Московский авиационный институт

(Национальный исследовательский университет)

Курсовой проект

по курсу

«Языки и методы программирования»

2 семестр

Задание 6

Выполнил: Смирнов Никита

Группа: М8О-111Б-21

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022г.

Задание

Разработать последовательную структуру данных для представления простейшей базы данных на файлах СП Си в соответствии с заданным вариантом. Составить программу генерации внешнего нетекстового файла заданной структуры, содержащего представительный набор записей (15-20). Распечатать содержимое сгенерированного файла в виде таблицы и выполнить над ним заданное действие для 2-3 значений параметров запроса p и распечатать результат.

Действие по выборке данных из файла оформить в виде отдельной программы с параметрами запроса, вводимыми из стандартного входного текстового файла, или получаемых из командной строки Unix. Второй способ задания параметров обязателен для работ, оцениваемых на хорошо и отлично. Параметры задаются с помощью ключей-f (распечатка файла) или-p (параметры конкретного варианта задания). Получение параметров из командной строки производится с помощью стандартных библиотечных функций argc и argv.

Структуры данных и константы, совместно используемые программами, следует вынести в отдельный заголовочный файл.

В процессе отладки и тестирования рекомендуется использовать команды обработки текстовых файлов ОС Unix и переадресацию ввода-вывода. Сгенерированные и отформатированные тестовые данные необходимо заранее поместить в текстовые файлы и распечатывать при протоколировании. Число наборов тестовых данных должно быть не менее трех. Имя файла с бинарными данными является обязательным параметром второй программы.

Вариант 24.

Сведения о вступительных экзаменах абитуриентов: фамилия, инициалы, пол, номер школы, наличие медали, оценки в баллах, зачёт/незачёт по сочинению. Найти

абитуриентов, имеющих заданную сумму баллов ***p***

Структура проекта.

Проект состоит из 4 файлов:

1) **input.c** - сохраняет полученные данные в бинарный файл.

2) **output.c** - выводит содержимое бинарного файла в виде таблицы, находит абитуриентов,имеющих заданную сумму баллов p

3) **person.h** - заголовочный файл с описанием структуры.

4) **Data.txt** - входный файл с данными.

Формат программы и примеры работы.

Формат программы input.c:

./input <Входной текстовый файл> < Выходной бинарный файл>

Пример работы input.c:

```
~/Laba24$ cat Data.txt
Smirnov N.E M 1741 0 82 90 90 yes
Solovev S.G M 1741 1 78 76 95 yes
Risun M.G M 1741 0 84 76 78 yes
Dekkusheva A.E F 1741 1 78 96 90 yes
Ivanov P.A M 843 1 99 99 99 yes
Debilov I.G M 883 0 0 0 0 no
Krytov A.I M 228 1 100 100 100 yes
Vasilieva N.S F 6979 1 95 93 90 yes
```

~/Laba24$ ./input Data.txt base2.bin
~/Laba24$ ls
base2.bin input.c main   Makefile output.d replit.nix
Data.txt   input.d main.d output   output.o
input   input.o main.o output.c person.h
~/Laba24$ ./input
Введите ещё два аргумента в формате:
./input <Входной текстовый файл> <Выходной бинарный файл>

Формат работы программы output.c:

./output <Бинарный файл> <ключ -f/-p>

-f - вывести информацию о всех абитуриентах

-p - вывести абитуриентов, имеющих заданную сумму баллов p

Примеры работы:

```
~/Laba24$ ./output base2.bin -f
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
|    Фамилия    | Инициалы | Пол | Школа | Медаль | Математика | Русский | Информатика | Сочинение |
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
|        Smirnov|       N.E|    M|   1741|       0|          82|       90|           90|        yes|
|        Solovev|       S.G|    M|   1741|       1|          78|       76|           95|        yes|
|          Risun|       M.G|    M|   1741|       0|          84|       76|           78|        yes|
|     Dekkusheva|       A.E|    F|   1741|       1|          78|       96|           90|        yes|
|         Ivanov|       P.A|    M|    843|       1|          99|       99|           99|        yes|
|        Debilov|       I.G|    M|    883|       0|           0|        0|            0|         no|
|         Krytov|       A.I|    M|    228|       1|         100|      100|          100|        yes|
|      Vasilieva|       N.S|    F|   6979|       1|          95|       93|           90|        yes|
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
```

~/Laba24$ ./output base2.bin -f

~/Laba24$ ./output base2.bin -p

Введите p: 262

```
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
|    Фамилия    | Инициалы | Пол | Школа | Медаль | Математика | Русский | Информатика | Сочинение |
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
|        Smirnov|       N.E|    M|   1741|       0|          82|       90|           90|        yes|
+---------------+----------+-----+-------+--------+------------+---------+-------------+-----------+
```

~/Laba24$ ./output base2.bin -d

Неправильный параметр!

Программный код и тестовые данные.

input.c:

```
#include <stdio.h>
#include <stdlib.h>
#include "person.h"
int main(int argc , char* argv[]) {
   if(argc!=3) {
      printf("Введите ещё два аргумента в формате:\n./input <Входной текстовый
файл> <Выходной бинарный файл>\n");
      return 2;
   }
   Person per;
   FILE *output;
   FILE *input;
   input = fopen(argv[1],"r");
   output = fopen(argv[2],"wb");
   while(fscanf(input,"%s %s %s %hu %hd %hd %hd %hd %s\n",
      per.surname,
      per.init,
      per.gender,
      &per.school,
      &per.medal,
      &per.math,
      &per.russk,
      &per.informat,
      per.essay
   ) != EOF) {
      fwrite(&per,sizeof(per), 1, output);
   }
   fclose(input);
   fclose(output);
   return 0;
}
```

output.c:

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "person.h"

int main(int argc, char* argv[]) {
   int sum, avg, counter, p = 0;
   if(argc != 3) {
      printf("Введены
   неправильные параметры.
   Должно быть: programm
   <binfile> -p/-f \n");
```

```
    return 2;
}
FILE* input;
input = fopen(argv[1], "rb");
Person per;
if(strcmp(argv[2], "-f") == 0) {
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
    printf("|    Фамилия    |
Инициалы | Пол | Школа |
Медаль | Математика |
Русский | Информатика |
Сочинение |\n");
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
    while(fread(&per,
sizeof(per), 1, input)) {

printf("|%15s|%10s|%5s|%7hd|
%8hd|%12hd|%9hd|%13hd|%1
1s|\n",
        per.surname,
        per.init,
        per.gender,
        per.school,
        per.medal,
        per.math,
        per.russk,
        per.informat,
        per.essay
        );
    }
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
}
else if(strcmp(argv[2], "-p") ==
```

```
0) {
    printf("Введите p: ");
    scanf("%d", &p);
    input = fopen(argv[1], "rb");
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
    printf("|    Фамилия    |
Инициалы | Пол | Школа |
Медаль | Математика |
Русский | Информатика |
Сочинение |\n");
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
    while(fread(&per,
sizeof(per), 1, input)) {
        if((per.math + per.russk +
per.informat) == p) {

printf("|%15s|%10s|%5s|%7hd|
%8hd|%12hd|%9hd|%13hd|%1
1s|\n",
            per.surname,
            per.init,
            per.gender,
            per.school,
            per.medal,
            per.math,
            per.russk,
            per.informat,
            per.essay
            );
        }
    }
    printf("+---------------+--------
--+-----+-------+--------+---------
---+---------+-------------+--------
---+\n");
}
```

```
else {
    printf("Неправильный
параметр!\n");
    return 1;
}
return 0;
}
```

person.h:

```
#ifndef __person_h__

#define __person_h__

#include <stdio.h>
typedef struct {
   char surname[15]; // Фамилия
   char init[4]; // Инициалы
   char gender[2]; // Пол
   unsigned short int school; // Номер школы
   short int medal; // Общий вес вещей
   short int math; // Баллы по математике
   short int russk; // Баллы по русскому языку
   short int informat; // Баллы по информатике
   char essay[3]; // Информация о зачёте/незачёте по сочинению
} Person;
#endif
```

Data.txt:

Smirnov N.E M 1741 0 82 90 90 yes
Solovev S.G M 1741 1 78 76 95 yes
Risun M.G M 1741 0 84 76 78 yes
Dekkusheva A.E F 1741 1 78 96 90 yes
Ivanov P.A M 843 1 99 99 99 yes
Debilov I.G M 883 0 0 0 0 no
Krytov A.I M 228 1 100 100 100 yes
Vasilieva N.S F 6979 1 95 93 90 yes

Заключение.

Я составил последовательную структуру данных для представления простейшей базы данных на файлах в СП Си на примере информации о пассажирах самолета. Реализовал программу, сохраняющую данные в бинарный файл, программу, выводящую данные из бинарного файла в виде таблицы и со специальным условием. Я освоил работу со структурами в Си, работу с файлами, с бинарными файлами и основы реализации системы базы данных.

Использованные источники.

1. Керниган Б., Ритчи Д. Язык программирования Си = The C programming language. — 2-е изд. — М.: Вильямс, 2007. —С. 304. —ISBN 0-13-110362-8.
2. Эндрю Таненбаум, Structured Computer Organization,

Московский авиационный институт

(Национальный исследовательский университет)

Курсовой проект

по курсу

«Языки и методы программирования»

2 семестр

Задание 7

Выполнил: Смирнов Никита

Группа: М8О-111Б-21

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022г.

Задание

Составить программу на языке Си с функциями для обработки квадратных разряженных матриц с элементами целого типа, предусмотреть выполнение следующих стандартных функций.

Стандартные:

1. Ввод матрицы
2. Печать матрицы в обычном виде
3. Печать матрицы во внутреннем виде
4. Произвести операцию над матрицей
5. Выход из программы

Вариант физического хранения 1 - отображение на массивы

Вариант схемы размещения 1 - Цепочка ненулевых элементов в векторе a

Вариант преобразования 10 - вычислить матрочлен (a*M + b*E), где E - единичная матрица, a и b - числа, вводимые пользователем

Структура проекта.

Проект состоит из 3 файлов:

1) **main.c** - Меню.

2) **matrix.c** - Функции для работы с разряженной матрицей

3) **matrix.h** - заголовочный файл с описанием функций.

Формат программы и примеры работы.

```
./main
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 1

Введите количество строк: 3

Введите количество столбцов: 3
1 0 0
0 0 0
2 0 4
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 2
1   0   0
0   0   0
2   0   4
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 3
0 -1 3
0 1 -1 0 2 6 2 4 -1
0 1 -1 0 2 6 2 4 -1
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 4

Введите число a: 3

Введите число b: 7


Результат работы функции - вычисление матрочлена a*A + b*E - матрица

Матрица - результат:

10  0   0
0   7   0
6   0   19
Меню
```

```
6   0   19
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 1

Введите количество строк: 5

Введите количество столбцов: 5
1 0 0 0 0
0 0 0 0 0
0 0 2 3 4
0 0 0 5 0
0 0 0 6 0
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 2
1   0   0   0   0
0   0   0   0   0
0   0   2   3   4
0   0   0   5   0
0   0   0   6   0
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 3
0 -1 3 12 15
0 1 -1 2 2 6 3 3 9 4 4 -1 3 5 -1 3 6 -1
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 4

Введите число a: 2

Введите число b: 3


Результат работы функции - вычисление матрочлена a*A + b*E - матрица

Матрица - результат:

5   0   0   0   0
0   3   0   0   0
0   0   7   6   8
0   0   0   13  0
0   0   0   12  3
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 1

Введите количество строк: 8

Введите количество столбцов: 8
0 0 1 0 0 0 0 0
1 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0
0 0 0 4 0 0 0 0
0 0 0 0 0 0 0 0
0 5 0 0 0 0 0 0
0 0 0 0 0 8 0 0
0 0 0 0 6 0 0 0
```

```
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 2
0   0   1   0   0   0   0   0
1   0   0   0   0   0   1   0
0   0   0   0   0   0   0   0
0   0   0   4   0   0   0   0
0   0   0   0   0   0   0   0
0   5   0   0   0   0   0   0
0   0   0   0   0   8   0   0
0   0   0   0   6   0   0   0
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 3
0 3 -1 9 -1 12 15 18
2 1 -1 0 1 6 6 1 -1 3 4 -1 1 5 -1 5 8 -1 4 6 -1
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 4
Введите номер операции: 4

Введите число a: 6

Введите число b: 9


Результат работы функции - вычисление матрочлена a*A + b*E - матрица

Матрица - результат:

9   0   6   0   0   0   0   0
6   9   0   0   0   0   6   0
0   0   9   0   0   0   0   0
0   0   0   33  0   0   0   0
0   0   0   0   9   0   0   0
0   30  0   0   0   9   0   0
0   0   0   0   0   48  9   0
0   0   0   0   36  0   0   9
Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 6

Такой команды нет!!

Меню
1) Ввести матрицу
2) Вывести матрицу в обычном виде
3) Вывести матрицу в соответствии с схемой размещения
4) Произвести операцию над матрицой
5) Выйти
Введите номер операции: 5
```

Программный код.

main.c:

```
#include <stdio.h>
#include "matrix.h"
#include <stdlib.h>

void print_menu(){
   printf("Меню\n1) Ввести матрицу\n2) Вывести матрицу в обычном виде\n3)
Вывести матрицу в соответствии с схемой размещения\n4) Произвести операцию
над матрицой\n5) Выйти\nВведите номер операции: ");
}


int main(void) {
   int n_1, m_1, flag = 1, num, cv = 0;
   int massa[10], massm[40];
   while (flag){
      print_menu();
      scanf("%d", &num);
      switch(num){
         case 1:{
            printf("\nВведите количество строк: ");
            scanf("%d", &n_1);
            printf("\nВведите количество столбцов: ");
            scanf("%d", &m_1);
            int am = -1;
            cv = 0;
            for (int i=0; i<n_1; i++){
               for(int j = 0; j<m_1;j++){
                  int nb;
                  scanf("%d", &nb);
                  if (nb != 0 && am != i){
                     massa[i] = cv;
                     massm[cv] = j;
                     massm[cv+1] = nb;
                     massm[cv+2] = -2;
                     cv += 2;
                     am++;
                  }
                  else if (nb != 0 && am == i){
                     massm[cv] = cv + 1;
                     massm[cv+1] = j;
                     massm[cv+2] = nb;
                     massm[cv+3] = -2;
                     cv += 3;
                  }
               }
               if (am != i){
```

```
                massa[i] = -1;
                am++;
            }
            if (massm[cv] == -2){
                massm[cv] = -1;
                cv++;
            }
        }
        break;
        }
    case 2:{
        printmatr(massa, massm, n_1, m_1, cv);
        break;
    }
    case 3:{
        prntadj(massa, massm, n_1, cv);
        break;
    }
    case 4:{
        int a, b;
        printf("\nВведите число a: ");
        scanf("%d", &a);
        printf("\nВведите число b: ");
        scanf("%d", &b);
        procedure(massa, massm, a, b, n_1, m_1, cv);
        break;
    }
    case 5:{
        flag = 0;
        break;
    }
    default:{
        printf("\nТакой команды нет!!\n\n");
        break;
    }
  }
 }
 return 0;
}
```

matrix.c:

```
#include "matrix.h"
#include <stdio.h>

void prntadj(int *a, int *m, int
  n_1, int cv){
  for (int i=0; i <n_1; i++){
```

```
        printf("%d ", (int)*(a + i));
        }
    printf("\n");
    for (int i =0; i < cv; i++){
        printf("%d ", (int)*(m+i));
        }
    printf("\n");
}

void printmatr(int a[], int m[], int
    n, int st, int cv){
    for (int i=0; i< n; i++){
        if (a[i] == -1){
            for (int t=0; t< st;t++){
                printf("0\t");
            }
        }
        else{
            int stat = -2;
            for(int j = 0;j < st;j++){
                if (m[a[i]] == j && stat
    == -2){
                    printf("%d\t",  m[a[i]
    + 1]);
                    stat = m[a[i] + 2];

                }
                else if(stat == -1 || stat
    == -2 || m[stat] != j){
                    printf("0\t");
                }
                else if(stat != -2 &&
    m[stat] == j){
                    printf("%d\t",
    m[stat+1]);
                    stat = m[stat + 2];
                }

            }
        }
        printf("\n");
    }
```

```
}

void procedure(int a[], int m[], int
    am, int bm, int n, int st, int cv){
    printf("\n\nРезультат работы
    функции - вычисление
    матрочлена a*A + b*E -
    матрица\n\nМатрица -
    результат:\n\n");
    for (int i=0; i< n; i++){
        if (a[i] == -1){
            for (int t=0; t< st;t++){
                if (i == t){
                    printf("%d\t", bm);
                }
                else{
                    printf("0\t");
                }
            }
        }
        else{
            int stat = -2;
            for(int j = 0;j < st;j++){
                if (m[a[i]] == j && stat
    == -2){
                    if (i==j){
                        printf("%d\t", am *
    m[a[i] + 1] + bm);
                    }
                    else{
                        printf("%d\t",  am
    * m[a[i] + 1]);
                    }
                    stat = m[a[i] + 2];

                }
                else if (i==j && stat
    <0){
                    printf("%d\t", bm);
                }
                else if(stat == -1 || stat
    == -2 || m[stat] != j){
```

```
                printf("0\t");
            }
            else if(stat > 0 &&
    m[stat] == j && i!=j){
                printf("%d\t",
    am*m[stat+1]);
                stat = m[stat+2];
            }
            else if(stat != -2 &&
    m[stat] == j){
                printf("%d\t", am *
    m[stat + 1] + bm);
                stat = m[stat + 2];
            }

        }
      }
      printf("\n");
    }
}
}
```

matrix.h:

```
#ifndef __matrix_h__
#define __matrix_h__


void prntadj(int *a, int *m, int n_1, int cv);
void printmatr(int a[], int m[], int n, int st, int cv);
void procedure(int a[], int m[], int am, int bm, int n, int st, int cv);
#endif
```

Заключение.

Я разобрал, что такое разреженные матрицы, научился задавать, работать, обрабатывать их. Считаю, что данные знания пригодятся мне в будущем.

Московский авиационный институт

(Национальный исследовательский университет)

Курсовой проект

по курсу

«Языки и методы программирования»

2 семестр

Задание 8

Выполнил: Смирнов Никита

Группа: М8О-111Б-21

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022г.

Задание

Составить программу на языке Си для обработки двунаправленного списка с отображением на динамические структуры. Предусмотреть реализацию 4 стандартных действий и одного нестандартного.

Стандартные:

1. Печать списка
2. Вставка нового элемента в список
3. Удаление элемента из списка
4. Подсчёт длины списка

Тип элемента списка - строковый

Вид списка - линейный двунаправленный

Нестандартное действие - удалить элементы списка со значениями, находящимися в заданном диапазоне(в качестве диапазона выбрана первая буква строки)

Программа должна вводить значения элементов неупорядоченной таблицы и проверять работу процедуры сортировки в трёх случаях:

Структура проекта.

Проект состоит из 3 файлов:

1) **main.c** - Меню.

2) **list.c** - Функции списка

3) **list.h** - заголовочный файл с описанием структуры.

Формат программы и примеры работы.

 ./main
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1

Список пуст!
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 4
Список пуст

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
abcdefg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
bbc
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
cdef
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 4
Длина списка: 3

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg bbc cdef
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 3
Введите индекс элемента: 1

Готово

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg cdef
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 5
Введите значения диапазона в формате [a, b]
Вводите символы с которых должна начинаться строка
Введите первое значение диапазона: c

Введите второе значение диапазона: c

Готово

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
ff
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
ggg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
hhh
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg ff ggg hhh
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 4
Длина списка: 4

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка

5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 5
Введите значения диапазона в формате [a, b]
Вводите символы с которых должна начинаться строка
Введите первое значение диапазона: b

Введите второе значение диапазона: r

Готово

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
b
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
c
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
d
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
e
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg b c d e
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 4
Длина списка: 5

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 5
Введите значения диапазона в формате [a, b]
Вводите символы с которых должна начинаться строка

Введите первое значение диапазона: b

Введите второе значение диапазона: e

Готово

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
b
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 2
c
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1
abcdefg b c
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 5
Введите значения диапазона в формате [a, b]
Вводите символы с которых должна начинаться строка
Введите первое значение диапазона: a

Введите второе значение диапазона: z

Готово

1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 1

Список пуст!
1: Напечатать список 2: Вставить новый элемент в список
3: Удалить элемент из списка 4: Посчитать длину списка
5: Удалить значения из списка в диапазоне 6: Выйти
Введите номер операции
Номер == 6

Программный код и текстовые данные.

main.c:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "ctype.h"
#include "lsst.h"


void printCommands() {
   printf("1: Напечатать список\t2: Вставить новый элемент в список\n3: Удалить
элемент из списка\t4: Посчитать длину списка\n5: Удалить значения из списка в
диапазоне\t6: Выйти\nВведите номер операции\nНомер == ");
}

int is_empty(Dblist *lst){
   return lst->size == 0;
}

int main(void) {
   Dblist *list = createList();
   int flag = 1, rule, c = 1, ind;
   char sym, a, b;
   char *string;// Объявление строки, которая будет заполнятся символами
   int i = 0;
   bool countSpace = false;
   while (flag) {
      printCommands();
      scanf("%d", &rule);
      switch (rule) {
         case 1:
            if (!is_empty(list)){
               printList(list);
            }
            else{
               printf("\nСписок пуст!\n");
            }
            break;
         case 2:
            getchar();
            string = (char *)malloc(sizeof(char));
            while (c) {
               sym = getchar();
               if (sym == '\n'){
                  string[i] = '\0';
                  addElem(list, string); //добавление строки в список
                  i = 0;
                  break;
```

```
                }
            else if (sym >= 35 && sym <= 126){
                string[i] = sym;
                i++;
                string = realloc(string, sizeof(char) * (i + 1));
                }
            }
        // string = (char *)realloc(string, sizeof(char) * (i + 1));
            break;
        case 3:
            if (!is_empty(list)){
                printf("Введите индекс элемента: ");
                scanf("%d", &ind);
                if (ind < list->size && ind >= 0){
                    delElem(list, ind);
                    printf("\nГотово\n\n");
                }
                else{
                    printf("\nОшибка, введён некорректный индекс!\n\n");
                }
            }
            else{
                printf("\nСписок пуст!\n");
            }

            break;
        case 4:
            if (!is_empty(list)){
                printf("Длина списка: %d\n\n", list->size);
            }
            else{
                printf("Список пуст\n\n");
            }
            break;
        case 5:
            printf("Введите значения диапазона в формате [a, b]\nВводите символы с
которых должна начинаться строка\n");
            getchar();
            printf("Введите первое значение диапазона: ");
            scanf("%c", &a);
            getchar();
            printf("\nВведите второе значение диапазона: ");
            scanf("%c", &b);
            if (tolower(a) <= tolower(b)){
                delDiap(list, tolower(a), tolower(b));
                printf("\nГотово\n\n");
            }

            break;
```

```
            case 6:
                flag = 0;
                break;
            }
        }
    }
```

table.c:

```
    #include "lsst.h"
    #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>



    Dblist* createList() {
        Dblist *tmp = (Dblist
        *)malloc(sizeof(Dblist));
        tmp->size = 0;
        tmp->head = tmp->end =
        NULL;

        return tmp;
    }


    //Добавление в конец списка
    void addElem(Dblist *list, char
        *value) {
        element *tmp = (element
        *)malloc(sizeof(element));
        tmp->str = (char
        *)malloc(sizeof(char) *
        (strlen(value) + 1));
        tmp->str = strcpy(tmp->str,
        value);
        tmp->next = NULL;
        tmp->prev = list->end;

        if (list->end) {
            list->end->next = tmp;
        }
```

```
    list->end = tmp;

    if (list->head == NULL) {
        list->head = tmp;
    }

    list->size++;
}


void delElem(Dblist *list, int
    index){
    element *side = list->head;
    int flag = 0;
    if (list->size != 0 && side !=
    NULL){
        int flag = 0;
        for(int i=0; i<list->size;i++){
            if (i == index){
                if (side != list->end &&
    side != list->head){
                    side->prev->next =
    side->next;
                    side->next->prev =
    side->prev;
                    free(side);
                    flag = 1;
                    break;
                }
                else if(side == list->end
    && side != list->head){
                    side->prev->next =
    NULL;
                    list->end = side-
    >prev;
                    free(side);
                    flag = 1;
                    break;
                }
                else if (side == list-
    >head && side != list->end){
                    side->next->prev =
```

```
    NULL;
                list->head = side-
    >next;
                free(side);
                flag = 1;
                break;
            }
            else if(list->size == 1){
                free(side);
                list->head = list->end
    = NULL;
                flag = 1;
                break;
            }
        }
        else{
            side = side->next;
        }
    }
    if (flag){
        list->size--;
    }
  }
  else{
    printf("Список пуст\n");
  }
}


void delDiap(Dblist *list, char a,
    char b){
    element *now = list->head;
    for(int i = 0; i<list->size;i++){
        // printf("%c", now->str[0]);
        char check = now->str[0];
        if (check >= a && check <=
    b){
            if (i == 0){
                delElem(list, i);
                i = -1;
            }
            else{
```

```
                delElem(list, i);
                i--;
            }
        }
        now = now->next;
    }
}

//Вывод списка
void printList(Dblist *list) {
    element *tmp = list->head;

    while (tmp) {
        printf("%s ", tmp->str);
        tmp = tmp->next;
    }
    printf("\n");
}
```

list.h:

```
#ifndef __lsst_h__
#define __lsst_h__
#include <stdio.h>
#include "stdlib.h"
#include <stdbool.h>

typedef struct element {
   char *str;
   struct element *prev;
   struct element *next;
} element;

typedef struct Dblist {
   int size;
   element *head;
   element *end;
} Dblist;

Dblist* createList();
void addElem(Dblist *list, char *value);
void printList(Dblist *list);
void delElem(Dblist *list, int index);
void delDiap(Dblist *list, char a, char b);

#endif
```

Заключение.

Я научился составлять программы на языке Си с использованием процедур и функций для работы с списками, научился их реализовывать.

Московский авиационный институт

(Национальный исследовательский университет)

Курсовой проект

по курсу

«Языки и методы программирования»

2 семестр

Задание 9

Выполнил: Смирнов Никита

Группа: М8О-111Б-21

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022г.

Задание

Составить программу на языке Си с использованием процедур и функций для сортировки таблицы заданным методом и двоичного поиска по ключу в таблицу.

Программа должна вводить значения элементов неупорядоченной таблицы и проверять работу процедуры сортировки в трёх случаях:

1) Элементы таблицы с самого начала упорядочены.

2) Элементы таблицы расставлены в обратном порядке

3) Элементы таблицы не упорядочены

В последнем случае можно использовать встроенные процедуры генерации псевдослучайных чисел.

Для каждого вызова процедуры сортировки необходимо печатать исходное состояние таблицы и результаты сортировки. После выполнения сортировки программа должна вводить ключи и для каждого из них выполнять поиск в упорядоченной таблице с помощью процедуры двоичного поиска и печатать найденные элементы, если они присутствуют в таблице.

Метод сортировки: Сортировка слиянием

Структура таблицы:

Тип ключа: целый

Длина ключа байтах: 8

Хранение данных и ключей: вместе

Структура проекта.

Проект состоит из 4 файлов:

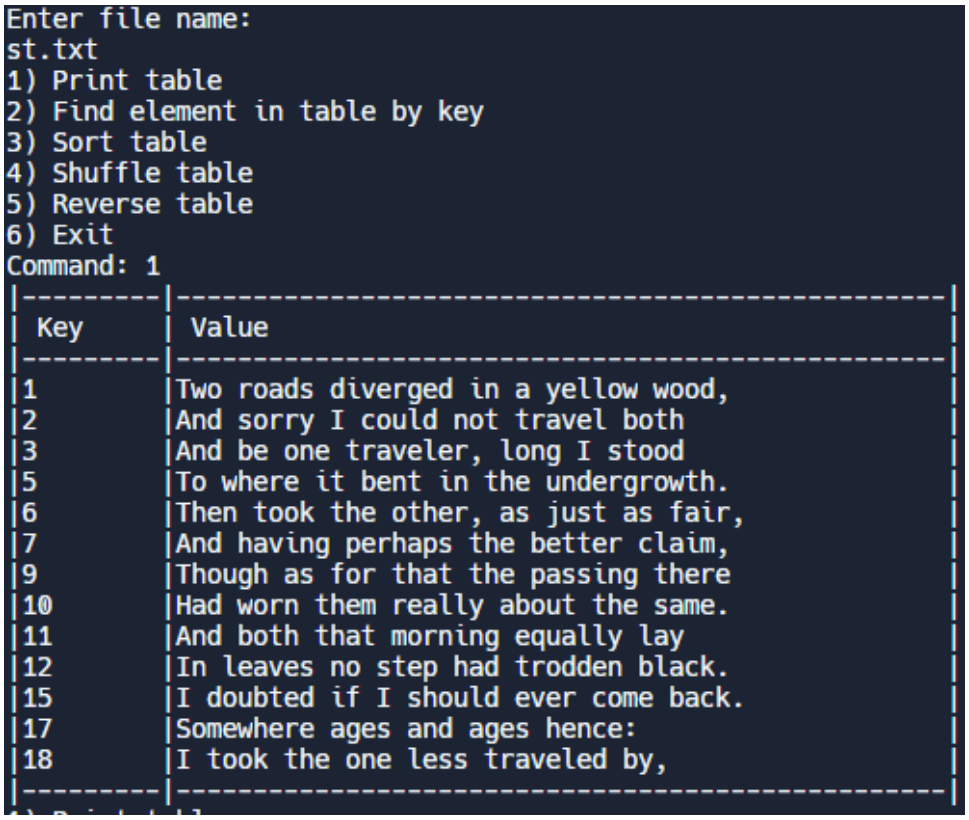
1) **main.c** - Меню, считывание файла.

2) **table.c** - Функции таблицы

3) **table.h** - заголовочный файл с описанием структуры.

4) **st.txt** - входный файл с данными.

Формат программы и примеры работы.



Начальная таблица

```
|---------|-------------------------------------------------|
| Key     | Value                                           |
|---------|-------------------------------------------------|
|1        |Two roads diverged in a yellow wood,             |
|2        |And sorry I could not travel both                |
|3        |And be one traveler, long I stood                |
|5        |To where it bent in the undergrowth.             |
|6        |Then took the other, as just as fair,            |
|7        |And having perhaps the better claim,             |
|9        |Though as for that the passing there             |
|10       |Had worn them really about the same.             |
|11       |And both that morning equally lay                |
|12       |In leaves no step had trodden black.             |
|15       |I doubted if I should ever come back.            |
|17       |Somewhere ages and ages hence:                   |
|18       |I took the one less traveled by,                 |
|---------|-------------------------------------------------|
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 2
Enter key
10
Element found "Had worn them really about the same." by key 10
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 2
Enter key
17
Element found "Somewhere ages and ages hence:  " by key 17
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 2
Enter key
1
Element found "Two roads diverged in a yellow wood," by key 1
```

Нахождение по ключу

```
|---------|-------------------------------------------------|
| Key     | Value                                           |
|---------|-------------------------------------------------|
|1        |Two roads diverged in a yellow wood,             |
|2        |And sorry I could not travel both                |
|3        |And be one traveler, long I stood                |
|5        |To where it bent in the undergrowth.             |
|6        |Then took the other, as just as fair,            |
|7        |And having perhaps the better claim,             |
|9        |Though as for that the passing there             |
|10       |Had worn them really about the same.             |
|11       |And both that morning equally lay                |
|12       |In leaves no step had trodden black.             |
|15       |I doubted if I should ever come back.            |
|17       |Somewhere ages and ages hence:                   |
|18       |I took the one less traveled by,                 |
|---------|-------------------------------------------------|
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 3
Table sorted
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 1
|---------|-------------------------------------------------|
| Key     | Value                                           |
|---------|-------------------------------------------------|
|1        |Two roads diverged in a yellow wood,             |
|2        |And sorry I could not travel both                |
|3        |And be one traveler, long I stood                |
|5        |To where it bent in the undergrowth.             |
|6        |Then took the other, as just as fair,            |
|7        |And having perhaps the better claim,             |
|9        |Though as for that the passing there             |
|10       |Had worn them really about the same.             |
|11       |And both that morning equally lay                |
|12       |In leaves no step had trodden black.             |
|15       |I doubted if I should ever come back.            |
|17       |Somewhere ages and ages hence:                   |
|18       |I took the one less traveled by,                 |
|---------|-------------------------------------------------|
```

Сортировка отсортированной таблицы

```
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 4
Table shuffled
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 1
|---------|-------------------------------------------------
| Key     | Value
|---------|-------------------------------------------------
|10       |Had worn them really about the same.
|2        |And sorry I could not travel both
|3        |And be one traveler, long I stood
|5        |To where it bent in the undergrowth.
|11       |And both that morning equally lay
|15       |I doubted if I should ever come back.
|1        |Two roads diverged in a yellow wood,
|18       |I took the one less traveled by,
|12       |In leaves no step had trodden black.
|7        |And having perhaps the better claim,
|6        |Then took the other, as just as fair,
|17       |Somewhere ages and ages hence:
|9        |Though as for that the passing there
|---------|-------------------------------------------------
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 3
Table sorted
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 1
```

Перемешивание таблицы и её сортировка:

```
Command: 3
Table sorted
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 1
|---------|-------------------------------------------------|
| Key     | Value                                           |
|---------|-------------------------------------------------|
|1        |Two roads diverged in a yellow wood,             |
|2        |And sorry I could not travel both                |
|3        |And be one traveler, long I stood                |
|5        |To where it bent in the undergrowth.             |
|6        |Then took the other, as just as fair,            |
|7        |And having perhaps the better claim,             |
|9        |Though as for that the passing there             |
|10       |Had worn them really about the same.             |
|11       |And both that morning equally lay                |
|12       |In leaves no step had trodden black.             |
|15       |I doubted if I should ever come back.            |
|17       |Somewhere ages and ages hence:                   |
|18       |I took the one less traveled by,                 |
|---------|-------------------------------------------------|
```

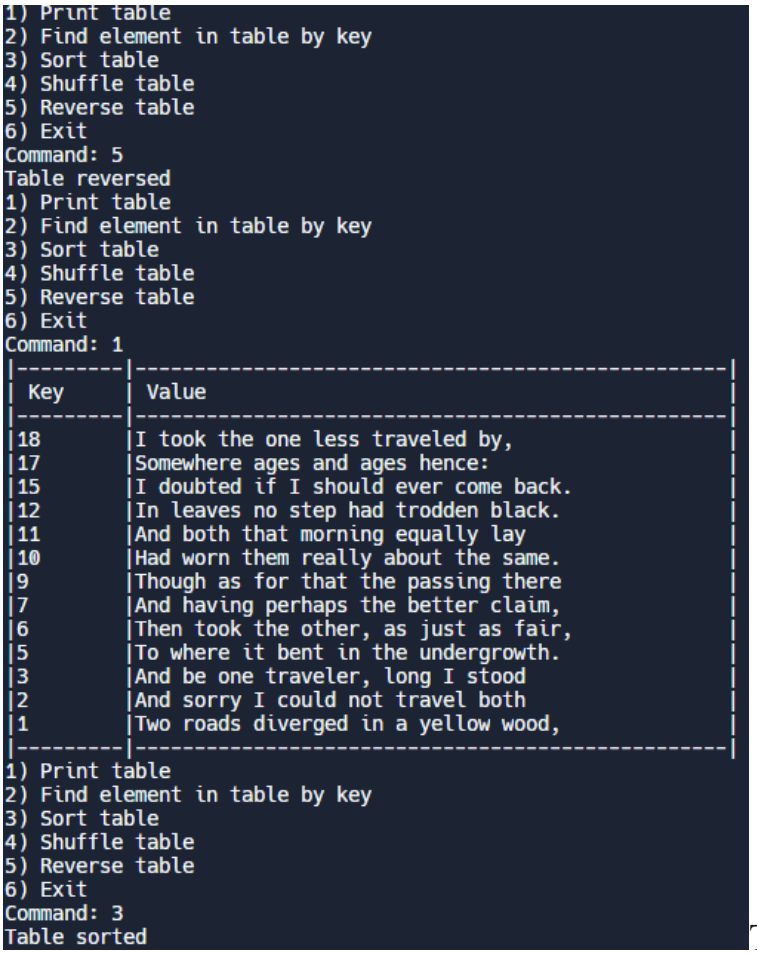


Таблица в обратном порядке

Пример сортировки обратной таблицы и выход из программы

```
Command: 3
Table sorted
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 1
|---------|-------------------------------------------------|
| Key     | Value                                           |
|---------|-------------------------------------------------|
|1        |Two roads diverged in a yellow wood,             |
|2        |And sorry I could not travel both                |
|3        |And be one traveler, long I stood                |
|5        |To where it bent in the undergrowth.             |
|6        |Then took the other, as just as fair,            |
|7        |And having perhaps the better claim,             |
|9        |Though as for that the passing there             |
|10       |Had worn them really about the same.             |
|11       |And both that morning equally lay                |
|12       |In leaves no step had trodden black.             |
|15       |I doubted if I should ever come back.            |
|17       |Somewhere ages and ages hence:                   |
|18       |I took the one less traveled by,                 |
|---------|-------------------------------------------------|
1) Print table
2) Find element in table by key
3) Sort table
4) Shuffle table
5) Reverse table
6) Exit
Command: 6
```

Программный код и текстовые данные.

main.c:

```
#include <stdio.h>
#include "table.h"


void printCommands() {
   printf("1) Print table\n2) Find element in table by key\n3) Sort table\n4) Shuffle
table\n5) Reverse table\n6) Exit\nCommand: ");
}


int main() {
   int i, cnt, rule, key, flag = 1, N=50;
   char ch;
   row arr[N];
   char filename[100];
   printf("Enter file name:\n");
   scanf("%s", filename);
   FILE *file = fopen(filename, "r");
   if (file == NULL) {
      printf("File does not exist!\n");
      return 0;
   }
   i = 0;
   while (i < N && fscanf(file, "%d", &arr[i]._key) == 1) {
      fscanf(file, "%c", &ch);
      getRow(file, arr[i]._str, sizeof(arr[i]._str));
      i++;
   }
   fclose(file);
   cnt = i;
   printCommands();
   while (flag) {
      scanf("%d", &rule);
      switch (rule) {
         case 1:
            printTable(arr, cnt);
            printCommands();
            break;
         case 2:
            if (!isSorted(arr, cnt)) {
               printf("Table not sorted!\n");
            }
            else {
```

```
printf("Enter key\n");
scanf("%d", &key);
i = binSearch(arr, cnt, key);
```

```
                if (i > -1) {
                    printf("Element found \"%s\" by key %d\n", arr[i]._str,key);
                }
                else
                    printf("Element not found\n");
                }
                printCommands();
                break;
            case 3:
                mergesort(arr, 0, cnt - 1);
                printf("Table sorted\n");
                printCommands();
                break;
            case 4:
                printf("Table shuffled\n");
                shuffle(arr, cnt);
                printCommands();
                break;
            case 5:
                printf("Table reversed\n");
                reverse(arr, cnt);
                printCommands();
                break;
            case 6:
                flag = 0;
                return 0;
        }
    }
}
```

table.c:

```
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "table.h"


void printTable(row *arr, int size)
    {
    printf("|_____|_____
    _____|\n");
    printf("| %-7s | %-48s |\n",
    "Key", "Value");
    printf("|_____|_____
```

```
    ____________________|\n");
    for (int i = 0; i < size; i++){
        printf("|%-9d|%-50s|\n",
    arr[i]._key, arr[i]._str);
    }
    printf("|_____|_____
    ____________________|\n");
}


void getRow(FILE *stream, char
    *str, int size) {
    int cnt = 0;
    char ch;
    while ((ch = getc(stream)) !=
    '\n' && cnt < size - 1) {
        str[cnt++] = ch;
        str[cnt] = '\0';
    }
}


void swapRows(row *r1, row
    *r2) {
    row tmp;
    tmp = *r1;
    *r1 = *r2;
    *r2 = tmp;
}


int binSearch(row *arr, int size,
    int key) {
    int start = 0, end = size - 1,
    mid;
    if (size <= 0) {
        return -1;
    }
    while (start < end) {
    mid = start + (end - start) / 2;
        if (arr[mid]._key == key)
            return mid;
```

```
        else
            if (arr[mid]._key < key)
                start = mid + 1;
        else
            end = mid;
    }
    if (arr[end]._key == key)
        return end;
    return -1;
}


void sort(row *arr, int first, int
    last)
{

    int left_iterator, right_iterator,
    middle;
    row mas[last - first + 1];
    middle = (first + last) / 2;
    left_iterator = first;
    right_iterator = middle + 1;

    for(int j = first; j <= last; j++){
        if ((left_iterator <= middle)
    && ((right_iterator > last) ||
    (arr[left_iterator]._key <
    arr[right_iterator]._key))){
            mas[j - first] =
    arr[left_iterator];
            left_iterator++;
        }
        else{
            mas[j - first] =
    arr[right_iterator];
            right_iterator++;
        }
    }
    for(int j=first; j <=last; j++){
        arr[j] = mas[j - first];
    }
}
```

```
void mergesort(row *arr, int start,
    int end){
    if (start < end){
        mergesort(arr, start, (start +
    end) / 2);
        mergesort(arr, (start + end)/2
    + 1, end);
        sort(arr, start, end);
    }
}
int randomtwo(const int a, const
    int b) {
    return a + rand() % (b - a + 1);
}

void shuffle(row *arr, const int
    size) {
    int i, j, k;
    srand((unsigned int)time(0));
    for (k = 0; k < size; k++) {
        i = randomtwo(0, size - 1);
        j = randomtwo(0, size - 1);
        swapRows(&arr[i], &arr[j]);
    }
}
void reverse(row *arr, int size) {
    int i, j;
    for (i = 0, j = size - 1; i < j; i++,
    j--)
        swapRows(&arr[i], &arr[j]);
}
int isSorted(row *arr, int size) {
    for (int i = 0; i < size - 1; i++){
        if (arr[i]._key > arr[i +
    1]._key){
            return 0;
        }
    }
    return 1;
}
```

table.h:

```
de <stdio.h>
#include "table.h"
#
i
f
n
d
e
f
T
A
B
L
E
_
H
#
d
e
f
i
n
e
T
A
B
L
E
_
H
#
i
n
c
l
u
```

```
#include "table.h"

typedef struct _row {
int _key;
char _str[100];
} row;

void printTable(row *arr, int size);
void getRow(FILE *stream, char *str, int size);
void swapRows(row *r1, row *r2);
int binSearch(row *arr, int size, int key);
void mergesort(row *arr, int first, int last);
void shuffle(row *arr, int size);
void reverse(row *arr, int size);
int randomtwo(int a, int b);
int isSorted(row *arr, int size);
#endif
```

st.txt - файл с стихотворением:

1 Two roads diverged in a yellow wood,
2 And sorry I could not travel both
3 And be one traveler, long I stood
5 To where it bent in the undergrowth.
6 Then took the other, as just as fair,
7 And having perhaps the better claim,
9 Though as for that the passing there
10 Had worn them really about the same.
11 And both that morning equally lay
12 In leaves no step had trodden black.
15 I doubted if I should ever come back.
17 Somewhere ages and ages hence:
18 I took the one less traveled by,

Заключение.

Я научился составлять программы на языке Си с использованием процедур и функций для сортировки таблицы заданным методом и двоичного поиска по ключу в таблицу. Также я повторил сортировку слиянием.

Отчет по лабораторной работе № 20 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 9, 22, 33

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: « » _____2022г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Стандартные утилиты UNIX для обработки файлов
2. **Цель работы:** Ознакомится с стандартными утилитами Unix для обработки файлов
3. **Задание (9-tail, 22*-awk, 33-xargs):**
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Linux, наименование Ubuntu, версия 22.04
Прикладные системы и программы: терминал ОС UNIX, текстовый редактор emacs

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

Cmp - побайтовое сравнение двух файлов
Tail - вывод заданного количества строк с конца файла
Awk - обработка и фильтрация текста
Xargs – объединение нескольких команд в конвейер
Sort - вывод текстовых строк в определённом порядке
Comm - сравнение двух текстовых файлов с отсортированными по алфавиту строками
Wc - подсчитывает количество строк или слов в тексте
File - показывает тип файла
Head - выводит начальные строки файла
Diff - сравнение файлов между собой
Join - объединяет строки из двух файлов на основе наличия общего поля
Tee - запись вывода команды в один или несколько файлов
Grep - ищет строки в файлах, фильтрует вывод команд и т.д

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

1) Включим виртуальную машину
2) Зайдём в терминал
3) Начнём работу
4) Закончим работу, выключим виртуальную машину

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

```
nikita@ubuntu:~$ cd /home/nikita/Documents
nikita@ubuntu:~/Documents$ cat f1.txt
you will blow up
boy previous door
nikita@ubuntu:~/Documents$ cat f2.txt
I will blow up
a minute later
nikita@ubuntu:~/Documents$ cmp f1.txt f2.txt
f1.txt f2.txt differ: byte 1, line 1
nikita@ubuntu:~/Documents$ cmp -l f1.txt f2.txt
1 171 111
2 157 40
3 165 167
4 40 151
5 167 154
6 151 154
7 154 40
8 154 142
9 40 154
10 142 157
11 154 167
12 157 40
13 167 165
14 40 160
15 165 12
16 160 141
17 12 40
18 142 155
19 157 151
20 171 156
21 40 165
22 160 164
23 162 145
24 145 40
25 166 154
26 151 141

27 157 164
28 165 145
29 163 162
30 40 12
cmp: EOF on f2.txt after byte 30
nikita@ubuntu:~/Documents$ sort f1.txt > f3
nikita@ubuntu:~/Documents$ sort f2.txt > f4
nikita@ubuntu:~/Documents$ sort -c f3
nikita@ubuntu:~/Documents$ comm f3 f4
a minute later
boy previous door
I will blow up
you will blow up
nikita@ubuntu:~/Documents$ cat f3
boy previous door
you will blow up
```

```
nikita@ubuntu:~/Documents$ cat f4
a minute later
I will blow up
nikita@ubuntu:~/Documents$ comm -l f3 f4
comm: invalid option -- 'l'
Try 'comm --help' for more information.
nikita@ubuntu:~/Documents$ comm -1 f3 f4
a minute later
I will blow up
nikita@ubuntu:~/Documents$ wc f3
2 7 35 f3
nikita@ubuntu:~/Documents$ wc -c f3
35 f3
nikita@ubuntu:~/Documents$ wc -m f3
35 f3

nikita@ubuntu:~/Documents$ wc -l f3
2 f3
nikita@ubuntu:~/Documents$ wc -w f3
7 f3
nikita@ubuntu:~/Documents$ diff f1 f2
diff: f1: No such file or directory
diff: f2: No such file or directory
nikita@ubuntu:~/Documents$ diff f1.txt f2.txt
1,2c1,2
< you will blow up
< boy previous door
---
> I will blow up
> a minute later
nikita@ubuntu:~/Documents$ diff -b f1.txt f2.txt
1,2c1,2
< you will blow up
< boy previous door
---
> I will blow up
> a minute later
nikita@ubuntu:~/Documents$ cat f2.txt
I will blow up
a minute later
nikita@ubuntu:~/Documents$ grep ccc f1.txt
nikita@ubuntu:~/Documents$ grep cc f1.txt*
nikita@ubuntu:~/Documents$ grep -in up f3*
2:you will blow up
nikita@ubuntu:~/Documents$ grep up f1.txt
you will blow up
nikita@ubuntu:~/Documents$ cat f3
boy previous door
you will blow up
nikita@ubuntu:~/Documents$ cat f4
a minute later
I will blow up
nikita@ubuntu:~/Documents$ join f3 f4
nikita@ubuntu:~/Documents$ cat > join f3 f4
nikita@ubuntu:~/Documents$ cat join
boy previous door
you will blow up
```

a minute later
I will blow up
nikita@ubuntu:~/Documents$ cat < t1
bash: t1: No such file or directory
nikita@ubuntu:~/Documents$ cat > t1
testted testy
nikita@ubuntu:~/Documents$ cat t1
testted testy
nikita@ubuntu:~/Documents$ tee t1
1223
1223
123123
123123
333
333
nikita@ubuntu:~/Documents$ cat t1
1223
123123
333
nikita@ubuntu:~/Documents$ tee -a t1

======
======
....
....
nikita@ubuntu:~/Documents$ cat t1
1223
123123
333

======
....
nikita@ubuntu:~/Documents$ tr 1223 666 <t1
6666
666666
666

======
....
nikita@ubuntu:~/Documents$ cat > t1
111 555 666
run away
nikita@ubuntu:~/Documents$ tr -d 111 <t1
555 666
run away
nikita@ubuntu:~/Documents$ cat > t2
111 111 111
222 222 222
343 567 890
nikita@ubuntu:~/Documents$ uniq t2
111 111 111
222 222 222
343 567 890
nikita@ubuntu:~/Documents$ cat > t2
111 222 333

```
111 222 333
=== === ===
nikita@ubuntu:~/Documents$ uniq t2
111 222 333
=== === ===
nikita@ubuntu:~/Documents$ cat f3
boy previous door
you will blow up
nikita@ubuntu:~/Documents$ od f3
0000000 067542 020171 071160 073145 067551 071565 062040 067557
0000020 005162 067571 020165 064567 066154 061040 067554 020167
0000040 070165 000012
0000043
nikita@ubuntu:~/Documents$ od -d f3
0000000 28514 8313 29296 30309 28521 29557 25632 28527
0000020 2674 28537 8309 26999 27756 25120 28524 8311
0000040 28789 10
0000043
nikita@ubuntu:~/Documents$ sum f3
56976 1
nikita@ubuntu:~/Documents$ sum f4
55801 1
nikita@ubuntu:~/Documents$ awk '{print $2}' f3
previous
will
nikita@ubuntu:~/Documents$ awk '{ print $2" " $3}' f3
previous door
will blow
nikita@ubuntu:~/Documents$ ls -l f*| awk '{ print $5" " $9}'
35 f1.txt
30 f2.txt
35 f3
30 f4
nikita@ubuntu:~/Documents$ head -2 f1.txt
you will blow up
boy previous door
nikita@ubuntu:~/Documents$ head -2c f3
bonikita@ubuntu:~/Documents$ head -2c f1.txt
yonikita@ubuntu:~/Documents$ head -2 f4
a minute later
I will blow up
nikita@ubuntu:~/Documents$ head -2c f4
a nikita@ubuntu:~/Documents$ file f4
f4: ASCII text
nikita@ubuntu:~/Documents$ file -l f4
Set 0:
Binary patterns:
Text patterns:
Set 1:
Binary patterns:
Text patterns:
Set 0:
Binary patterns:
Strength = 500@47: Biosig/Brainvision Marker file [biosig/brainvision]
Strength = 490@122: Biosig/TMSiLOG [biosig/tmsilog]
Strength = 461@127: Biosig/SYNERGY [biosig/synergy]
Strength = 460@46: Biosig/Brainvision V-Amp file []
```

Strength = 410@45: Biosig/Brainvision data file []
Strength = 380@6: OpenSSH private key []

Strength = 361@107: EICAR virus test files []
Strength = 360@19: Biosig/ATES MEDICA SOFT. EEG for Windows [biosig/ates]
Strength = 350@315: SketchUp Model [application/vnd.sketchup.skp]
Strength = 340@631: sc68 Atari ST music []
Strength = 340@35: T64 tape Image []
Strength = 340@40: T64 tape Image []
Strength = 340@19: Erlang JAM file - version 4.3 []
Strength = 340@51: Mathematica binary file []
Strength = 340@66: Bazaar merge directive []
Strength = 340@92: SQLite 2.x database []
Strength = 340@136: Paged COBALT boot rom []
Strength = 331@280: NetImmerse game engine file []
Strength = 330@1586: LyNX archive []
Strength = 330@61: FrameMaker IPL file [application/x-mif]
Strength = 330@267: Gamebryo game engine file []
Strength = 330@61: PGP public key block [application/pgp-keys]
Set 1:
Binary patterns:
Text patterns:
nikita@ubuntu:~/Documents$ file -i f3
f3: text/plain; charset=us-ascii
nikita@ubuntu:~/Documents$ file -b f4
ASCII text
nikita@ubuntu:~/Documents$ file -s f3
f3: ASCII text
nikita@ubuntu:~/Documents$ find -name 'f*'
./f1.txt
./f2.txt
./f3
./f4
nikita@ubuntu:~/Documents$ ls
f1.txt f2.txt f3 f4 join t1 t2
nikita@ubuntu:~/Documents$ paste f3 f4
boy previous door a minute later
you will blow up I will blow up
nikita@ubuntu:~/Documents$ paste -s f3 f4
boy previous door you will blow up
a minute later I will blow up
nikita@ubuntu:~/Documents$ split -1 3 f1.txt

split: cannot open '3' for reading: No such file or directory
nikita@ubuntu:~/Documents$ split -1 f3
nikita@ubuntu:~/Documents$ cat f3
boy previous door
you will blow up
nikita@ubuntu:~/Documents$ ls
f1.txt f2.txt f3 f4 join t1 t2 xaa xab
nikita@ubuntu:~/Documents$ cat xaa
boy previous door
nikita@ubuntu:~/Documents$ cat xab
you will blow up
nikita@ubuntu:~/Documents$ ls ~/Pictures | xargs -t -L1 -d - echo
echo 'kot1.jpeg'$'\n''kot2.jpeg'$'\n''kot3.jpg'$'\n'
kot1.jpeg

kot2.jpeg
kot3.jpg

nikita@ubuntu:~/Documents$ find ~/Pictures -name "kot*" -type f -print0 | xargs -t -0 -L1 echo
echo /home/nikita/Pictures/kot2.jpeg
/home/nikita/Pictures/kot2.jpeg
echo /home/nikita/Pictures/kot1.jpeg
/home/nikita/Pictures/kot1.jpeg
echo /home/nikita/Pictures/kot3.jpg
/home/nikita/Pictures/kot3.jpg
nikita@ubuntu:~/Documents$ xargs

9. **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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. **Замечания автора** по существу работы: -
11. **Выводы:**

Я изучил способы использования утилит для обработки файлов в системе ОС UNIX и реализовал их на практике.

Подпись студента
_____

Отчет по лабораторной работе № 21 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 8

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: «06» __мая_____2021г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Программирование на интерпретируемых командных языках
2. **Цель работы:** Научиться программировать на интерпретируемом командном языке (bash)
3. **Задание (вариант 8 + доп.условие 1 + доп.условие 3):** Создание синонима всех файлов с указанным суффиксом и числом связей, большим 1, путем перестановки суффиксов в именах исходных файлов в начало и удаления точки. (Если параметры опущены, то они должны быть запрошены у пользователя, если указан параметр «?», то выводится подсказка (спецификация программы))
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Linux, наименование Ubuntu, версия 22.04
Прикладные системы и программы: терминал ОС UNIX, текстовый редактор emacs

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

Перед началом работы создадим файлы и ссылки на них. Если на вход подан один аргумент, то программа выводит подсказку о необходимости введения второго аргумента - суффикса. Если аргументы не введены, то программа выводит подсказку с необходимостью ввести два аргумента - путь к директории, суффикс. Если на вход подан «?», то выводится подсказка (спецификация программы) и программа завершается.
Напишем проверку вводимых параметров программы. После напишем функцию find, внутри которой запустим обход заданной директории с проверкой для подбора необходимых файлов с указанным суффиксом и числом связей больших 1. соответственно создаём синоним файла при соблюдении всех условий и выводим в терминал полный путь до файлов, к которым мы создаём синонимы (для наглядности).

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

Создадим несколько файлов с разными именами и расширениями.
Выполним скрипт lab21.sh
Сначала вызовем его с неверным количеством аргументов, чтобы увидеть подсказку с правильным синтаксисом.
Потом протестируем программу, подавая на вход разные суффиксы.

Список файлов:
f1.txt
f2.txt
f3.txt
f3
f4

Список ссылок:
f1link
f2link
f3link
f4link
f5link

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

Отчёт по 21 лабораторной работе, вариант 8

Задание:
Создание синонима всех файлов с указанным суффиксом и числом связей, большим 1, путём перестановки суффиксов в именах исходных файлов в начало и удаления точки.
Программа

```
#!/bin/bash
dir=$1
suff=$2
el="?"
#Проверка параметров
if [[ -z $dir ]]
then
echo "Для работы программы требуется ввести директорию и суффикс файла!"
exit
elif [[ $dir = $el ]]
then
echo "Правильный порядок ввода: ./myscript директория суффикс"
exit
elif [[ -z $suff ]]
then
echo "Для работы программы требуется ввести суффикс файла"
exit
fi
cd $dir
find() {
array=$(ls $PWD/*)
for file in $array
do
#Если встретили файл
if [[ -f $file ]] && [[ $(stat --format=%h $file) -gt 1 ]] && [[ ${file: -3} == $suff ]]
then
ln $file $dir"/$suff$(basename $file ".$suff")"
echo $file
fi
cd ..
done
}
find
```

Протокол

```
nikita@ubuntu:~/Documents$ cat > f1.txt
abcdefg
nikita@ubuntu:~/Documents$ cat > f2.txt
ggrfe
nikita@ubuntu:~/Documents$ cat > f3.txt
abcdefghigl;o
nikita@ubuntu:~/Documents$ touch f4
nikita@ubuntu:~/Documents$ touch f5
nikita@ubuntu:~/Documents$ ln f1.txt f1link
nikita@ubuntu:~/Documents$ ln f2.txt f2link
nikita@ubuntu:~/Documents$ ln f3.txt f3link
nikita@ubuntu:~/Documents$ ln f4 f4link
nikita@ubuntu:~/Documents$ ln f5 f5link
```

```
nikita@ubuntu:~/Documents$ ls -l
total 28
-rw-rw-r-- 2 nikita nikita 8 May 6 08:17 f1link
-rw-rw-r-- 2 nikita nikita 8 May 6 08:17 f1.txt
-rw-rw-r-- 2 nikita nikita 6 May 6 08:17 f2link
-rw-rw-r-- 2 nikita nikita 6 May 6 08:17 f2.txt
-rw-rw-r-- 2 nikita nikita 14 May 6 08:17 f3link
-rw-rw-r-- 2 nikita nikita 14 May 6 08:17 f3.txt
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f4
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f4link
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f5
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f5link
-rwxrwxrwx 1 nikita nikita 932 May 6 08:15 myscript
nikita@ubuntu:~/Documents$ ./myscript /home/nikita/Documents txt
/home/nikita/Documents/f1.txt
/home/nikita/Documents/f2.txt
/home/nikita/Documents/f3.txt
nikita@ubuntu:~/Documents$ ls -l
total 40
-rw-rw-r-- 3 nikita nikita 8 May 6 08:17 f1link
-rw-rw-r-- 3 nikita nikita 8 May 6 08:17 f1.txt
-rw-rw-r-- 3 nikita nikita 6 May 6 08:17 f2link
-rw-rw-r-- 3 nikita nikita 6 May 6 08:17 f2.txt
-rw-rw-r-- 3 nikita nikita 14 May 6 08:17 f3link
-rw-rw-r-- 3 nikita nikita 14 May 6 08:17 f3.txt
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f4
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f4link
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f5
-rw-rw-r-- 2 nikita nikita 0 May 6 08:18 f5link
-rwxrwxrwx 1 nikita nikita 932 May 6 08:15 myscript
-rw-rw-r-- 3 nikita nikita 8 May 6 08:17 txtf1
-rw-rw-r-- 3 nikita nikita 6 May 6 08:17 txtf2
-rw-rw-r-- 3 nikita nikita 14 May 6 08:17 txtf3
nikita@ubuntu:~/Documents$ ./myscript ?
Правильный порядок ввода: ./myscript директория суффикс
nikita@ubuntu:~/Documents$ ./myscript /home/nikita/Documents
Для работы программы требуется ввести суффикс файла
nikita@ubuntu:~/Documents$ ./myscript
Для работы программы требуется ввести директорию и суффикс файла!
```

9. **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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. **Замечания автора** по существу работы: -
11. **Выводы:**

    Я изучил способы написания программ на интерпретируемом языке ОС UNIX и реализовал их на практике.

Подпись студента

_____

Отчет по лабораторной работе № 22 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 139-140

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: « » _____2022г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Издательская система Tex
2. **Цель работы:** Научиться вёрстке на TeX
3. **Задание (страницы 139-140):**
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Windows, наименование Windows 11 pro
Прикладные системы и программы: браузер Opera GX, онлайн-редактор LaTex, текстовый редактор NotePads.

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

1. Подключим все необходимы для вёрстки пакеты, определим класс

```
\documentclass[14pt, a4paper]{article}

\usepackage[T2A]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[english, russian]{babel}
\usepackage{amsmath,amsfonts,amssymb,amsthm,mathtools}
\usepackage{indentfirst}
\usepackage{geometry}
```

2. Для больших формул будем использовать конструкцию

```
\begin{equation(*)} (* необязательна и нужна лишь для введения счётчика)
….....
\end{equation(*)}
```

3. Для многострочных формул будем использовать

```
\begin{multline(*)} (* необязательна и нужна лишь для введения счётчика)
….....
\end{multline(*)}
```

4. Для использования дробей используем

```
\frac{числитель}{знаменатель}
```

5. Для использования корней используем

```
\sgqrt{выражение под корнем}
```

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

1. Зарегистрируемся в онлайн редакторе Overleaf
2. Подключим все необходимые пакеты и начнём работу
3. После окончания вёрстки просмотрим результат в пдф файле, который компилируется автоматически
4. Сравним исходные страницы с теми, что мы получили.
5. Скачаем пдф файл и скопируем tex файл для отчёта

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

```
\documentclass[14pt, a4paper]{article}

\usepackage[T2A]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[english, russian]{babel}
\usepackage{amsmath,amsfonts,amssymb,amsthm,mathtools}
\usepackage{indentfirst}
\usepackage{geometry}

\geometry{top=31mm}
\geometry{bottom=34mm}
\geometry{left=31mm}
\geometry{right=30mm}

\begin{document}
\pagestyle{empty}

\section*{\small §2. \textbf{Частные производные и дифференциалы функции векторного аргумента}
\qquad\quad{\textnormal 139}}
\noindent и полагая $t = 1$, получаем требуемое равенство. $\blacktriangleright$

   \textbf{67.} Пусть $x^2 = v w$, $y^2 = u v$ и $f(x, y, z) = F(u, v, w)$. Доказать, что $x f_x' + y f_y + z f_x' = u F_u' + v F_v' + w F_w'$.

   $\blacktriangleleft$ Согласно условию, имеем
   \begin{equation*}
      F(u, v, w) = f(\sqrt{v w}, \sqrt{u w}, \sqrt{u v}).
   \end{equation*}

   \noindent Дифференцируя это равенство по $u$, $v$ и $w$, находим

   \begin{equation}
      F_u' = f_y'\frac{w}{2\sqrt{u v}} + f_x'\frac{v}{2\sqrt{u v}}, F_v' = f_x'\frac{w}{2\sqrt{v w}} + f_z'\frac{u}{2\sqrt{u v}}, F_w' = f_x'\frac{v}{2\sqrt{v w}} + f_y'\frac{u}{2\sqrt{u w}}.
   \end{equation}

   \noindent Умножая первое из равенств (1) на $u$, второе на $v$, а третье на $w$ и складывая их, получаем
   \begin{multline*}
      u F_u' + v F_v' + w F_w' = f_y'\frac{u w}{2\sqrt{u w}} + f_z'\frac{u v}{2\sqrt{u v}} + f_x'\frac{v w}{2\sqrt{v w}} + \\ f_z'\frac{u v}{2\sqrt{u v}} + f_x'\frac{v w}{2\sqrt{v w}} + f_y'\frac{u w}{2\sqrt{u w}} = \sqrt{v w}f_x' + \sqrt{u w}f_y' + \sqrt{u v}f_z'.
   \end{multline*}
   \noindent Отсюда, используя условие задачи, окончательно находим
   \begin{equation*}
      u F_u' + v F_v' + w F_w' = x f_x' + y f_y' + z f_z'.\blacktriangleright
   \end{equation*}

   Путём последовательного дифференцирования исключить произвольные функции $\phi$ и $\psi$:
```

\textbf{68.} $z = x + \phi (x y)$.

$\blacktriangleleft$ Найдем частные производные по x и по y:
\begin{equation*}
\frac{\partial z}{\partial x} = 1 + y \phi', \frac{\partial z}{\partial y} = x \phi'.
\end{equation*}
\noindent Сложим полученные равенства, умножив первое из них на x, а второе на $-y$. Тогда получим
\begin{equation*}
\frac{\partial z}{\partial x} - y \frac{\partial z}{\partial y} = x. \blacktriangleright
\end{equation*}

\quad \textbf{69.} $u = \phi (x - y, y - z)$.

\quad $\blacktriangleleft$ Имеем $\frac{\partial u}{\partial x} = \phi_1'$, $\frac{\partial u}{\partial y} = -\phi_1' + \phi_2'$, $\frac{\partial u}{\partial z} = -\phi_2'$. Складывая эти равенства, получаем

$\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} + \frac{\partial u}{\partial z} = 0$.
$\blacktriangleright$

\quad\textbf{70.} $z = \phi (x) \psi (y)$.

\quad$\blacktriangleleft$ Имеем $\frac{\partial z}{\partial x} = \phi' \psi$, $\frac{\partial z}{\partial y} = \phi \psi'$. Отсюда $\frac{\partial z}{\partial x} \frac{\partial z}{\partial y} = \phi \psi \phi' \psi' = z \phi' \psi'$.

\quadС другой стороны, $\frac{\partial^2 z}{\partial x \partial y} = \phi' \psi'$. Следовательно, из последних двух равенст непосредственно вытекает, что $\frac{\partial z}{\partial x} \frac{\partial z}{\partial y} = z \frac{\partial^2 z}{\partial x \partial y}$. $\blacktriangleright$

\quad\textbf{71.} $z = \phi (x y) + \psi (\frac{x}{y})$.

\quad$\blacktriangleleft$ Используя равенства

\noindent
\begin{equation*}
\frac{\partial z}{\partial x} = y \phi' + \frac{1}{y} \psi', \frac{\partial^2 z}{\partial x^2} = y^2 \phi'' + \frac{1}{y^2} \psi'', \frac{\partial z}{\partial y} = x \phi' - \frac{x}{y^2} \psi', \frac{\partial^2 z}{\partial y^2} = x^2 \phi'' + \frac{x^2}{y^4} \psi'' + \frac{2x}{y^3} \psi',
\end{equation*}

\noindent получаем следующие соотношения:

\noindent
\begin{equation*}
x \frac{\partial z}{\partial x} - y \frac{\partial z}{\partial y} = \frac{2x}{y} \psi', x^2 \frac{\partial^2 z}{\partial x^2} - y^2 \frac{\partial^2 z}{\partial y^2} = -\frac{2x}{y} \psi',
\end{equation*}

\noindent из которых непосредственно вытекает, что
\begin{equation*}

x^2 \frac{\partial^2 z}{\partial x^2} - y^2 \frac{\partial^2 z}{\partial y^2} + x \frac{\partial z}{\partial x} - y \frac{\partial z}{\partial y} = 0. \blacktriangleright
\end{equation*}

\small \textbf{72. Найти производную функции $\mathbf{z = x^2 - y^2}$ в точке $\mathbf{M = (1, 1)}$ в направлении $\mathbf{l}$, составляющем угол $\mathbf{\alpha = 60^\circ}$ с положительным направлением оси $\mathbf{O x}$}.

\newpage

$\blacktriangleleft$ Имеем $\frac{\partial z (M)}{\partial l} = \frac{\partial z (M)}{\partial x} cos \alpha + \frac{\partial z (M)}{\partial y} cos \beta = 2cos \alpha - 2cos \beta$. Таким образом, $\frac{\partial z(M)}{\partial l} = \\
1 - \sqrt{3}$. $\blacktriangleright$

\textbf{73.} Найти производную функции $z = ln(x^2 + y^2)$ в точке $M = (x_0, y_0)$ в направлении l, \\ перпендикулярном к линии уровня, проходящей через эту точку.

$\blacktriangleleft$ Поскольку вектор $grad \ u$ в точке M ортогонален к линии уровня $c = ln(x^2 + y^2)$,\\проходящей через точку M, то направляющие косинусы вектора l равны направляющим косинусам $grad u$ в точке M, т. е.

\begin{equation*}
cos \alpha = \frac{\frac{\partial z(M)}{\partial x}}{||grad \ u(M)||}, \quad cos \beta = \frac{\frac{\partial z(M)}{\partial y}}{||grad \ u(M)||}.
\end{equation*}

\noindent Но $\frac{\partial z(M)}{\partial x} = \frac{2x_0}{x_0^2 + y_0^2}$, $\frac{\partial z(M)}{\partial y} = \frac{2y_0}{x_0^2 + y_0^2}$,
\begin{equation*}
||grad \ u(M)|| = \sqrt{\Biggl(\frac{\partial z(M)}{\partial x}\Biggr)^2 + \Biggl(\frac{\partial z(M)}{\partial y}\Biggr)^2} = \frac{2}{\sqrt{x_0^2 + y_0^2}},
\end{equation*}

\noindent поэтому $cos \alpha = \frac{x_0}{\sqrt{x_0^2 + y_0^2}}$, $cos \beta = \frac{y_0}{\sqrt{x_0^2 + y_0^2}}$. Следовательно,
\begin{equation*}
\frac{\partial z(M)}{\partial l} = \frac{\partial z(M)}{\partial x} cos\alpha + \frac{\partial z(M)}{\partial y} cos \beta = \frac{2}{\sqrt{x_0^2 + y_0^2}} \quad (x_0^2 + y_0^2 \not = 0). \blacktriangleright
\end{equation*}

\textbf{74.} Найти производную функции $z = 1 - (\frac{x^2}{a^2} + \frac{y^2}{b^2})$ в точке $M = (\frac{a}{\sqrt{2}}, \frac{b}{\sqrt{2}})$ по \\ направлению внутренней нормали к этой точке к кривой $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$.

$\blacktriangleleft$ Тангенс угла наклона нормали к данной кривой определяется формулой $tg \alpha = - \frac{1}{y'(\frac{a}{\sqrt{2}})})$, \\ где $y = \frac{b}{a}\sqrt{a^2 - x^2}$. Отсюда $tg \alpha = \frac{a}{b}$, а направляющие косинусы внутренней нормали выражаются формулами $cos \alpha = - \frac{b}{\sqrt{a^2

+ b^2}}$, $cos \beta = - \frac{a}{\sqrt{a^2 + b^2}}$(мы берем знак минус, поскольку нормаль внутренняя). Воспользуемся формулой производной по направлению $n = (cos \alpha; cos \beta)$:

\begin{equation*}
  \frac{\partial z(M)}{\partial n} = \frac{\partial z(M)}{\partial x} cos \alpha + \frac{\partial z(M)}{\partial y} cos \beta.
\end{equation*}

\noindent Вычисляя производные $\frac{\partial z(M)}{\partial x} = - \frac{\sqrt{2}}{a}$, $\frac{\partial z(M)}{\partial y} = - \frac{\sqrt{2}}{b}$, находим
\begin{equation*}
  \frac{\partial z(M)}{\partial n} = \frac{b\sqrt{2}}{a\sqrt{a^2 + b^2}} + \frac{a\sqrt{2}}{b\sqrt{a^2 + b^2}} = \frac{\sqrt{2(a^2 + b^2)}}{ab}. \blacktriangleright
\end{equation*}

\textbf{75.} Найти производную функции $u = x y z$ в точке $M = (1, 1, 1)$ в направлении $l = \\ (cos\alpha, cos\beta, cos\gamma)$. Чему равна величина градиента функции в этой точке?

$\blacktriangleleft$ Очевидно, $\frac{\partial u(M)}{\partial x} = 1$, $\frac{\partial u(M)}{\partial y} = 1$, $\frac{\partial u(M)}{\partial z} = 1$. По формуле производной по направлению, получим
\begin{equation*}
  \frac{\partial u(M)}{\partial l} = \frac{\partial u(M)}{\partial x} cos\alpha + \frac{\partial u(M)}{\partial y} cos\beta + \frac{\partial u(M)}{\partial z} cos\gamma = cos\alpha + cos\beta + cos\gamma.
\end{equation*}

Величину градиента определим по формуле
\begin{equation*}
  ||grad \ u(M)|| = \sqrt{\Biggl(\frac{\partial u(M)}{\partial x}\Biggr)^ 2 + \Biggl(\frac{\partial u(M)}{\partial y}\Biggr)^2 + \Biggl(\frac{\partial u(M)}{\partial z}\Biggr)^2} = \sqrt{3}. \blacktriangleright
\end{equation*}

\textbf{76. Определить угол между градиентами функции $\mathbf{u = x^2+y^2+z^2}$ в точках $\mathbf{A = (\varepsilon, 0, 0)}$ и $\mathbf{B = (0, \varepsilon, 0)}$.}
\end{document}

и полагая $t = 1$, получаем требуемое равенство. ►

67. Пусть $x^2 = vw$, $y^2 = uv$ и $f(x, y, z) = F(u, v, w)$. Доказать, что $xf'_x + yf_y + zf'_x = uF'_u + vF'_v + wF'_w$.

◄ Согласно условию, имеем

$$F(u, v, w) = f(\sqrt{vw}, \sqrt{uw}, \sqrt{uv}).$$

Дифференцируя это равенство по u, v и w, находим

$$F'_u = f'_y\frac{w}{2\sqrt{uv}} + f'_x\frac{v}{2\sqrt{uv}}, F'_v = f'_x\frac{w}{2\sqrt{vw}} + f'_z\frac{u}{2\sqrt{uv}}, F'_w = f'_x\frac{v}{2\sqrt{vw}} + f'_y\frac{u}{2\sqrt{uw}}. \quad (1)$$

Умножая первое из равенств (1) на u, второе на v, а третье на w и складывая их, получаем

$$uF'_u + vF'_v + wF'_w = f'_y\frac{uw}{2\sqrt{uw}} + f'_z\frac{uv}{2\sqrt{uv}} + f'_x\frac{vw}{2\sqrt{vw}} + f'_z\frac{uv}{2\sqrt{uv}} + f'_x\frac{vw}{2\sqrt{vw}} + f'_y\frac{uw}{2\sqrt{uw}} = \sqrt{vw}f'_x + \sqrt{uw}f'_y + \sqrt{uv}f'_z.$$

Отсюда, используя условие задачи, окончательно находим

$$uF'_u + vF'_v + wF'_w = xf'_x + yf'_y + zf'_z. \blacktriangleright$$

Путём последовательного дифференцирования исключить произвольные функции ϕ и ψ:

68. $z = x + \phi(xy)$.

◄ Найдем частные производные по x и по y:

$$\frac{\partial z}{\partial x} = 1 + y\phi', \frac{\partial z}{\partial y} = x\phi'.$$

Сложим полученные равенства, умножив первое из них на x, а второе на $-y$. Тогда получим

$$\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = x. \blacktriangleright$$

69. $u = \phi(x - y, y - z)$.

◄ Имеем $\frac{\partial u}{\partial x} = \phi'_1$, $\frac{\partial u}{\partial y} = -\phi'_1 + \phi'_2$, $\frac{\partial u}{\partial z} = -\phi'_2$. Складывая эти равенства, получаем $\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} + \frac{\partial u}{\partial z} = 0$. ►

70. $z = \phi(x)\psi(y)$.

◄ Имеем $\frac{\partial z}{\partial x} = \phi'\psi$, $\frac{\partial z}{\partial y} = \phi\psi'$. Отсюда $\frac{\partial z}{\partial x}\frac{\partial z}{\partial y} = \phi\psi\phi'\psi' = z\phi'\psi'$.

С другой стороны, $\frac{\partial^2 z}{\partial x\partial y} = \phi'\psi'$. Следовательно, из последних двух равенст непосредственно вытекает, что $\frac{\partial z}{\partial x}\frac{\partial z}{\partial y} = z\frac{\partial^2 z}{\partial x\partial y}$. ►

71. $z = \phi(xy) + \psi(\frac{x}{y})$.

◄ Используя равенства

$$\frac{\partial z}{\partial x} = y\phi' + \frac{1}{y}\psi', \frac{\partial^2 z}{\partial x^2} = y^2\phi'' + \frac{1}{y^2}\psi'', \frac{\partial z}{\partial y} = x\phi' - \frac{x}{y^2}\psi', \frac{\partial^2 z}{\partial y^2} = x^2\phi'' + \frac{x^2}{y^4}\psi'' + \frac{2x}{y^3}\psi',$$

получаем следующие соотношения:

$$x\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = \frac{2x}{y}\psi', x^2\frac{\partial^2 z}{\partial x^2} - y^2\frac{\partial^2 z}{\partial y^2} = -\frac{2x}{y}\psi',$$

из которых непосредственно вытекает, что

$$x^2\frac{\partial^2 z}{\partial x^2} - y^2\frac{\partial^2 z}{\partial y^2} + x\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = 0. \blacktriangleright$$

72. Найти производную функции $\mathbf{z = x^2 - y^2}$ в точке $\mathbf{M = (1, 1)}$ в направлении l, составляющем угол $\alpha = \mathbf{60^\circ}$ с положительным направлением оси Ox.

◄ Имеем $\frac{\partial z(M)}{\partial l} = \frac{\partial z(M)}{\partial x}cos\alpha + \frac{\partial z(M)}{\partial y}cos\beta = 2cos\alpha - 2cos\beta$. Таким образом, $\frac{\partial z(M)}{\partial l} = 1 - \sqrt{3}$. ►

73. Найти производную функции $z = ln(x^2 + y^2)$ в точке $M = (x_0, y_0)$ в направлении l, перпендикулярном к линии уровня, проходящей через эту точку.

◄ Поскольку вектор $grad\ u$ в точке M ортогонален к линии уровня $c = ln(x^2 + y^2)$, проходящей через точку M, то направляющие косинусы вектора l равны направляющим косинусам $gradu$ в точке M, т. е.

$$cos\alpha = \frac{\frac{\partial z(M)}{\partial x}}{||grad\ u(M)||}, \quad cos\beta = \frac{\frac{\partial z(M)}{\partial y}}{||grad\ u(M)||}.$$

Но $\frac{\partial z(M)}{\partial x} = \frac{2x_0}{x_0^2+y_0^2}$, $\frac{\partial z(M)}{\partial y} = \frac{2y_0}{x_0^2+y_0^2}$,

$$||grad\ u(M)|| = \sqrt{\left(\frac{\partial z(M)}{\partial x}\right)^2 + \left(\frac{\partial z(M)}{\partial y}\right)^2} = \frac{2}{\sqrt{x_0^2 + y_0^2}},$$

поэтому $cos\alpha = \frac{x_0}{\sqrt{x_0^2+y_0^2}}$, $cos\beta = \frac{y_0}{\sqrt{x_0^2+y_0^2}}$. Следовательно,

$$\frac{\partial z(M)}{\partial l} = \frac{\partial z(M)}{\partial x}cos\alpha + \frac{\partial z(M)}{\partial y}cos\beta = \frac{2}{\sqrt{x_0^2 + y_0^2}} \quad (x_0^2 + y_0^2 \neq 0). \blacktriangleright$$

74. Найти производную функции $z = 1 - (\frac{x^2}{a^2} + \frac{y^2}{b^2})$ в точке $M = (\frac{a}{\sqrt{2}}, \frac{b}{\sqrt{2}})$ по направлению внутренней нормали к этой точке к кривой $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$.

◄ Тангенс угла наклона нормали к данной кривой определяется формулой $tg\alpha = -\frac{1}{y'(\frac{a}{\sqrt{2}})}$, где $y = \frac{b}{a}\sqrt{a^2 - x^2}$. Отсюда $tg\alpha = \frac{a}{b}$, а направляющие косинусы внутренней нормали выражаются формулами $cos\alpha = -\frac{b}{\sqrt{a^2+b^2}}$, $cos\beta = -\frac{a}{\sqrt{a^2+b^2}}$(мы берем знак минус, поскольку нормаль внутренняя). Воспользуемся формулой производной по направлению $n = (cos\alpha; cos\beta)$:

$$\frac{\partial z(M)}{\partial n} = \frac{\partial z(M)}{\partial x}cos\alpha + \frac{\partial z(M)}{\partial y}cos\beta.$$

Вычисляя производные $\frac{\partial z(M)}{\partial x} = -\frac{\sqrt{2}}{a}$, $\frac{\partial z(M)}{\partial y} = -\frac{\sqrt{2}}{b}$, находим

$$\frac{\partial z(M)}{\partial n} = \frac{b\sqrt{2}}{a\sqrt{a^2 + b^2}} + \frac{a\sqrt{2}}{b\sqrt{a^2 + b^2}} = \frac{\sqrt{2(a^2 + b^2)}}{ab}. \blacktriangleright$$

75. Найти производную функции $u = xyz$ в точке $M = (1, 1, 1)$ в направлении $l = (cos\alpha, cos\beta, cos\gamma)$. Чему равна величина градиента функции в этой точке?

◄ Очевидно, $\frac{\partial u(M)}{\partial x} = 1$, $\frac{\partial u(M)}{\partial y} = 1$, $\frac{\partial u(M)}{\partial z} = 1$. По формуле производной по направлению, получим

$$\frac{\partial u(M)}{\partial l} = \frac{\partial u(M)}{\partial x}cos\alpha + \frac{\partial u(M)}{\partial y}cos\beta + \frac{\partial u(M)}{\partial z}cos\gamma = cos\alpha + cos\beta + cos\gamma.$$

Величину градиента определим по формуле

$$||grad\ u(M)|| = \sqrt{\left(\frac{\partial u(M)}{\partial x}\right)^2 + \left(\frac{\partial u(M)}{\partial y}\right)^2 + \left(\frac{\partial u(M)}{\partial z}\right)^2} = \sqrt{3}. \blacktriangleright$$

76. Определить угол между градиентами функции $\mathbf{u = x^2 + y^2 + z^2}$ в точках $\mathbf{A} = (\varepsilon, \mathbf{0}, \mathbf{0})$ и $\mathbf{B} = (\mathbf{0}, \varepsilon, \mathbf{0})$.

и полагая $t = 1$, получаем требуемое равенство. ▶

67. Пусть $x^2 = vw$, $y^2 = uw$, $z^2 = uv$ и $f(x, y, z) = F(u, v, w)$. Доказать, что $xf'_x + yf'_y + zf'_z = uF'_u + vF'_v + wF'_w$.

◀ Согласно условию, имеем

$$F(u, v, w) = f(\sqrt{vw}, \sqrt{uw}, \sqrt{uv}).$$

Дифференцируя это равенство по u, v и w, находим

$$F'_u = f'_y\frac{w}{2\sqrt{uw}} + f'_z\frac{v}{2\sqrt{uv}}, \quad F'_v = f'_x\frac{w}{2\sqrt{vw}} + f'_z\frac{u}{2\sqrt{uv}}, \quad F'_w = f'_x\frac{v}{2\sqrt{vw}} + f'_y\frac{u}{2\sqrt{uw}}. \tag{1}$$

Умножая первое из равенств (1) на u, второе на v, а третье на w и складывая их, получаем

$$uF'_u + vF'_v + wF'_w = f'_y\frac{uw}{2\sqrt{uw}} + f'_z\frac{uv}{2\sqrt{uv}} + f'_x\frac{vw}{2\sqrt{vw}} + f'_z\frac{uv}{2\sqrt{uv}} + \\ + f'_x\frac{vw}{2\sqrt{vw}} + f'_y\frac{uw}{2\sqrt{uw}} = \sqrt{vw}f'_x + \sqrt{uw}f'_y + \sqrt{uv}f'_z.$$

Отсюда, используя условие задачи, окончательно находим

$$uF'_u + vF'_v + wF'_w = xf'_x + yf'_y + zf'_z. ▶$$

Путем последовательного дифференцирования исключить произвольные функции φ и ψ:

68. $z = x + \varphi(xy)$.

◀ Найдем частные производные по x и по y:

$$\frac{\partial z}{\partial x} = 1 + y\varphi', \quad \frac{\partial z}{\partial y} = x\varphi'.$$

Сложим полученные равенства, умножив первое из них на x, а второе на $-y$. Тогда получим

$$x\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = x. ▶$$

69. $u = \varphi(x - y, y - z)$.

◀ Имеем $\frac{\partial u}{\partial x} = \varphi'_1$, $\frac{\partial u}{\partial y} = -\varphi'_1 + \varphi'_2$, $\frac{\partial u}{\partial z} = -\varphi'_2$. Складывая эти равенства, получаем $\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} + \frac{\partial u}{\partial z} = 0$. ▶

70. $z = \varphi(x)\psi(y)$.

◀ Имеем $\frac{\partial z}{\partial x} = \varphi'\psi$, $\frac{\partial z}{\partial y} = \varphi\psi'$. Отсюда $\frac{\partial z}{\partial x}\frac{\partial z}{\partial y} = \varphi\psi\varphi'\psi' = z\varphi'\psi'$.

С другой стороны, $\frac{\partial^2 z}{\partial x\,\partial y} = \varphi'\psi'$. Следовательно, из последних двух равенств непосредственно вытекает, что $\frac{\partial z}{\partial x}\frac{\partial z}{\partial y} = z\frac{\partial^2 z}{\partial x\,\partial y}$. ▶

71. $z = \varphi(xy) + \psi\left(\frac{x}{y}\right)$.

◀ Используя равенства

$$\frac{\partial z}{\partial x} = y\varphi' + \frac{1}{y}\psi', \quad \frac{\partial^2 z}{\partial x^2} = y^2\varphi'' + \frac{1}{y^2}\psi'', \quad \frac{\partial z}{\partial y} = x\varphi' - \frac{x}{y^2}\psi', \quad \frac{\partial^2 z}{\partial y^2} = x^2\varphi'' + \frac{x^2}{y^4}\psi'' + \frac{2x}{y^3}\psi',$$

получаем следующие соотношения:

$$x\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = \frac{2x}{y}\psi', \quad x^2\frac{\partial^2 z}{\partial x^2} - y^2\frac{\partial^2 z}{\partial y^2} = -\frac{2x}{y}\psi',$$

из которых непосредственно вытекает, что

$$x^2\frac{\partial^2 z}{\partial x^2} - y^2\frac{\partial^2 z}{\partial y^2} + x\frac{\partial z}{\partial x} - y\frac{\partial z}{\partial y} = 0. ▶$$

72. Найти производную функции $z = x^2 - y^2$ в точке $M = (1, 1)$ в направлении l, составляющем угол $\alpha = 60°$ с положительным направлением оси Ox.

◀ Имеем $\frac{\partial z(M)}{\partial l} = \frac{\partial z(M)}{\partial x}\cos\alpha + \frac{\partial z(M)}{\partial y}\cos\beta = 2\cos\alpha - 2\cos\beta$. Таким образом, $\frac{\partial z(M)}{\partial l} = 1-\sqrt{3}$. ▶

73. Найти производную фуикции $z = \ln(x^2+y^2)$ в точке $M = (x_0, y_0)$ в направлении l, перпендикулярном к линии уровня, проходящей через эту точку.

◀ Поскольку вектор $\operatorname{grad} u$ в точке M ортогонален к линии уровня $c = \ln(x^2+y^2)$, проходящей через точку M, то направляющие косинусы вектора l равны направляющим косинусам $\operatorname{grad} u$ в точке M, т. е.

$$\cos\alpha = \frac{\frac{\partial z(M)}{\partial x}}{\|\operatorname{grad} u(M)\|}, \quad \cos\beta = \frac{\frac{\partial z(M)}{\partial y}}{\|\operatorname{grad} u(M)\|}.$$

Но $\frac{\partial z(M)}{\partial x} = \frac{2x_0}{x_0^2+y_0^2}$, $\frac{\partial z(M)}{\partial y} = \frac{2y_0}{x_0^2+y_0^2}$,

$$\|\operatorname{grad} u(M)\| = \sqrt{\left(\frac{\partial z(M)}{\partial x}\right)^2 + \left(\frac{\partial z(M)}{\partial y}\right)^2} = \frac{2}{\sqrt{x_0^2+y_0^2}},$$

поэтому $\cos\alpha = \frac{x_0}{\sqrt{x_0^2+y_0^2}}$, $\cos\beta = \frac{y_0}{\sqrt{x_0^2+y_0^2}}$. Следовательно,

$$\frac{\partial z(M)}{\partial l} = \frac{\partial z(M)}{\partial x}\cos\alpha + \frac{\partial z(M)}{\partial y}\cos\beta = \frac{2}{\sqrt{x_0^2+y_0^2}} \quad (x_0^2+y_0^2 \neq 0). \blacktriangleright$$

74. Найти производную функции $z = 1 - \left(\frac{x^2}{a^2} + \frac{y^2}{b^2}\right)$ в точке $M = \left(\frac{a}{\sqrt{2}}, \frac{b}{\sqrt{2}}\right)$ по направлению внутренней нормали в этой точке к кривой $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$.

◀ Тангенс угла наклона нормали к данной кривой определяется формулой $\operatorname{tg}\alpha = -\frac{1}{y'\left(\frac{a}{\sqrt{2}}\right)}$, где $y = \frac{b}{a}\sqrt{a^2-x^2}$. Отсюда $\operatorname{tg}\alpha = \frac{a}{b}$, а направляющие косинусы внутренней нормали выражаются формулами $\cos\alpha = -\frac{b}{\sqrt{a^2+b^2}}$, $\cos\beta = -\frac{a}{\sqrt{a^2+b^2}}$ (мы берем знак минус, поскольку нормаль внутренняя). Воспользуемся формулой производной по направлению $n = (\cos\alpha, \cos\beta)$:

$$\frac{\partial z(M)}{\partial n} = \frac{\partial z(M)}{\partial x}\cos\alpha + \frac{\partial z(M)}{\partial y}\cos\beta.$$

Вычисляя производные $\frac{\partial z(M)}{\partial x} = -\frac{\sqrt{2}}{a}$, $\frac{\partial z(M)}{\partial y} = -\frac{\sqrt{2}}{b}$, находим

$$\frac{\partial z(M)}{\partial n} = \frac{b\sqrt{2}}{a\sqrt{a^2+b^2}} + \frac{a\sqrt{2}}{b\sqrt{a^2+b^2}} = \frac{\sqrt{2(a^2+b^2)}}{ab}. \blacktriangleright$$

75. Найти производную функции $u = xyz$ в точке $M = (1, 1, 1)$ в направлении $l = (\cos\alpha, \cos\beta, \cos\gamma)$. Чему равна величина градиента функции в этой точке?

◀ Очевидно, $\frac{\partial u(M)}{\partial x} = 1$, $\frac{\partial u(M)}{\partial y} = 1$, $\frac{\partial u(M)}{\partial z} = 1$. По формуле производной по направлению, получим

$$\frac{\partial u(M)}{\partial l} = \frac{\partial u(M)}{\partial x}\cos\alpha + \frac{\partial u(M)}{\partial y}\cos\beta + \frac{\partial u(M)}{\partial z}\cos\gamma = \cos\alpha + \cos\beta + \cos\gamma.$$

Величину градиента определим по формуле

$$\|\operatorname{grad} u(M)\| = \sqrt{\left(\frac{\partial u(M)}{\partial x}\right)^2 + \left(\frac{\partial u(M)}{\partial y}\right)^2 + \left(\frac{\partial u(M)}{\partial z}\right)^2} = \sqrt{3}. \blacktriangleright$$

76. Определить угол между градиентами функции $u = x^2+y^2+z^2$ в точках $A = (\varepsilon, 0, 0)$ и $B = (0, \varepsilon, 0)$.

9. 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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы: -

11. Выводы:

Я научился верстать на TeX.

Подпись студента

_____

Отчет по лабораторной работе № 23 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 23

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: « » _____2022г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема: Динамические структуры. Обработка деревьев**
2. **Цель работы:** Составить программу на языке си для обработки деревьев.
3. **Задание Посчитать количество вершин двоичного дерева, тип данных - float:**
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Windows, наименование Windows 11 pro
Прикладные системы и программы: виртуальная машина VMware.

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)
tree.c - содержит все функции, требуемые в задании
tree.h - заголовочный файл с описанием структуры
Main.c - меню для работы с деревом

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

1. Создать заголовочный файл с описанием структуры и функций
2. Написать функции для преобразований
3. Написать основной файл с меню, продумать реализацию вывода

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

Тесты.

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 1

Введите значение узла, которое вы хотите добавить в дерево: 5.333

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 1

Введите значение узла, которое вы хотите добавить в дерево: 4.321

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 4

Количество вершин равно: 2

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 1

Введите значение узла, которое вы хотите добавить в дерево: 10.01

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 2

_____

10.010000

5.333000

4.321000

---

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 1

Введите значение узла, которое вы хотите добавить в дерево: 34.001

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 2

_____

34.000999

10.010000

5.333000

4.321000

---

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 3

Введите значение узла, который вы хотите удалить из дерева: 10.01

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 4

Количество вершин равно: 3

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 2

_____

34.000999

5.333000

4.321000

---

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 3

Введите значение узла, который вы хотите удалить из дерева: 4.321

Готово

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 2

_____

34.000999

5.333000

---

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 4

Количество вершин равно: 2

1) Добавить узел в дерево

2) Напечатать дерево

3) Удалить узел

4) Вычислить количество вершин

Введите 0, чтобы выйти

Номер операции: 0

Содержимое файлов

Main.c

```
#include <stdio.h>
#include "tree.h"


void print_menu() {
printf("1) Добавить узел в дерево\n2) Напечатать дерево\n3) Удалить узел\n4) Вычислить количество вершин\nВведите 0, чтобы выйти\nНомер операции: ");
}
```

```
int main(void)
{
Tree *t = NULL;
float value;
char c;
print_menu();
while ((c = getchar()) != EOF) {
value = 0;
if (c == '\n') continue;
switch (c) {
        case '1':
           printf("\nВведите значение узла, которое вы хотите добавить в дерево:\t");
           scanf("%f", &value);
           t = tree_add_element(t, value);
           break;
        case '2':
           printf("\n_____\n");
           tree_print(t, 1);
           printf("\n---------------------------------------\n");
           break;
        case '3':
           printf("\nВведите значение узла, который вы хотите удалить из дерева:\t");
           scanf("%f", &value);
           t = delete_element(t, value);
           break;
        case '4':
           tree_val_count(t);
           break;
        case '0':
           return 0;
```

```
        default:
printf("Такой команды нет в меню\n");
break;
}
print_menu();
}
}
```

Tree.c

```
#include <stdio.h>
#include <stdlib.h>
#include "tree.h"



Tree *root_create(float value)
{
   Tree *tree = (Tree*)malloc(sizeof(Tree));
   tree->data = value;
   tree->left = NULL;
   tree->right = NULL;
   tree->parent = NULL;
   return tree;
}



Tree *tree_add_element(Tree *root, float value)
{
   if (root == NULL) {
      printf("Готово\n\n");
      return root_create(value);
   }
```

```
Tree *NewTree = (Tree*)malloc(sizeof(Tree));
NewTree->data = value;
Tree *tree1 = root;
Tree *tree2 = NULL;

while (tree1 != NULL) {
   tree2 = tree1;
   if (value < tree1->data) {
      tree1 = tree1->left;
   }
   else if (value > tree1->data) {
      tree1 = tree1->right;
   }
   else {
      printf("Это значение уже есть в дереве\n\n");
      return root;
   }
}
NewTree->parent = tree2;
NewTree->left = NULL;
NewTree->right = NULL;
if (value < tree2->data) {
   tree2->left = NewTree;
} else {
   tree2->right = NewTree;
}
printf("Готово\n\n");
return root;
```

```
void tree_print(Tree *root, int n)
{
   if (root != NULL) {
      tree_print(root->right, n + 1);
      for (int i = 0; i < n; i++) printf("\t");
      printf("%f\n", root->data);
      tree_print(root->left, n + 1);
   }
}


void tree_val_count(Tree *root) {
   int n = 0;
   tree_val_counter(root, &n);
   printf("\nКоличество вершин равно:\t%d\n\n", n);
}


void tree_val_counter(Tree *root, int *n)
{
   if (root != NULL) {
      *n += 1;
      tree_val_counter(root->right, n);
      tree_val_counter(root->left, n);
   }
}


Tree *delete_element(Tree *root, float value)
{
```

```
Tree* tree1 = NULL, *tree2 = NULL, *tree3 = root, *tree4 = NULL;


if (root == NULL) {
   printf("Дерево не существует\n\n");
   return root;
}
tree1 = search_in_tree(tree3, value);
if (tree1 == NULL) {
   printf("Заданного элемента нет\n\n");
   return root;
}
 // Элемент является листом
if (tree1->left == NULL && tree1->right == NULL) {
   if (tree1->parent == NULL) { // если корень
      free(tree1);
      tree1 = NULL;
      printf("Готово\n\n");
      return NULL;
   }
   tree2 = tree1->parent;
   if (tree2->left == tree1) {
      tree2->left = NULL;
   }
   else {
      tree2->right = NULL;
   }
   free(tree1);
}
 // Одно поддерево доступно
else if (tree1->left != NULL && tree1->right == NULL) { // левое существует
```

```
    if (tree1->parent == NULL) { // если корень
        tree4 = tree1->left;
        tree4->parent = NULL;
        free(tree1);
        printf("Готово\n\n");
        return tree4;
    }
    tree2 = tree1->parent;
    if (tree2->left == tree1) {
        tree2->left = tree1->left;
    }
    else {
        tree2->right = tree1->left;
    }
    free(tree1);
}
else if (tree1->left == NULL && tree1->right != NULL) { // правое существует
    if (tree1->parent == NULL) { // если корень
        tree4 = tree1->right;
        tree4->parent = NULL;
        free(tree1);
        printf("Готово\n\n");
        return tree4;
    }
    tree2 = tree1->parent;
    if (tree2->left == tree1) {
        tree2->left = tree1->right;
    }
    else {
        tree2->right = tree1->right;
```

```
        }
        free(tree1);
    }
    // Оба поддерева доступны
    else if (tree1->left != NULL && tree1->right != NULL) {
        tree2 = minimum(tree1->right);
        tree1->data = tree2->data;
        tree4 = tree2->parent;
        if (tree4->left == tree2) {
            free(tree2);
            tree4->left = NULL;
        }
        if (tree4->right == tree2) {
            free(tree2);
            tree4->right = NULL;
        }
    }
    printf("Готово\n\n");
    return root;
}


Tree *search_in_tree(Tree *root, float value)
{
    if (root == NULL) {
        return NULL;
    }
    if (root->data == value) {
        return root;
    }
```

```
    if (value > root->data) {
        return search_in_tree(root->right,value);
    } else {
        return search_in_tree(root->left, value);
    }
}


Tree *minimum(Tree *t)
{
    if (t->left == NULL) {
        return t;
    }
    return minimum(t->left);
}
```

tree.h

```
#ifndef _TREE_H_
#define _TREE_H_
#include <stdio.h>
#include <stdlib.h>


typedef struct Tree{
    float data;
    struct Tree *left;
    struct Tree *right;
    struct Tree *parent;
} Tree;
```

```
Tree *root_create(float value);
Tree *tree_add_element(Tree *parent, float value);
Tree *delete_element(Tree *parent, float value);
void tree_print(Tree *t, int n);
void tree_val_count(Tree *t);
void tree_val_counter(Tree *t, int *n);
Tree* minimum(Tree *t);
Tree* search_in_tree(Tree *t, float value);


#endif
```

9. 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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы: -

11. Выводы:

Я научился работать с деревьяви.

Подпись студента

_____

Отчет по лабораторной работе № 24 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 4

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: « » _____2022г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Преобразование выражения в дерево
2. **Цель работы:** Научиться преобразовывать выражения в дерево и работать с ним.
3. **Задание Упростить выражение, выполнив деление: 4*a/2->2*a:**
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Windows, наименование Windows 11 pro
Прикладные системы и программы: виртуальная машина VMware.

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)
tree.c - содержит все функции, требуемые в задании
tree.h - заголовочный файл с описанием структуры
Main.c - меню для работы с дэком

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

1. Создать заголовочный файл с описанием структуры и функций
2. Написать функции для преобразований
3. Написать основной файл с меню, продумать реализацию ввода

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

Тесты.

```
nikita@ubuntu:~/Documents$ ./prog
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: 4*a/2
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
2*a
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 4
Дерево исходного выражения
   2
/
      a
   *
      4
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения
   a
*
```

```
   2
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: (15/3)+(20/2)-(55/11)
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
(5+10)-5
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 4
Дерево исходного выражения
      11
   /
      55
-
         2
      /
         20
   +
         3
      /
         15
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения
   5
```

```
-
      10
   +
      5
```
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: 10*(4*a*2/2)
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 4
Дерево исходного выражения
```
      2
   /
         2
      *
            a
         *
            4
*
   10
```
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
10*((2*a)*2)
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения

```
      2
   *
         a
      *
         2
*
   10
```
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: ((8*a)/(6*3*c))*4*3
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
(a/((3*3)*c))*3
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 4
Дерево исходного выражения
```
      3
   *
         4
      *
               c
            *
                  3
               *
                  6
         /
               a
            *
               8
```
Меню:
1) Ввести выражение

2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения
```
   3
*
        c
     *
          3
        *
          3
  /
     a
```
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: (15*7)/(21*75)
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
1/15
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 4
Дерево исходного выражения
```
     75
  *
     21
/
     7
  *
```

```
      15
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения
   15
/
   1
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 1
Введите выражение: 4*b+7-c
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 3
((4*b)+7)-c
Меню:
1) Ввести выражение
2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 5
Дерево преобразованного выражения
   c
-
      7
   +
         b
      *
         4
Меню:
1) Ввести выражение
```

2) Печать исходного выражения
3) Печать преобразованного выражения
4) Печать исходного дерева
5) Печать преобразованного дерева
6) Выход
Выберите действие: 6
nikita@ubuntu:~/Documents$

Содержимое файлов

Main.c

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include "tree.h"



int main(void)
{
int action;
char expr[255];
Node *root = NULL, *root2 = NULL;
Stack stPost;

while (1)
{
printf("Меню:\n1) Ввести выражение\n2) Печать исходного выражения\n3) Печать
преобразованного выражения\n4) Печать исходного дерева\n5) Печать преобразованного
дерева\n6) Выход\nВыберите действие: ");
scanf("%d", &action);

switch (action)
{
case 1:
{
printf("Введите выражение: ");
scanf("%s", expr);

treeDestroy(&root);
treeDestroy(&root2);
stackCreate(&stPost);
postOrder(expr, &stPost);
treeBuild(&root, &stPost);
stackDestroy(&stPost);

root2 = treeCopy(&root);

Node *prev = NULL;
```

```
Node *node = root2;
int *firstNumSide = NULL;
firstNumSide = malloc(sizeof(int));
int *secondNumSide = NULL;
secondNumSide = malloc(sizeof(int));
Node *startPos = root2;
Node *firstNumNode = NULL;
Node *secondNumNode = NULL;
int nod;
Node *divNode = NULL;
while(1) {
firstNumNode = findDiv(&node,&startPos,&divNode,firstNumSide,secondNumSide);

*secondNumSide = NONE;

if(firstNumNode != NULL) {

secondNumNode = findAnother(&startPos,&firstNumNode,firstNumSide,secondNumSide);

if(secondNumNode != NULL) {

nod = NOD(firstNumNode->_num,secondNumNode->_num);
firstNumNode->_num = firstNumNode->_num / nod;

if (firstNumNode->_num == 1) {

if(*firstNumSide != LEFT) {

free(firstNumNode);
firstNumNode = NULL;

} else if(deleteOnesInNumerator(&(divNode->_left), &firstNumNode)){
free(firstNumNode);
firstNumNode = NULL;
}
}

secondNumNode->_num = secondNumNode->_num / nod;
if (secondNumNode->_num == 1) {

if(*secondNumSide != LEFT) {

free(secondNumNode);
secondNumNode = NULL;

} else if(deleteOnesInNumerator(&(divNode->_left), &secondNumNode)) {

free(secondNumNode);
secondNumNode = NULL;
}
}
```

```
} else break;

} else break;

node = root2;
deleteEmptyNodes(&node);
while(1) {
if(deleteVarOp(&root2,&node,&prev) == 0)
break;
node = root2;
}

prev = NULL;
node = root2;
startPos = root2;
*firstNumSide = NONE;
*secondNumSide = NONE;
}

break;
}

case 2:
{
printf("Исходное выражение: %s\n", expr);

break;
}

case 3:
{
LKP(&root2);
printf("\n");

break;
}

case 4:
{
if (root != NULL) {

printf("Дерево исходного выражения\n");
PKL(&root, 0);
}
else
printf("Дерево исходного выражения пусто\n");

break;
}

case 5:
{
```

```
if (root2 != NULL) {

printf("Дерево преобразованного выражения\n");
PKL(&root2, 0);
}
else
printf("Дерево преобразованного выражения пусто\n");

break;
}

case 6: break;

default:
{
printf("Ошибка. Такого пункта меню не существует\n");

break;
}
}

if (action == 6)
break;
}

treeDestroy(&root);
treeDestroy(&root2);

return 0;
}
```

tree.c

```
#include "tree.h"

void stackCreate(Stack *s)
{
s->_size = 0;
s->_top = NULL;
}

int stackEmpty(const Stack *s)
{
return s->_top == NULL;
}

int stackSize(const Stack *s)
{
return s->_size;
}
```

```
int stackPush(Stack *s, const STACK_TYPE value)
{
ItemStack *item = (ItemStack *)malloc(sizeof(ItemStack));

if (!item)
return 0;

item->_data = value;
item->_prev = s->_top;
s->_top = item;
s->_size++;

return 1;
}

int stackPop(Stack *s)
{
ItemStack *item = NULL;

if (!s->_size)
return 0;

item = s->_top;
s->_top = s->_top->_prev;
s->_size--;

free(item);

return 1;
}

STACK_TYPE stackTop(const Stack *s)
{
return s->_top->_data;
}

void stackDestroy(Stack *s)
{
ItemStack *item = NULL;

while (s->_top)
{
item = s->_top;
s->_top = s->_top->_prev;

free(item);
}

s->_size = 0;
s->_top = NULL;
}
```

```
int NOD(int a, int b) {
while(a != b) {

if (a > b)
a -= b;
else
b -= a;
}
return a;
}

Node *treeNodeCreate() {

Node *tmpNode = (Node *)malloc(sizeof(Node));

tmpNode->_varOp = '\0';
tmpNode->_num = 0.0;
tmpNode->_left = NULL;
tmpNode->_right = NULL;

return tmpNode;
}

Node *treeCopy(Node **node) {

Node *tmpNode = NULL;

if (*node == NULL)
return NULL;

tmpNode = treeNodeCreate();
tmpNode->_varOp = (*node)->_varOp;
tmpNode->_num = (*node)->_num;
tmpNode->_left = treeCopy(&(*node)->_left);
tmpNode->_right = treeCopy(&(*node)->_right);

return tmpNode;
}

int treeIsMinusNode(Node **node) {

if (*node == NULL)
return 0;

if ((*node)->_left == NULL || (*node)->_right == NULL)
return 0;

return ((*node)->_varOp == '-' && (*node)->_left->_varOp == '\0' && (*node)->_left->_num == 0.0);
}

void treeBuild(Node **node, Stack *st) {

Token token;
```

```
if (stackEmpty(st))
return;

token = stackTop(st);

stackPop(st);

(*node) = treeNodeCreate();
(*node)->_varOp = token._varOp;
(*node)->_num = token._num;

if (isOp((*node)->_varOp)) {

treeBuild(&(*node)->_right, st);
treeBuild(&(*node)->_left, st);
}
}

void treeDestroy(Node **node) {

if (*node == NULL)
return;

if ((*node)->_left != NULL)
treeDestroy(&(*node)->_left);

if ((*node)->_right != NULL)
treeDestroy(&(*node)->_right);

free(*node);

*node = NULL;
}

Node *findDiv(Node **node, Node **startPos, Node **divNode, int *firstNumSide, int
*anotherNumSide) {

if(*node == NULL) return NULL;

if ((*node)->_num != 0 && (*node)->_num != 1 && (*firstNumSide == LEFT || *firstNumSide ==
RIGHT)) {

if(findAnother(&(*startPos), &(*node), firstNumSide, anotherNumSide) != NULL)
return *node;
else
return NULL;
}

if ((*node)->_varOp == '+' || (*node)->_varOp == '-') {

if(*firstNumSide != NONE)
return NULL;
```

```
*startPos = NULL;

} else if(((*node)->_varOp == '*' || (*node)->_varOp == '/') && *startPos == NULL && *firstNumSide
== NONE) {

*startPos = *node;
}

if((*node)->_varOp == '/') {

*divNode = *node;

*firstNumSide = LEFT;

Node *tmpNode;

tmpNode = findDiv(&(*node)->_left, &(*startPos), &(*divNode), firstNumSide, anotherNumSide);
if(tmpNode != NULL) return tmpNode;

*firstNumSide = RIGHT;

tmpNode = findDiv(&(*node)->_right, &(*startPos), &(*divNode), firstNumSide, anotherNumSide);
if(tmpNode != NULL) return tmpNode;

} else {

Node *tmpNode;

tmpNode = findDiv(&(*node)->_left, &(*startPos), &(*divNode), firstNumSide, anotherNumSide);
if(tmpNode != NULL) return tmpNode;

tmpNode = findDiv(&(*node)->_right, &(*startPos), &(*divNode), firstNumSide, anotherNumSide);
if(tmpNode != NULL) return tmpNode;
}

return NULL;
}
int deleteOnesInNumerator(Node **node, Node **divNumNode) {

if(*node == NULL) return 0;

if (((*node)->_num != 0 && ((*node) != (*divNumNode))) || isLetter((*node)->_varOp))
return 1;

if ((*node)->_varOp == '+' || (*node)->_varOp == '-')
return 0;

if(deleteOnesInNumerator(&(*node)->_left,&(*divNumNode)))
return 1;

if(deleteOnesInNumerator(&(*node)->_right,&(*divNumNode)))
return 1;

return 0;
```

```
}
Node *findAnother(Node **node, Node **firstNumNode, int *firstNumSide, int *anotherNumSide) {

if(*node == NULL) return NULL;

if ((*node)->_num != 0 && (*node)->_num != 1 && ((*node) != (*firstNumNode)) &&
NOD((*firstNumNode)->_num, (*node)->_num) != 1) {

if(*firstNumSide == LEFT && *anotherNumSide == RIGHT)
return *node;
else if(*firstNumSide == RIGHT && *anotherNumSide != RIGHT)
return *node;
else
return NULL;

}

if ((*node)->_varOp == '+' || (*node)->_varOp == '-')
return NULL;

if((*node)->_varOp == '/') {

Node *tmpNode;

*anotherNumSide = LEFT;

tmpNode = findAnother(&(*node)->_left,&(*firstNumNode),firstNumSide,anotherNumSide);
if(tmpNode != NULL) return tmpNode;

*anotherNumSide = RIGHT;

tmpNode = findAnother(&(*node)->_right,&(*firstNumNode),firstNumSide,anotherNumSide);
if(tmpNode != NULL) return tmpNode;

} else {

Node *tmpNode;

tmpNode = findAnother(&(*node)->_left,&(*firstNumNode),firstNumSide,anotherNumSide);
if(tmpNode != NULL) return tmpNode;

tmpNode = findAnother(&(*node)->_right,&(*firstNumNode),firstNumSide,anotherNumSide);
if(tmpNode != NULL) return tmpNode;
}

return NULL;
}

void deleteEmptyNodes(Node **node) {

if(*node == NULL) return;

if (!(isOp((*node)->_varOp) || isLetter((*node)->_varOp)) && (*node)->_num == 0) {
```

```
*node = NULL;
return;
}

deleteEmptyNodes(&(*node)->_left);
deleteEmptyNodes(&(*node)->_right);

return;

}

int deleteVarOp(Node **root, Node **node, Node **prev) {

if ((*node)->_varOp == '\0' && (*node)->_num == 0) {

free(*node);
*node = NULL;
return 1;
}

if((*node)->_varOp != '\0' && !isLetter((*node)->_varOp)) {

if((*node)->_left == NULL && (*node)->_right == NULL) {

free(*node);
*node = NULL;
return 1;

} else if((*node)->_right == NULL) {

if(*prev == NULL) {

Node *tmpNode = *node;
*node = (*node)->_left;
*root = *node;
free(tmpNode);
tmpNode = NULL;
return 1;

} else {

if((*prev)->_right == *node) {

Node *tmpNode = *node;
(*prev)->_right = (*node)->_left;
free(tmpNode);
return 1;

} else {

Node *tmpNode = *node;
(*prev)->_left = (*node)->_left;
free(tmpNode);
```

```
return 1;
}
}
} else if((*node)->_left == NULL) {

if(*prev == NULL) {

Node *tmpNode = *node;
*node = (*node)->_right;
*root = *node;
free(tmpNode);
tmpNode = NULL;
return 1;

} else {

if((*prev)->_right == *node) {

Node *tmpNode = *node;
(*prev)->_right = (*node)->_right;
free(tmpNode);
return 1;

} else {

Node *tmpNode = *node;
(*prev)->_left = (*node)->_right;
free(tmpNode);
return 1;
}
}
}
}
if ((*node)->_left != NULL) {

if(deleteVarOp(&(*root),&(*node)->_left,&(*node))) {

return 1;

}
}
if ((*node)->_right != NULL) {

if(deleteVarOp(&(*root),&(*node)->_right,&(*node))){

return 1;

}
}
return 0;
}
```

```
void PKL(Node **node, const int level) {

if (*node == NULL)
return;

if ((*node)->_right != NULL)
PKL(&(*node)->_right, level + 1);

if ((*node)->_varOp != '\0')
printf("%*s%c\n", level * 4, "", (*node)->_varOp);
else
printf("%*s%d\n", level * 4, "", (*node)->_num);

if ((*node)->_left != NULL)
PKL(&(*node)->_left, level + 1);
}

void LKP(Node **node) {

if (*node == NULL)
return;

if ((*node)->_left != NULL && !treeIsMinusNode(node)) {

if ((*node)->_left->_left != NULL)
printf("(");

LKP(&(*node)->_left);

if ((*node)->_left->_left != NULL)
printf(")");
}

if ((*node)->_varOp != '\0')
printf("%c", (*node)->_varOp);
else
printf("%d", (*node)->_num);

if ((*node)->_right != NULL) {

if ((*node)->_right->_left != NULL)
printf("(");

LKP(&(*node)->_right);

if ((*node)->_right->_left != NULL)
printf(")");
}
}

int isLetter(const char ch) {
```

```
return ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'));
}

int isNumber(const char ch) {

return (ch >= '0' && ch <= '9');
}

int isOp(const char ch) {

return (ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '^' || ch == '(' || ch == ')');
}

int opPrior(const char op) {

if (op == '^')
return 4;

if (op == '*' || op == '/')
return 3;

if (op == '+' || op == '-')
return 2;

return 1;
}

int isOpHigh(const char op1, const char op2) {

if (op1 == '(' || op2 == '(' || op2 == ')')
return 0;

if (op1 == op2 && op2 == '^')
return 0;

return (opPrior(op1) >= opPrior(op2));
}

void postOrder(const char *str, Stack *st) {

int i = 0, step = -1, isBracket = 0, isDot = 0;
char tmpCh;
Token tk;
Stack stOp;

stackCreate(&stOp);

tk._varOp = '\0';
tk._num = 0.0;

while (str[i] != '\0') {

if (str[i] == '.')
```

```
isDot = 1;
else if (isLetter(str[i])) {

tk._varOp = str[i];

stackPush(st, tk);
}
else if (isNumber(str[i])) {

tk._varOp = '\0';

if (!isDot)
tk._num = tk._num * 10.0 + str[i] - '0';
else {

tk._num = tk._num + pow(10.0, step) * (str[i] - '0');
step--;
}

if (str[i + 1] != '.' && !isNumber(str[i + 1])) {

stackPush(st, tk);

tk._num = 0.0;
step = -1;
isDot = 0;
}
}
else if (isOp(str[i])) {

tk._varOp = str[i];

if (str[i] == ')')
isBracket = 1;
else if (str[i] == '-' && (i == 0 || str[i - 1] == '(')) {

tmpCh = tk._varOp;
tk._varOp = '\0';
tk._num = 0.0;

stackPush(st, tk);

tk._varOp = tmpCh;
}

while (!stackEmpty(&stOp) && (isOpHigh(stackTop(&stOp)._varOp, str[i]) || isBracket)) {

if (stackTop(&stOp)._varOp == '(')
isBracket = 0;
else
stackPush(st, stackTop(&stOp));

stackPop(&stOp);
```

```
}

if (str[i] != ')')
stackPush(&stOp, tk);
}

i++;
}

while (!stackEmpty(&stOp)) {

stackPush(st, stackTop(&stOp));
stackPop(&stOp);
}

stackDestroy(&stOp);
}
```

tree.h

```
#ifndef TREE_H
#define TREE_H

#include <stdlib.h>

typedef struct _Token
{
char _varOp;
double _num;
} Token;

typedef Token STACK_TYPE;

typedef struct _ItemStack
{
STACK_TYPE _data;
struct _ItemStack *_prev;
} ItemStack;

typedef struct _Stack
{
int _size;
struct _ItemStack *_top;
} Stack;

typedef enum _side {

NONE,
LEFT,
RIGHT
} side;
```

```
typedef struct _Node {

char _varOp;
int _num;
struct _Node *_left;
struct _Node *_right;

} Node;

// Функции непосредственно для задания:
int NOD(int a, int b);
Node *findDiv(Node **node, Node **startPos, Node **divNode, int *firstNumSide, int
*anotherNumSide);
Node *findAnother(Node **node, Node **firstNumNode, int *firstNumSide, int *anotherNumSide);
int deleteOnesInNumerator(Node **node, Node **div);
void deleteEmptyNodes(Node **node);
int deleteVarOp(Node **root,Node **node,Node **prev);

// Функции для работы с деревом:
Node *treeNodeCreate(void);
Node *treeCopy(Node **node);
int treeIsMinusNode(Node **node);
void treeBuild(Node **node, Stack *st);
void treeDestroy(Node **node);
void treePowReduce(Node **node);
void PKL(Node **node, const int level);
void LKP(Node **node);
int isLetter(const char ch);
int isNumber(const char ch);
int isOp(const char ch);
int isOpHigh(const char op1, const char op2);
void postOrder(const char *str, Stack *st);

void stackCreate(Stack *s);
int stackEmpty(const Stack *s);
int stackSize(const Stack *s);
int stackPush(Stack *s, const STACK_TYPE value);
int stackPop(Stack *s);
STACK_TYPE stackTop(const Stack *s);
void stackDestroy(Stack *s);

#endif
```

9. 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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы: -

11. Выводы:

Я научился преобразовывать выражение в дерево и работать с ним.

Подпись студента

_____

Отчет по лабораторной работе № 25-26 по курсу “ Практикум на ЭВМ ”

Студент группы М8О-111Б-21, Смирнов Никита Евгеньевич, № по списку 21, вариант 3, 5

Контакты www, e-mail, icq, skype: smirnov.nikita64@gmail.com

Работа выполнена: « » _____2022г.

Преподаватель: Никулин Сергей Петрович
Каф.806_____

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Абстрактные типы данных, рекурсия, сортировка, утилита make
2. **Цель работы:** Составить программу программу сортировки атд
3. **Задание 3 - Дэк, 5 - сортировка слиянием:**
4. **Оборудование:**

Оборудование ПЭВМ студента:
Процессор: intel-core i5-10500H, с ОЗУ 8 гб (виртуальная машина), SSD 512 гб. Монитор: встроенный 15 дюймов IPS 2560×1600, 220 PPI.

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

Операционная система семейства Windows, наименование Windows 11 pro
Прикладные системы и программы: виртуальная машина VMware.

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)
Deque.c - содержит все функции, требуемые в задании
Deque.h - заголовочный файл с описанием структуры
Main.c - меню для работы с дэком

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

1. Открыть консоль виртуальной машины
2. Собрать модули, использовать для этого утилиту make
3. Проверить работу программы на разных тестах

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя**

_____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

Работа по созданию Makefile и тесты.

```
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  main.c  makefile
nikita@ubuntu:~/Documents$ make main.o
gcc  -c main.c
nikita@ubuntu:~/Documents$ make deck.o
gcc  -c deque.c
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  deque.o  main.c  main.o  makefile
nikita@ubuntu:~/Documents$ make all
gcc  -c deque.c
gcc  deque.o main.o -o main
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  deque.o  main  main.c  main.o  makefile
nikita@ubuntu:~/Documents$ make clean
rm -rf *.o
rm -rf main
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  main.c  makefile
nikita@ubuntu:~/Documents$ make all
gcc  -c main.c
gcc  -c deque.c
gcc  deque.o main.o -o main
nikita@ubuntu:~/Documents$ make run
gcc  -c deque.c
gcc  deque.o main.o -o main
./main
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка        8: Размер Дэка
9: Выйти
Номер: 1
Введите значение: 10
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка        8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 12
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка        8: Размер Дэка
```

9: Выйти
Номер: 5
10 12
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 8
Размер дэка: 2
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 3
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
12
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 10
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
12 10
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 4
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5

12
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 4
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 1
Введите значение: 9
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 7
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 1
Введите значение: 5
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 3
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
5 9 7 3
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 1
Введите значение: 8

Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 4
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
8 5 9 7 3 4
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 6
Дэк отсортирован

Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
3 4 5 7 8 9
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 2
Введите значение: 1
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 5
3 4 5 7 8 9 1
Введите номер команды.
1: Добавить узел в начало 2: Добавить узел в конец
3: Удалить узел с начала 4: Удалить узел с конца 5: Напечатать Дэк
6: Отсортировать Дэк 7: Проверка на пустоту Дэка 8: Размер Дэка
9: Выйти
Номер: 6

```
Дэк отсортирован

Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 5
1 3 4 5 7 8 9
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 7
Пустой ли дэк?: Нет
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 8
Размер дэка: 7
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 6
Дэк отсортирован

Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 5
1 3 4 5 7 8 9
Введите номер команды.
1: Добавить узел в начало          2: Добавить узел в конец
3: Удалить узел с начала   4: Удалить узел с конца     5: Напечатать Дэк
6: Отсортировать Дэк        7: Проверка на пустоту Дэка       8: Размер Дэка
9: Выйти
Номер: 9
Остановка
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  deque.o  main  main.c  main.o  makefile
nikita@ubuntu:~/Documents$ make clean
rm -rf *.o
rm -rf main
```

```
nikita@ubuntu:~/Documents$ ls
deque.c  deque.h  main.c  makefile
nikita@ubuntu:~/Documents$
```

Содержимое файлов

Main.c

```
#include "deque.h"
#include <stdio.h>
#include <stdlib.h>

int main() {
   deque *d;
   int working = 1;
   d = deque_create(d);
   while(working) {
      int input = 0, n;
      printf("Введите номер команды.\n1: Добавить узел в начало\t2: Добавить узел в конец\n3:
Удалить узел с начала\t4: Удалить узел с конца\t5: Напечатать Дэк\n6: Отсортировать Дэк\t7:
Проверка на пустоту Дэка\t8: Размер Дэка\n9: Выйти\nНомер: ");
      scanf("%d", &n);
      switch(n){
         case 1:
         {
               int a;
               printf("Введите значение: ");
               scanf("%d", &a);
               push_front(d, a);

            break;
         }
         case 2:
         {
               int a;
               printf("Введите значение: ");
               scanf("%d", &a);
               push_back(d, a);
            break;
         }
         case 3:
         {
               pop_front(d);
            break;
         }
         case 4: {
            if(d->capasity != 0){
               pop_back(d);
            }
```

```
            else printf("Дэк не существует\n\n");
            break;
        }
        case 5:
            if(d->capasity != 0) {
                int ind = d->number_of_elements;
                for (int i = 0; i < ind; i++){
                    printf("%d ", d->elements[i]);
                }
                printf("\n");
            }
            else printf("Дэк не существует\n\n");
            break;
        case 6:
            if(d->capasity != 0) {
                merge_sort(d);
                printf("Дэк отсортирован\n\n");
            }
            else printf("Дэк не существует\n\n");
            break;
        case 7:
            if(d->capasity != 0) {
                printf("Пустой ли дэк?: %s\n", !empty(d) ? "Нет" : "Да");
            }
            else printf("Дэк не существует\n");
            break;
        case 8:
            if(d->capasity != 0) {
                printf("Размер дэка: %d\n", size(d));
            }
            else printf("Дэк не существует\n");
            break;
        case 9:
            printf("Остановка\n");
            working = 0;
            break;
      }
   }
   return 0;
}
```

Deque.c

```
#include "deque.h"
#include <stdlib.h>

deque *deque_create()
{
   deque *a = (deque*)malloc(sizeof(deque));
```

```
    a->elements = malloc(sizeof(int));
    a->capasity = 1;
    a->number_of_elements = 0;
    return a;
}

void push_front(deque *a, int b)
{
    if(a->number_of_elements == a->capasity) {
        resize(a);
    }
    int tmp1, tmp2;
    tmp1 = b;
    for(int i = 0; i < a->number_of_elements + 1; i++) {
        tmp2 = a->elements[i];
        a->elements[i] = tmp1;
        tmp1 = tmp2;
    }
    a->number_of_elements++;
}

void push_back(deque *a, int b)
{
    if (a->number_of_elements == deque_size(a)) {
        resize(a);
    }
    a->elements[a->number_of_elements] = b;
    a->number_of_elements++;
}

void pop_back(deque *a)
{
    if (a->number_of_elements > 0) {
        a->number_of_elements--;
    }
}

void pop_front(deque *a)
{
    if (a->number_of_elements > 0) {
        for(int i = 0; i < a->number_of_elements - 1; i++) {
            a->elements[i] = a->elements[i + 1];
        }
        a->number_of_elements--;
    }
}

int first_front(deque* a)
{
    return a->elements[0];
```

```
}

int first_back(deque* a)
{
   return a->elements[a->number_of_elements - 1];
}

int empty(deque *a)
{
   if (a->number_of_elements == 0) {
      return 1;
   } else {
      return 0;
   }
}

void resize(deque *a)
{
   a->capasity++;
   a->elements = realloc(a->elements, a->capasity * sizeof(int));
}

int size(deque* a)
{
   return a->number_of_elements;
}

int deque_size(deque *a)
{
   return a->capasity;
}

deque* reverse(deque* a)
{
   deque* b = deque_create();
   while (!empty(a)) {
      push_front(b, first_front(a));
      pop_front(a);
   }
   return b;
}

void merge(deque* res, deque* a, deque* b)
{
   deque *c =  deque_create();
   while (!empty(a) && !empty(b)) {
      if (first_front(a) < first_front(b)) {
         push_back(c, first_front(a));
         pop_front(a);
      } else {
         push_back(c, first_front(b));
```

```
            pop_front(b);
        }
    }
    while (!empty(a)) {
        push_back(c, first_front(a));
        pop_front(a);
    }
    while (!empty(b)) {
        push_back(c, first_front(b));
        pop_front(b);
    }
    deque* new = reverse(c);
    while (!empty(new)) {
        push_front(res, first_front(new));
        pop_front(new);
    }
}

void merge_sort(deque* a)
{
    if (size(a) > 1) {
        deque *b = deque_create(), *c = deque_create();;
        while (!empty(a)) {
            if (size(a) % 2 == 0) {
                push_front(b, first_front(a));
                pop_front(a);
            } else {
                push_front(c, first_front(a));
                pop_front(a);
            }
        }
        merge_sort(b);
        merge_sort(c);
        merge(a, b, c);
    }
}
```

Deque.h

```
#ifndef DEQUE
#define DEQUE

#include <stdio.h>

typedef struct _deque deque;

struct _deque
{
    int *elements;
    int capasity;
    int number_of_elements;
```

```
};

deque *deque_create();
void push_front(deque* a, int b);
void push_back(deque* a, int b);
int first_front(deque* a);
int first_back(deque* a);
int empty(deque* a);
void pop_front(deque* a);
void pop_back(deque* a);
void resize(deque* a);
int size(deque* a);
int deque_size(deque* a);

deque* reverse(deque* a);
void merge(deque* res, deque* a, deque* b);
void merge_sort(deque* a);
#endif
```

Makefile

```
GCCFLAGS=-w
run: all
./main
all: main.o deck.o
gcc $(GCCFLAGC) deque.o main.o -o main
main.o:
gcc $(GCCFLAGC) -c main.c
deck.o:
gcc $(GCCFLAGC) -c deque.c
clean:
rm -rf *.o
```

9. 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. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы: -

11. Выводы:

Я научился пользоваться утилитой make, а также изучил атд - дэк и модульное программирование.

Подпись студента

_____