Node* root) {
if (root == NULL) return;
/tmp/I42L9wZdQV.o
In-order traversal of the 2-3-4-5 tree:
5 6 10 15 20 25 30
=== Code Execution Successful ===
Search
ENG
IN
9:40 AM
7/31/2024


main.c
Share
Run
Output
/tmp/nkmKTd1oDt.o
Segmentation fault
=== Code Exited With Errors ===
#include <stdio.h>
#include <stdlib.h>
#define MAX_KEYS 2
#define MIN_KEYS 1
typedef struct Node {
int keys[MAX_KEYS - 1];
struct Node *children[MAX_KEYS - 2];
int numKeys;
int isLeaf;
} Node;
// Function to create a new node
Node* createNode(int isLeaf) {
Node* node = (Node*)malloc(sizeof(Node));
node->numKeys = 0;
node->isLeaf = isLeaf;
for (int i = 0; i < MAX_KEYS - 2; i++) {
node->children[i] = NULL;
}
return node;
}
// Function to traverse the B-tree in order
void traverse(Node* root) {
if (root == NULL) return;
ENG
IN