**Отчет по лабораторной работе №24** по курсу фундаментальная информатика

Студентка группы М8О-113Б-22, № по списку 1, Астахова Анастасия Сергеевна

Контакты astakhovaanastasia0201@gmail.com

Работа выполнена: « » \_\_\_\_\_\_\_\_ 2023 г.

Преподаватель: доцент каф. 806 Никулин Сергей Петрович

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_2023 г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* 1. **Тема:** Дерево выражения \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Цель работы:** Составить программу на языке Си выполнения заданного преобразования с применением дерева..\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Задание (номер варианта 5):** Упростить выражение, выполнив возведение чисел в степень с целым показателем.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. **Оборудование** (лабораторное):

ЭВМ \_\_\_\_\_\_\_\_\_, процессор \_\_\_\_ \_\_, имя узла сети \_\_\_\_\_\_ с ОП \_\_\_\_\_\_\_\_ Мб, НМД \_\_\_\_\_\_\_ Мб. Терминал \_\_\_\_\_ адрес \_\_\_\_\_\_\_\_\_\_. Принтер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Другие устройства \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Оборудование ПЭВМ студента, если использовалось:*

Ноутбук: Процессор Intel Core i5-10210U с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь Bloody R80

*Компьютер: Процессор Ryzen 5 3600 с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь* Razer BASILISK X HYPERSPEED, [*Logitech G910 (920-008019) Orion Spectrum RGB Mechanical Gaming*](https://www.komus.ru/katalog/tekhnika/kompyuternaya-tekhnika/klaviatury-i-myshi/klaviatury-provodnye/klaviatura-logitech-g910-920-008019-orion-spectrum-rgb-mechanical-gaming/p/659916/?from=n-13776), монитор [*LG UltraGear 27GP750*](https://www.lg.com/ru/monitors/lg-27gp750-b)

1. **Программное обеспечение (лабораторное):**

Операционная система семейства \_\_\_Unix\_\_\_\_, наименование \_\_\_\_\_Ubuntu\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_4.15.0\_\_\_\_\_\_

интерпретатор команд \_\_\_\_bash\_\_\_\_\_\_\_\_ версия \_\_\_4.4.20\_\_\_\_

Система программирования \_\_С\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_последняя\_\_\_\_\_\_\_\_\_

Редактор текстов \_\_\_gedit\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_25.2.2 \_\_\_\_\_\_\_\_\_

Утилиты операционной системы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Прикладные системы и программы \_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных\_\_stud/208104 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**5.** *Программное обеспечение ПЭВМ студента, если использовалось****:***

Ноутбук: Операционная система семейства Ubuntu, наименование версия Ubuntu 20.04.3 LTS,

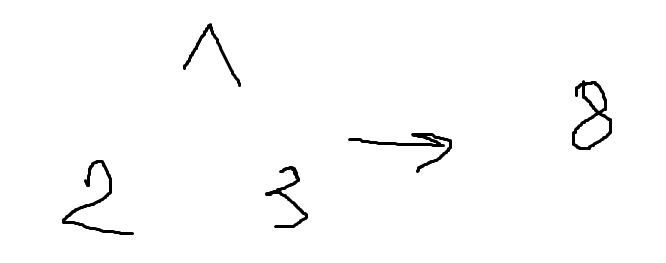
*интерпретатор команд bash версия 5.0.17. Система программирования C. Редактор текстов gedit версия 28.2. Утилиты: стандартные утилиты UNIX. Программное обеспечение: стандартное программное обеспечение системы UNIX.*

**6. Идея, метод, алгоритм** решения задачи(в формах:словесной,псевдокода,графической[блок-схема,диаграмма,рисунок,таблица] или формальные спецификации с пред- и постусловиями)

Программа будет содержать несколько функций, основные из которых:

1. Создание дерева
2. Преобразование дерева.

В этой функции должны выполняться следующие преобразования:



1. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию]
2. Изучу, как преобразовать выражение в дерево
3. Напишу функцию, которая бы преобразовывала нужные куски с возведением в степень

Тесты:

PS D:\c\_labs\lab\_24> ./main

Input your command:

1. Create tree.

2. Print tree.

3. Transform an expression.

4. Print expression.

5. Exit.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1

Input your expression

2^3^2

2

2

^

3

^

2

3

2

64

4

64

5

PS D:\c\_labs\lab\_24> ./main

Input your command:

1. Create tree.

2. Print tree.

3. Transform an expression.

4. Print expression.

5. Exit.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1

Input your expression

2^3+6^2

2

2

^

6

+

3

^

2

3

2

36

+

8

4

8+36

5

PS D:\c\_labs\lab\_24> ./main

Input your command:

1. Create tree.

2. Print tree.

3. Transform an expression.

4. Print expression.

5. Exit.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1

Input your expression

(2^3+a)\*(4^3^2-b)/x

2

x

/

b

-

2

^

3

^

4

\*

a

+

3

^

2

3

2

x

/

b

-

4096

\*

a

+

8

4

(8+a)\*(4096-b)/x

5

PS D:\c\_labs\lab\_24> ./main

Input your command:

1. Create tree.

2. Print tree.

3. Transform an expression.

4. Print expression.

5. Exit.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1

Input your expression

(4\*x-5\*y)\*24+7\*b

2

b

\*

7

+

24

\*

y

\*

5

-

x

\*

4

3

2

b

\*

7

+

24

\*

y

\*

5

-

x

\*

4

4

(4\*x-5\*y)\*24+7\*b

5

*Пункты 1-7 отчета составляются сторого до начала лабораторной работы.*

*Допущен к выполнению работы.* **Подпись преподавателя****\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами,подписанныйпреподавателем).

Код лабораторной работы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

typedef struct node {

char data[40];

struct node \*left, \*right;

} node;

int priority(char c) {

switch (c) {

case '+': case '-': return 1;

case '\*': case '/': return 2;

case '^': return 3;

default: return 100;

}

}

int power(int a, int b) {

int ans = 1;

for (int i = 0; i < b; i++) {

ans \*= a;

}

return ans;

}

int op(char data[40]) {

if (data[0] == '+' || data[0] == '-' ||

data[0] == '\*' || data[0] == '/' || data[0] == '^') {

return 1;

}

else {

return 0;

}

}

node \*make\_tree(char expr[], int first, int last) {

int min\_priority, k, prior, nest = 0;

node \*tree = (node\*)malloc(sizeof(node));

min\_priority = 100;

for (int i = first; i <= last; i++) {

if (expr[i] == '(') {

nest++;

}

else if (expr[i] == ')') {

nest--;

}

if (nest > 0) {

continue;

}

prior = priority(expr[i]);

if (prior <= min\_priority) {

min\_priority = prior;

k = i;

}

}

if (nest != 0) {

printf("Wrong expression\n");

exit(1);

}

if (min\_priority == 100) {

if (expr[first] == '(' && expr[last] == ')') {

free(tree);

return make\_tree(expr, first + 1, last - 1);

}

else {

k = last - first + 1;

for (int i = 0; i < k; i++) {

tree->data[i] = expr[first + i];

}

tree->data[k] = '\n';

tree->left = NULL;

tree->right = NULL;

return tree;

}

}

tree->data[0] = expr[k];

tree->data[1] = '\n';

tree->left = make\_tree(expr, first, k-1);

tree->right = make\_tree(expr, k+1, last);

return tree;

}

node \*parent(node \*tree, node \*son) {

if (tree == NULL) {

return NULL;

}

if (tree->left == son || tree->right == son) {

return tree;

}

node \*left = parent(tree->left, son);

node \*right = parent(tree->right, son);

return (left == NULL) ? right : left;

}

int is\_num(node \*nd) {

/\*

0, если операция

1, если х

2, если число

\*/

int n = 0;

while(nd->data[n] != '\n') {

if (!isdigit(nd->data[n])) {

if (nd->data[n] == '+' || nd->data[n] == '-' || nd->data[n] == '\*'

|| nd->data[n] == '/' || nd->data[n] == '^') {

return 0;

}

else {

return 1;

}

}

n++;

}

return 2;

}

void transform(node \*tree) {

if (tree == NULL) {

return;

}

if(tree->data[0] == '^') {

if (is\_num(tree->left) == 2 && is\_num(tree->right) == 2) {

int num1 = atoi(tree->left->data);

int num2 = atoi(tree->right->data);

int ans = power(num1, num2);

itoa(ans, tree->data, 10);

int n = 0;

while (ans > 0) {

ans /= 10;

n++;

}

tree->data[n] = '\n';

//tree->data[0] = power(num1, num2);

free(tree->left->left);

free(tree->left->right);

free(tree->right->left);

free(tree->right->right);

tree->left = NULL;

tree->right = NULL;

}

else if (is\_num(tree->left) == 0 && is\_num(tree->right) == 2) {

transform(tree->left);

transform(tree);

}

else if (is\_num(tree->left) == 2 && is\_num(tree->right) == 0) {

transform(tree->right);

transform(tree);

}

else if (is\_num(tree->left) == 0 && is\_num(tree->right) == 0) {

transform(tree->left);

transform(tree->right);

transform(tree);

}

else if (is\_num(tree->left) == 1 || is\_num(tree->right) == 1) {

return;

}

}

transform(tree->left);

transform(tree->right);

}

void print\_expression(node \*tree) {

if (tree == NULL) {

return;

}

if (op(tree->data) && op(tree->left->data) &&

(priority(tree->left->data[0]) < priority(tree->data[0]) ||

tree->left->data[0] == '^' && tree->data[0] == '^')) {

printf("(");

print\_expression(tree->left);

printf(")");

}

else {

print\_expression(tree->left);

}

for (int i = 0; i <= 40; i++) {

if (tree->data[i] != '\n') {

printf("%c", tree->data[i]);

continue;

}

break;

}

if (op(tree->data) && op(tree->right->data) &&

(priority(tree->right->data[0]) < priority(tree->data[0]) ||

tree->right->data[0] == '^' && tree->data[0] == '^')) {

printf("(");

print\_expression(tree->right);

printf(")");

}

else {

print\_expression(tree->right);

}

}

void print\_tree(node\* root, int n) {

if (root != NULL) {

print\_tree(root->right, n + 1);

for (int i = 0; i < n; i++) printf("\t");

printf("%s\n", root->data);

print\_tree(root->left, n + 1);

}

}

void print\_menu() {

printf("Input your command:\n");

printf("1. Create tree.\n");

printf("2. Print tree.\n");

printf("3. Transform an expression.\n");

printf("4. Print expression.\n");

printf("5. Exit.\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

}

int main() {

FILE \*inp, \*out;

inp = fopen("input.txt", "r");

out = fopen("output.txt", "w");

print\_menu();

node \*t = NULL;

while(1) {

int q;

//fscanf(inp, "%d", &q);

scanf("%d", &q);

if (q == 1) {

char expr[1000];

printf("Input your expression\n");

//fscanf(inp, "%s", &expr);

scanf("%s", &expr);

int n = 0;

while (expr[n] != '\0'){

n++;

}

t = make\_tree(expr, 0, n-1);

}

else if (q == 2) {

print\_tree(t, 0);

}

else if (q == 3) {

transform(t);

}

else if (q == 4) {

print\_expression(t);

printf("\n");

}

else if (q == 5) {

return 0;

}

}

}

1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные события(ошибки в сценарии и программе,нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  | или |  |  |  |  |  |
|  | дом. |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы:

Замечания отсутствуют.

1. **Выводы**

Я научилась преобразовывать выражение в бинарное дерево и делать преобразования в нем.

Недочёты при выполнении задания могут быть устранены следующим образом:

Недочеты отсутствуют.

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_