Отчёт по лабораторной работе№24 по курсу 1

студента группы М8О-108Б Попова Матвея, № по списку 18

Адреса www, e-mail, jabber, skype popov.m4tvei@yandex.ru

Работа выполнена: "21" апреля 2020г.

Преподаватель: Трубченко Никита Михайлович.

Входной контроль знаний с оценкой

Отчёт сдан "21" апреля 2021 г., итоговая оценка

Подпись преподавателя

- **Тема**: программа на языке Си

- **Цель работы**: научиться реализовывать деревья выражений
- **Задание (вариант 18)**: вывести из произведений упарные минусы
- **Оборудование** (*лабораторное*):
  ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____
  МБ
  НМД _____ ГБ. Терминал _____ адрес _____. Принтер _____
  Другие устройства _____

  Оборудование ПЭВМ студента, если использовалось:
  Процессор AMD Ryzen 7 4800U, ОП 16 ГБ, НМД 1 ТБ. Монитор _____
  Другие устройства _____

- **Программное обеспечение** (*лабораторное*):
  Операционная система семейства _____, наименование _____ версия _____
  Интерпретатор команд _____ версия _____
  Система программирования _____ версия _____
  Редактор текстов _____ версия _____
  Утилиты операционной системы _____
  Прикладные системы и программы _____
  Местонахождения и имена файлов программ и данных _____

  Программное обеспечение ЭВМ студента, если использовалось:
  Операционная система семейства _____, наименование _____ версия
  Интерпретатор команд _____ версия _____
  Система программирования _____ версия _____
  Редактор текстов _____ версия _____

Утилиты операционной системы _____

_____

Прикладные системы и программы _____

Местонахождения и имена файлов программ и данных _____

_____

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

```
#include <stdio.h> //#18
#include <stdlib.h>

struct node
{
   int k;
   char op;
   struct node *l;
   struct node *r;
};

struct node *add(struct node *t, int n, char op)
{
   if (t == NULL)
   {
      struct node *t = malloc(sizeof(struct node));
      t->k = n;
      t->op = op;
      t->r = NULL;
      t->l = NULL;
      return t;
   }

   if (n < t->k)
      t->l = add(t->l, n, op);
   if (n >= t->k)
      t->r = add(t->r, n, op);
   return t;
};

void inorder(struct node *t)
{
   if (t == NULL)
      return;
   inorder(t->l);
   if (t->op == '^')
      printf("*%d", t->k);
   else
      printf("*%c", t->op);
   inorder(t->r);
}

struct node *rem(struct node *t)
{
   if (t == NULL)
```

```
        return t;
    if (t->l == NULL && t->r == NULL)
    {
        free(t);
        return NULL;
    }
    if (t->r != NULL)
        t->r = rem(t->r);
    if (t->l != NULL)
        t->l = rem(t->l);
    return rem(t);
}

int main()
{
    struct node* T = NULL;
    int p = 0;
    char c = 'a';
    do
    {
        int n = 0;
        int rn = 0;
        int rd = 0;
        char f;
        scanf("%c", &c);
        if (c != '(')
        {
            f = c;
        }
        if (c == '(')
        {
            scanf("%c", &c);
            scanf("%c", &c);
            f = c;
            rd++;
        }
        if (c == '-')
        {
            scanf("%c", &c);
            f = c;
            rd++;
        }
        while ((c >= '0')&&(c <= '9'))
        {
            n *= 10;
            n += c - '0';
            scanf("%c", &c);
        }
        int g = 0;
        g = n;
        n = 0;
        while ((c != '+')&&(c != '/')&&(c != '\n')&&(c != '-'))
        {
            scanf("%c", &c);
            while ((c >= '0')&&(c <= '9'))
            {
```

```
        n *= 10;
        n += c - '0';
        scanf("%c", &c);
        rn++;
    }
    if (rn != 0)
    {
        T = add(T, n, '^');
        rn = 0;
        n = 0;
    }
    if ((c >= 'a')&&(c <= 'z'))
        T = add(T, 0, c);
    if (c == '(')
    {
        rd++;
        scanf("%c", &c);
        c = 'a';
    }
}
if ((rd % 2 == 0)&&(g == 0)&&(p == 0))
{
    printf("%c", f);
    inorder(T);
}
if ((rd % 2 == 0)&&(g == 0)&&(p > 0))
{
    printf("%c", f);
    inorder(T);
}
if ((rd % 2 == 0)&&(g != 0)&&(p == 0))
{
    printf("%d", g);
    inorder(T);
}
if ((rd % 2 == 0)&&(g != 0)&&(p > 0))
{
    printf("%d", g);
    inorder(T);
}
if ((rd % 2 == 1)&&(g == 0)&&(p == 0))
{
    printf("-%c", f);
    inorder(T);
}
if ((rd % 2 == 1)&&(g == 0)&&(p > 0))
{
    printf("(-%c", f);
    inorder(T);
    printf(")");
}
if ((rd % 2 == 1)&&(g != 0)&&(p == 0))
{
    printf("-%d", g);
    inorder(T);
}
```

```
        if ((rd % 2 == 1)&&(g != 0)&&(p > 0))
        {
           printf("(-%d", g);
           inorder(T);
           printf(")");
        }
        T = rem(T);
        printf("%c", c);
        p++;
    } while (c != '\n');
    T = rem(T);
    if (c != '\n')
        printf("\n");
    return 0;
}
```

*Пункты 1-7 отчёта составляются **строго до** начала лабораторной работы.*

Допущен к выполнению работы. Подпись преподавателя _____

- **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

- **Дневник отладки** должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) 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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

- Замечание автора по существу работы

- Выводы : реализовал дерево выражений
-

Недочеты, допущенные при выполнении задания, могут быть устранены следующим образом _____

Подпись студента _____