**Отчет по лабораторной работе №25-26**

по курсу Практикум программирования

Студент группы М8О-111Б-23 Бугренков Владимир Петрович,

№ по списку 4, Контакты e-mail: vladimir.bugrenkov@yandex.ru

Работа выполнена: «3» мая 2023 г.

Преподаватель: доцент каф. 806 Никулин Сергей Петрович

Отчет сдан « » \_\_\_\_\_\_\_\_\_20 \_\_\_ г., итоговая оценка \_\_\_\_\_\_\_

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

1. **Тема:** абстрактные типы данных, рекурсия, модульное программирование на ЯП Си. Автоматизация сборки программ модульной структуры с использованием утилиты make.
2. **Цель работы:** применение различных сортировок к различным типам данных и обучение по работе с утилитой make
3. **Задание** АТД – Очередь. Процедура – поиск и удаление минимального элемента. Метод – сортировка линейным выбором.
4. **Оборудование** (лабораторное):

ЭВМ \_\_\_\_\_\_\_\_\_, процессор \_\_\_\_\_\_\_. имя узла сети\_\_\_\_\_\_\_с ОП\_\_\_\_\_ Мб, НМД \_\_\_\_ Мб. Терминал \_\_\_\_\_\_\_\_адрес \_\_\_\_\_\_\_\_. Принтер\_\_\_\_\_\_\_

Другие устройства

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

*Оборудование ПЭВМ студента, если использовалось:*

Процессор \_AMD Ryzen 5 \_ с ОП 16 Гб НМД \_\_\_512 Гб. Монитор 1920x1080~60Hz Другие устройства

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

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

Операционная система семейства \_\_\_\_\_\_\_\_\_\_\_, наименование \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия

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

интерпретатор команд \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_\_\_\_\_\_

Система программирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Редактор текстов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утилиты операционной системы

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

Прикладные системы и программы

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

Местонахождение и имена файлов программ и данных\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства \_Linux\_, наименование \_Ubuntu\_ версия 22.04.2 интерпретатор команд \_GNU bash\_ версия 5.1.16.

Система программирования С.

Редактор текстов emacs версия 29.1

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

Прикладные системы и программы Emacs

Местонахождение и имена файлов программ и данных на домашнем компьютере /home/

1. **Идея, метод, алгоритм** решения задачи(в формах:словесной,псевдокода,графической[блок-схема,диаграмма,рисунок,таблица]

Для выполнения данного задания требуются структуры data\_type и queue:

typedef struct data\_type {

int key;

int value;

} data\_type;

typedef struct queue {

int first;

int size;

data\_type data[POOL\_SIZE];

} queue;

Функции:

1. Create: инициализирует очередь.
2. Empty: проверка на пустоту.
3. Size: размер очереди.
4. Push: добавление элемента в конец очереди.
5. Pop: удаление первого элемента очереди.
6. Top: возвращает значение первого элемента.
7. Print: печать очереди.
8. Destroy: удаляет очередь.
9. min\_pop: поиск и удаление минимального элемента
10. qsort: сортировка линейным выбором

Сортировка линейным выбором над очередью (qsort) реализована следующим образом:

1. Создаем вспомогательную очередь

2. Ищем в оригинальной очереди минимальный элемент

3. Добавляем минимальный элемент в вспомогательную очередь

4. Удаляем из оригинальной очереди минимальный элемент

5. Повторяем 2-3-4 пока оригинальная очередь не опустеет

6. Меняем оригинальную очередь и временную местами

7. Получаем отсортированную по возрастанию очередь

1. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию].

Код программы:

**main.c:**

#include <stdio.h>

#include <stdbool.h>

#include "queue.c"

#include "qsort.c"

**void** menu() {

puts("---------------------------------------------");

puts("█ 1) Добавить элемент в очередь.");

puts("█ 2) Удаление первого элемента в очереди.");

puts("█ 3) Размер очереди.");

puts("█ 4) Удаление минимального элемента.");

puts("█ 5) Печать очереди.");

puts("█ 6) Сортировка очереди.");

puts("█ 0) Выход.");

puts("---------------------------------------------");

printf("▷▷▷ ");

}

**int** main() {

queue q;

data\_type t;

Create(&q);

**char** c;

menu();

scanf("%c", &c);

**while** (c != '0') {

**switch** (c) {

**case** '1':

printf("Введите элемент(ключ - значение):**\n**▷▷▷ ");

scanf("%d - %d", &t.key, &t.value);

Push(&q, t);

**break**;

**case** '2':

**if** (!Empty(&q)) {

printf("Элемент <%d - %d> удалён.**\n**", Top(&q).key, Top(&q).value);

Pop(&q);

} **else** {

puts("Очередь пуста!");

}

**break**;

**case** '3':

printf("Размер очереди: %d", Size(&q));

**break**;

**case** '4':

q = min\_pop(&q);

puts("Минимальный элемент был удален");

**break**;

**case** '5':

Print(&q);

**break**;

**case** '6':

q = qsort(&q);

puts("Очередь отсортирована!");

**break**;

default:

puts("Неизвестное значение!");

**break**;

}

menu();

scanf("**\n**%c", &c);

}

Destroy(&q);

puts("Программа завершена!");

}

**queue.h:**

#ifndef \_QUEUE\_H\_

#define \_QUEUE\_H\_

#include <stdbool.h>

#define POOL\_SIZE 100

**typedef** **struct** data\_type {

**int** key;

**int** value;

} data\_type;

**typedef** **struct** queue {

**int** first;

**int** size;

data\_type data[POOL\_SIZE];

} queue;

**void** Create(queue \*q);

**bool** Empty(queue \*q);

**int** Size(queue \*q);

**bool** Push(queue \*q, **const** data\_type t);

**bool** Pop(queue \*q);

**void** Print(queue \*q);

data\_type Top(**const** queue \*q);

**void** Destroy(queue \*q);

#endif

**queue.c:**

#include "queue.h"

#include <stdio.h>

**void** Create(queue \*q) {

q->first = 0;

q->size = 0;

}

**bool** Empty(queue \*q) {

**return** q->size == 0;

}

**int** Size(queue \*q) {

**return** q->size;

}

**bool** Push(queue \*q, **const** data\_type t) {

**if** (q->size == POOL\_SIZE)

**return** false;

q->data[(q->first + q->size++) % POOL\_SIZE] = t;

**return** true;

}

**bool** Pop(queue \*q) {

**if** (!q->size)

**return** false;

q->first++;

q->first %= POOL\_SIZE;

q->size--;

**return** true;

}

data\_type Top(**const** queue \*q) {

**if** (q->size)

**return** q->data[q->first];

}

**void** Print(queue \*q) {

queue cur = \*q;

putchar('[');

**while** (!Empty(&cur)) {

printf(" %d:%d", Top(&cur).key, Top(&cur).value);

Pop(&cur);

}

puts(" ]");

}

**void** Destroy(queue \*q) {

q->size = 0;

}

**qsort.c:**

#include "queue.h"

#include "stdio.h"

**int** min\_search(queue \*q) {

*// Нашли минимальный элемент*

queue cur = \*q;

**int** min = 2147483647;

**while** (!Empty(&cur)) {

**if** (min > Top(&cur).value)

min = Top(&cur).value;

Pop(&cur);

}

**return** min;

}

queue minimal\_search(queue \*q) {

**if** (Empty(q))

**return** \*q;

**if** (Size(q) == 1)

**return** \*q;

**int** min = min\_search(q);

queue cur = \*q;

queue new\_temp\_que;

Create(&new\_temp\_que);

printf("До: ");

Print(&cur);

**int** flag = 0;

**while** (!Empty(&cur)) {

**if** ((min == Top(&cur).value) && (flag == 0)) {

flag = 1;

} **else** {

Push(&new\_temp\_que, Top(&cur));

}

Pop(&cur);

}

**return** new\_temp\_que;

}

queue min\_pop(queue \*q) {

**if** (Empty(q))

**return** \*q;

**if** (Size(q) == 1) {

**return** \*q;

}

\*q = minimal\_search(q);

printf("После: ");

Print(q);

**return** \*q;

}

*// линейная сортировка очереди*

*// ищем минимальный элемент, добавляем его во временную очередь и удаляем из оригинальной. Повторяем так пока оригинальная очередь не опустеет*

queue qsort(queue \*q) {

**if** (Empty(q))

**return** \*q;

**if** (Size(q) == 1) {

**return** \*q;

}

printf("ДО: ");

Print(q);

queue cur = \*q;

queue help\_que;

Create(&help\_que);

**const** **int** size\_que = Size(&cur);

**int** min\_value;

**for** (**int** i = 0; i < size\_que; i++) {

*// Ищем минимальный элемент в очереди*

min\_value = min\_search(&cur);

*// Cоздаем временную очередь для удаления из оригинальной минимального текущего элемента*

queue temp\_que;

Create(&temp\_que);

**int** flag = 0;

**while** (!Empty(&cur)) {

**if** ((min\_value == Top(&cur).value) && (flag == 0)) {

flag = 1;

*// Добавляем min\_value в вспомогательную очередь*

Push(&help\_que, Top(&cur));

} **else** {

Push(&temp\_que, Top(&cur));

}

Pop(&cur);

}

cur = temp\_que;

}

printf("После: ");

Print(&help\_que);

**return** help\_que;

}

**make:**

main: main.c qsort.o queue.o

gcc -o main main.c qsort.o queue.o

qsort.o: qsort.c queue.h

gcc -c qsort.c

queue.o: queue.c queue.h

gcc -c queue.c

clean:

rm \*.o main

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

qwental@DESKTOP-NKF1EUK:/mnt/c/Users/Holiday/Desktop/Мои материалы/Информатика/2 Семестр/ЛР25-26\_$ make main

cc main.c -o main

qwental@DESKTOP-NKF1EUK:/mnt/c/Users/Holiday/Desktop/Мои материалы/Информатика/2 Семестр/ЛР25-26\_$ ./main

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 5

[ ]

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 1-10

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 2-8

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 3-12

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 4-14

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 5-6

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 6-18

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 7-0

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 5

[ 1:10 2:8 3:12 4:14 5:6 6:18 7:0 ]

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 4

До: [ 1:10 2:8 3:12 4:14 5:6 6:18 7:0 ]

После: [ 1:10 2:8 3:12 4:14 5:6 6:18 ]

Минимальный элемент был удален

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 5

[ 1:10 2:8 3:12 4:14 5:6 6:18 ]

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 6

ДО: [ 1:10 2:8 3:12 4:14 5:6 6:18 ]

После: [ 5:6 2:8 1:10 3:12 4:14 6:18 ]

Очередь отсортирована!

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 5

[ 5:6 2:8 1:10 3:12 4:14 6:18 ]

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 22-3

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 2

Элемент <5 - 6> удалён.

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 2

Элемент <2 - 8> удалён.

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 6

ДО: [ 1:10 3:12 4:14 6:18 22:3 ]

После: [ 22:3 1:10 3:12 4:14 6:18 ]

Очередь отсортирована!

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 10-2342

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 12-5433

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 1-999999

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 4

До: [ 22:3 1:10 3:12 4:14 6:18 10:2342 12:5433 1:999999 ]

После: [ 1:10 3:12 4:14 6:18 10:2342 12:5433 1:999999 ]

Минимальный элемент был удален

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 4

До: [ 1:10 3:12 4:14 6:18 10:2342 12:5433 1:999999 ]

После: [ 3:12 4:14 6:18 10:2342 12:5433 1:999999 ]

Минимальный элемент был удален

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 4

До: [ 3:12 4:14 6:18 10:2342 12:5433 1:999999 ]

После: [ 4:14 6:18 10:2342 12:5433 1:999999 ]

Минимальный элемент был удален

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 2

Элемент <4 - 14> удалён.

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 5

[ 6:18 10:2342 12:5433 1:999999 ]

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 1

Введите элемент(ключ - значение):

▷▷▷ 1-0

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 6

ДО: [ 6:18 10:2342 12:5433 1:999999 1:0 ]

После: [ 1:0 6:18 10:2342 12:5433 1:999999 ]

Очередь отсортирована!

---------------------------------------------

█ 1) Добавить элемент в очередь.

█ 2) Удаление первого элемента в очереди.

█ 3) Размер очереди.

█ 4) Удаление минимального элемента.

█ 5) Печать очереди.

█ 6) Сортировка очереди.

█ 0) Выход.

---------------------------------------------

▷▷▷ 0

Программа завершена!

qwental@DESKTOP-NKF1EUK:/mnt/c/Users/Holiday/Desktop/Мои материалы/Информатика/2 Семестр/ЛР25-26\_$exit

**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 |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Выводы**

Я научился работать с абстрактными типами данных и сортировками в них.

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