**Федеральное государственное автономное образовательное учреждение**

**высшего образования**   
**"Национальный исследовательский университет**   
**"Высшая школа экономики"**

Московский институт электроники и математики им. А.Н. Тихонова НИУ ВШЭ

Департамент компьютерной инженерии

**Курс: Алгоритмизация вычислений**

|  |  |  |
| --- | --- | --- |
| **Раздел** | **Маx оценка** | **Оценка** |
| **Работа программы** | **1** |  |
| **Тесты** | **1** |  |
| **Правильность алгоритма** | **3** |  |
| **Ответы на вопросы** | **2** |  |
| **Дополнительное задание** | **3** |  |
| **Итого** |  |  |

**ОТЧЕТ**

**по лабораторной работе №6**

**Студент: Калиманов Иван Михайлович**

**Группа: БИВ194**

**Вариант:\_№220 (1, 13)**

**Руководитель: Волкова Лилия Леонидовна**

**Оценка:\_\_\_\_\_\_\_\_\_\_**

**МОСКВА 2020**

Содержание

[**Задание** 3](#_Toc38663665)

[**Листинг программы** 4](#_Toc38663666)

[Задание 1 4](#_Toc38663667)

[Залание 2 5](#_Toc38663668)

[**Распечатка тестов к программе и результатов** 11](#_Toc38663669)

# **Задание**

I. Написать программу, включающую рекурсивную функцию для вычисления:

1. НОД двух неотрицательных целых чисел n и m с использованием

второго алгоритма Евклида:

II. Написать программу, которая создает бинарное дерево, состоящее из целых

чисел, которые вводятся с клавиатуры. Затем программа выполняет

преобразования в соответствии с номером варианта и печатает все числа

исходного и преобразованного бинарного дерева (или другой результат работы

программы).

13. Находит сумму всех нечетных элементов дерева.

# **Листинг программы**

# Задание 1

Файл gcd.h

#include <iostream>

#include <chrono>

unsigned int gcd\_recursive(unsigned int n, unsigned int m);

unsigned int gcd\_iteration(unsigned int n, unsigned int m);

unsigned int gcd\_euclide(unsigned int n, unsigned int m);

Файл gcd.cpp

#include "gcd.h"

unsigned int gcd\_recursive(unsigned int n, unsigned int m)

{

if (n > m){

unsigned int temp = n;

n = m;

m = temp;

}

return gcd\_euclide(n, m);

}

unsigned int gcd\_euclide(unsigned int n, unsigned int m)

{

if (m == 0)

return n;

return gcd\_euclide(m, n % m);

}

unsigned int gcd\_iteration(unsigned int n, unsigned int m)

{

while (n != m) {

if (n > m) {

unsigned int tmp = n;

n = m;

m = tmp;

}

m = m - n;

}

return n;

}

# **Задание 2**

**#include <stdio.h>**

**#include <malloc.h>**

**using namespace std;**

**#define LINE\_LEN 255**

**#define WORD\_LEN 81**

**#define EDGE\_LEN 5**

**typedef struct tree\_node{**

**int value; /\*значение в узле\*/**

**tree\_node \*left; /\*левый потомок\*/**

**tree\_node \*right; /\*правый потомок\*/**

**} tree\_node;**

**tree\_node \*init\_root(int);**

**void set\_left(tree\_node \*, int);**

**void set\_right(tree\_node \*, int);**

**int sum\_of\_odd\_in\_tree(tree\_node \*);**

**int is\_odd\_in\_tree(tree\_node \*);**

**tree\_node \*get\_tree();**

**void print\_tree(tree\_node \*, int);**

**int main()**

**{**

**tree\_node \*tree = get\_tree();**

**if (!tree)**

**puts("Tree is empty");**

**else**

**print\_tree(tree, 0);**

**if (is\_odd\_in\_tree(tree)){**

**int sum = sum\_of\_odd\_in\_tree(tree);**

**printf("%s %d\n", "Sum of odd numbers in tree is", sum);**

**}**

**else**

**puts("There are no odd numbers");**

**}**

**/\***

**\*инизиализация узла дерева**

**\*/**

**tree\_node \*init\_root(int val)**

**{**

**tree\_node \*root = (tree\_node\*)malloc(sizeof(tree\_node));**

**root->left = NULL;**

**root->right = NULL;**

**root->value = val;**

**}**

**/\***

**\*задание левого потомка**

**\*/**

**void set\_left(tree\_node \*root, int val)**

**{**

**root->left = init\_root(val);**

**}**

**/\***

**\*задание правого потомка**

**\*/**

**void set\_right(tree\_node \*root, int val)**

**{**

**root->right = init\_root(val);**

**}**

**/\***

**\*считывание дерева из консоли**

**\*/**

**tree\_node \*get\_tree()**

**{**

**int n;**

**tree\_node \*root = NULL,\*next,\*temp;**

**puts("Input integer numbers\nletter means that input is over");**

**if(scanf("%d",&n) == 1){**

**root = init\_root(n);**

**while(scanf("%d",&n) == 1)**

**{**

**next=temp=root;**

**while (next != NULL)**

**{**

**temp=next;**

**if (n < temp->value)**

**next = temp->left;**

**else next=temp->right;**

**}**

**if (n < temp->value)**

**set\_left(temp, n);**

**else**

**set\_right(temp, n);**

**}**

**}**

**return root;**

**}**

**/\***

**\*печать дерева в консоль**

**\*/**

**void print\_tree(tree\_node \*root, int space)**

**{**

**if (root == NULL)**

**return;**

**space += EDGE\_LEN;**

**print\_tree(root->right, space);**

**printf("\n");**

**for (int i = EDGE\_LEN; i < space; i++)**

**printf(" ");**

**printf("%d\n", root->value);**

**print\_tree(root->left, space);**

**}**

**/\***

**\*вернёт 1, если в дереве есть нечетное число, иначе 0**

**\*/**

**int is\_odd\_in\_tree(tree\_node \*root)**

**{**

**if (root->value % 2 != 0)**

**return 1;**

**if (root->left != NULL)**

**if (is\_odd\_in\_tree(root->left))**

**return 1;**

**if (root->right != NULL)**

**if (is\_odd\_in\_tree(root->right))**

**return 1;**

**return 0;**

**}**

**/\***

**\*вернет сумму нечетных чисел в дереве**

**\*/**

**int sum\_of\_odd\_in\_tree(tree\_node \*root)**

**{**

**int sum = 0;**

**if (root->value % 2 != 0)**

**sum += root->value;**

**if (root->left != NULL)**

**sum += sum\_of\_odd\_in\_tree(root->left);**

**if (root->right != NULL)**

**sum += sum\_of\_odd\_in\_tree(root->right);**

**return sum;**

**}**

# **Распечатка тестов к программе и результатов**

|  |  |  |
| --- | --- | --- |
| **№** | **Исходные данные** | **Результаты** |
| **Задание 1** | | |
| **1** |  | |
| **2** |  | |
| **Задание 2** | | |
| **2** | Input integer numbers  letter means that input is over  3 1 5 1 -1 d | 5  3  1  1  -1  Sum of odd numbers in tree is 9 |
|  | Input integer numbers  letter means that input is over  2 4 62 8 d | 62  8  4  2  There are no odd numbers |