Контест

Демоконтест Настройки компиляторов Значения ошибок Команды

Тренировочный контест — бэкенд

Задачи Посылки Сообщения

C. Приснится же такое...

✓ Полное решение </> OK Go 1.23.0

Исходный код

```
package main

import (
    "bufio"
    "container/list"
    "fmt"
    "os"
    "strconv"
)

type Node struct {
    value  int
    left   *Node
    right  *Node
    parent *Node
}

func NewNode(value int, parent *Node) *Node {
    return &Node{
        value:  value,
        parent: parent,
    }
}

func swapWithLeftChild(p, v *Node, nodeMap map[int]*Node) {
    if v.right != nil {
        v.right.parent = p
    }
    if p.right != nil {
        p.right.parent = v
    }

    p.value, v.value = v.value, p.value
    p.right, v.right = v.right, p.right
    nodeMap[v.value] = v
    nodeMap[p.value] = p
}

func swapWithRightChild(p, v *Node, nodeMap map[int]*Node) {
    if v.left != nil {
        v.left.parent = p
    }
    if p.left != nil {
        p.left.parent = v
    }

    p.value, v.value = v.value, p.value
    p.left, v.left = v.left, p.left
    nodeMap[v.value] = v
    nodeMap[p.value] = p
}
```

```go

func swap(p, v *Node, nodeMap map[int]*Node) {
    if p.left == v {
        swapWithLeftChild(p, v, nodeMap)
    } else {
        swapWithRightChild(p, v, nodeMap)
    }
}

func buildTree(n int) (root *Node, nodeMap map[int]*Node) {
    root = NewNode(1, nil)
    queue := list.New()
    queue.PushBack(root)
    currentNodeId := 2
    nodeMap = make(map[int]*Node, 0)
    nodeMap[1] = root

    for queue.Len() > 0 && currentNodeId <= n {
        parent := queue.Front().Value.(*Node)
        queue.Remove(queue.Front())

        if currentNodeId <= n {
            parent.left = NewNode(currentNodeId, parent)
            queue.PushBack(parent.left)
            nodeMap[currentNodeId] = parent.left
            currentNodeId++
        }
        if currentNodeId <= n {
            parent.right = NewNode(currentNodeId, parent)
            queue.PushBack(parent.right)
            nodeMap[currentNodeId] = parent.right
            currentNodeId++
        }
    }

    return root, nodeMap
}

func lvr(root *Node) {
    if root == nil {
        return
    }

    stack := make([]*Node, 0)
    curr := root

    for curr != nil || len(stack) > 0 {
        for curr != nil {
            stack = append(stack, curr)
            curr = curr.left
        }

        curr = stack[len(stack)-1]
        stack = stack[:len(stack)-1]
        fmt.Printf("%d ", curr.value)
        curr = curr.right
    }
}

func main() {
    file, err := os.Open("input.txt")
    if err != nil {
        panic(err)
    }
    defer file.Close()

    scan := bufio.NewScanner(file)
    scan.Split(bufio.ScanWords)
    scan.Scan()
    N, err := strconv.Atoi(scan.Text())
    if err != nil {
        panic(err)
    }

    // Пропускаем число изменений
    scan.Scan()

    root, nodeMap := buildTree(N)

    for scan.Scan() {
        nodeId, err := strconv.Atoi(scan.Text())
        if err != nil {
            panic(err)
        }

        node := nodeMap[nodeId]
        if node == root {
            continue
        }

        swap(node.parent, node, nodeMap)
    }

    lvr(root)
}

```

Отличия от предыдущей посылки

Лог компиляции

№	Вердикт	Ресурсы	Баллы
1	ok	3ms / 2.12Mb	—
2	ok	3ms / 2.00Mb	—
3	ok	3ms / 2.00Mb	—
4	ok	3ms / 1.88Mb	—
5	ok	3ms / 1.88Mb	—
6	ok	3ms / 2.13Mb	—
7	ok	3ms / 1.88Mb	—
8	ok	3ms / 1.88Mb	—
9	ok	3ms / 2.00Mb	—
10	ok	3ms / 2.00Mb	—
11	ok	3ms / 2.12Mb	—
12	ok	3ms / 2.12Mb	—
13	ok	132ms / 532.00Kb	—
14	ok	106ms / 532.00Kb	—
15	ok	102ms / 400.00Kb	—
16	ok	100ms / 400.00Kb	—
17	ok	107ms / 400.00Kb	—
18	ok	101ms / 400.00Kb	—
19	ok	107ms / 400.00Kb	—

№	Вердикт	Ресурсы	Баллы
20	ok	101ms / 400.00Kb	—
21	ok	98ms / 400.00Kb	—
22	ok	86ms / 400.00Kb	—
23	ok	92ms / 400.00Kb	—
24	ok	86ms / 400.00Kb	—
25	ok	87ms / 400.00Kb	—
26	ok	75ms / 400.00Kb	—
27	ok	75ms / 400.00Kb	—
28	ok	3ms / 2.13Mb	—
29	ok	81ms / 532.00Kb	—
30	ok	91ms / 532.00Kb	—
31	ok	90ms / 532.00Kb	—
32	ok	94ms / 532.00Kb	—
33	ok	93ms / 532.00Kb	—
34	ok	102ms / 532.00Kb	—
35	ok	101ms / 532.00Kb	—
36	ok	97ms / 532.00Kb	—
37	ok	98ms / 532.00Kb	—
38	ok	195ms / 532.00Kb	—
39	ok	170ms / 532.00Kb	—
40	ok	164ms / 532.00Kb	—
41	ok	159ms / 532.00Kb	—
42	ok	159ms / 532.00Kb	—
43	ok	141ms / 532.00Kb	—
44	ok	134ms / 532.00Kb	—
45	ok	128ms / 532.00Kb	—
46	ok	128ms / 532.00Kb	—
47	ok	122ms / 532.00Kb	—