**Отчет по лабораторной работе №** 25-26 по курсу 1

Студент группы М80-108Б-18, Коростелев Д.В. № по списку 12

E-mail [dmitry.k48@yandex.ru](mailto:dmitry.k48@yandex.ru)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа выполнена: « 25 » мая 2019 г.

Преподаватель: Поповкин А.В. каф.806 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Отчет сдан « » \_\_\_\_\_\_\_\_\_ 201\_\_ г., итоговая оценка \_\_\_\_\_\_

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

1. **Тема:** Языки и методы программирования, алгоритмы и структуры данных.
2. **Цель работы:** составить MakeFile для модульной программы из лабораторной работы №26, составить и отладить модуль определений и модуль реализации по заданной схеме модуля определений для абстрактного типа данных, составить программный модуль сортирующий экземпляр указанного абстрактного типа данныз заданным методом, используя только операции, импортированные из модуля UDT.
3. **Задание (***вариант №* 12**):** АТД: очередь, Процедура: вставка элемента в очередь, упорядоченные по возрастанию с сохранением порядка. Метод: сортировка простой вставкой.
4. **Оборудование (лабораторное):**

ЭВМ Cameron, процессор Intel Core2 Duo CPU E8500 3.16ГГц, имя узла сити cameron с ОП 16029 Мб, НДМ \_\_\_\_\_\_\_\_\_ Гб. Терминал gnome, адрес 172.16.80.213. Принтер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

*Оборудование ПЭВМ студента, если использовалось:*

Процессор Intel Core i7-7700HQ 4x 2.808ГГц с ОП 3945 Мб, НДМ 48.9 Гб Монитор \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

Операционная система семейства UNIX , наименование UBUNTU версия \_16.04\_\_\_\_\_\_\_\_\_\_\_\_, интерпретатор команд bash версия 4.3.48 , Система программирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Утилиты операционной системы: cd, cat, nano, cc, make \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства UNIX, наименование UBUNTU\_\_\_ версия \_\_\_\_\_\_18.04\_\_\_\_\_\_\_, интерпретатор команд \_\_\_bash\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия\_\_\_\_\_\_4.3.48\_\_\_\_\_\_\_

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

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

Утилиты операционной системы: cd, cat, nano, cc. make\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Прикладные системы и программы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных на домашнем компьютере: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

В программе требуется реализовать 5 функций, функцию добавления в конец очереди, удаление первого элемента, печати очереди, добавления элемента в отсортированную очередь с сохранением порядка, и сортировку.

Функция добавления: принимает указатель на очередь, и добавляемое значение, с помощью realloc обновляем размер массива и вписываем новое значение.

Функция удаления первого элемента: делаем логический сдвиг массива в лево, удаляем последний элемент

Функция печати очереди: итерационно печатаем массив очереди.

Функция добавления в отсортированною очередь: используется метод простой вставки

Функция сортировки очереди: для нее требуется реализовать еще функцию удаления и печати последнего элемента очереди, далее мы size раз запоминаем последний элемент, удаляем его из очереди, а потом с помощью функции добавления в отсортированную очередь добавляем эти элементы в очередь.

1. **Сценарий выполнения работы** (план работы, первоначальный текст программы на черновике или отдельном листе, тесты либо соображения по тестированию)

*Функции*

|  |  |  |
| --- | --- | --- |
| **№** | **Функция** | **Описание** |
| 1. | QUEUE\* pop\_tail(QUEUE\* q); | Удалить последний элемент очереди |
| 2. | QUEUE\* push\_q(QUEUE\* q, int el);  QU | Добавить элемент в очередь |
| 3. | QUEUE\* printq(QUEUE\* q); | Напечатать очередь |
| 4. | QUEUE\* add(QUEUE\* q, int el); | Добавить элемент в отсортированную очередь |
| 5. | QUEUE\* pop\_font(QUEUE\* q); | Удалить первый элемент очереди |
| 6. | QUEUE\* sortq(QUEUE \*q); | Отсортировать очередь |
| 7. | int pop(QUEUE \*q); | Вывести первый элемент очереди |

Переменные

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Переменная** | **Тип** | **Назначение** |
| 1. | array | int\* | Указатель на массив данных, часть структуры queue |
| 2. | size | int | Размер очереди, часть структуры queue |
| 3. | temp | Временное хранилище |
| 4. | q | struct queue\* | Очередь |

*Тесты*

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Выходные данные** |
| 1. | 1 2 3 | 1 2 3 |
| 2. | 1 2 | 1 2 |
| 3. | 1 3 2 | 1 2 3 |
| 4. | 1 2 3 4 | 1 2 3 4 |
| 5 | 1 2 4 3 | 1 2 3 4 |

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

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

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

|  |
| --- |
| dmitry@dmitry-VirtualBox:~/lubs/25-26$ cat MakeFile  CC = cc #имя компилятора  CCFLAGS = -c -Wall #ключи компилятора  RESULT\_NAME = 26.out  all: $(RESULT\_NAME) clean  $(RESULT\_NAME): main.o lib.o variant.o  $(CC) main.o lib.o variant.o -o $(RESULT\_NAME)  main.o: main.c  $(CC) $(CCFLAGS) header.h main.c  lib.o: lib.c  $(CC) $(CCFLAGS) header.h lib.c  clean:  rm \*.o \*.gch  dmitry@dmitry-VirtualBox:~/lubs/25-26$ cat main.c  #include<stdio.h>  #include<malloc.h>  #include"header.h"  int main(void){  QUEUE\* q;  q = malloc(sizeof(QUEUE));  q->size = 0;  int menu;  int value;  while(1){  printf("1. Add\n2. Pop\n3. Push\_back\n4. Print\n5. Sort\n");  printf("|");  scanf("%d",&menu);  switch(menu){  case 1:  scanf("%d",&value);  q = add(q, value);  break;  case 2:  q = pop\_font(q);  break;  case 3:  scanf("%d", &value);  q = push\_q(q,value);  break;  case 4:  printq(q);  break;  case 5:  sortq(q);  break;  default: return 0;  }  }  return 0;  }  dmitry@dmitry-VirtualBox:~/lubs/25-26$ cat header.h  typedef struct queue{  int \*array;  int size;  } QUEUE;  int tailq(QUEUE\* q);  QUEUE\* pop\_tail(QUEUE\* q);  QUEUE\* push\_q(QUEUE\* q, int el);  QUEUE\* printq(QUEUE\* q);  QUEUE\* add(QUEUE\* q, int el);  int pop(QUEUE \*q);  QUEUE\* pop\_font(QUEUE\* q);  QUEUE\* sortq(QUEUE \*q);  dmitry@dmitry-VirtualBox:~/lubs/25-26$ cat variant.c  #include<stdio.h>  #include<malloc.h>  #include<stdlib.h>  #include"header.h"  QUEUE\* sortq(QUEUE \*q){  for(int i = 0;i< q->size;i++){  int temp = tailq(q);  q = pop\_tail(q);  q = add(q, temp);  }  return (q);  }  dmitry@dmitry-VirtualBox:~/lubs/25-26$ cat lib.c  #include<stdio.h>  #include<malloc.h>  #include<stdlib.h>  #include"header.h"  int tailq(QUEUE\* q){  return q->array[q->size - 1];  }  QUEUE\* pop\_tail(QUEUE\* q){  if(q->array == NULL)  return (q);  q->array = realloc(q->array,(q->size - 1) \* sizeof(int));  q->size--;  return (q);  }  QUEUE\* push\_q(QUEUE\* q, int el){  if(q->array==NULL){  q->array = malloc(sizeof(int));  q->array[0] = el;  q->size++;  return (q);  }  q->array = realloc(q->array, q->size\*sizeof(int));  q->array[q->size] = el;  q->size++;  return (q);  }  QUEUE\* printq(QUEUE\* q){  if(q->array == NULL){  printf("Queue is empty\n");  return(q);  }  for(int i = 0;i< q->size;i++){  printf("%d ",q->array[i]);  }  printf("\n");  }  QUEUE\* add(QUEUE\* q, int el){  if(q->array == NULL){  q = push\_q(q,el);  return (q);  }  int i = 0;  while(q->array[i] < el && i< q->size && q->array[j]<q->array[j+1]){  i++;  }  if(i == q->size){  q = push\_q(q,el);  return(q);  }  q->array = realloc(q->array, q->size \* sizeof(int));  q->size++;  int temp = q->array[i];  for(int k = q->size;k>i;k--){  q->array[k] = q->array[k-1];  //temp = q->array[k+1];  }  q->array[i] = el;  return (q);  }  int pop(QUEUE \*q){  return q->array[0];  }  QUEUE\* pop\_font(QUEUE\* q){  if(q->array == NULL)  return (q);  //int temp = q->array[0];  for(int i = 1;i< q->size;i++){  q->array[i-1] = q->array[i];  }  q->array = realloc(q->array,(q->size - 1) \* sizeof(int));  q->size--;  return (q);  }  dmitry@dmitry-VirtualBox:~/lubs/25-26$ ls  26.c a.out header.h lib.c main.c MakeFile variant.c  dmitry@dmitry-VirtualBox:~/lubs/25-26$ make -f MakeFile  cc -c -Wall header.h main.c  cc -c -Wall header.h lib.c  lib.c: In function ‘add’:  lib.c:58:6: warning: unused variable ‘temp’ [-Wunused-variable]  int temp = q->array[i];  ^~~~  lib.c: In function ‘printq’:  lib.c:41:1: warning: control reaches end of non-void function [-Wreturn-type]  }  ^  cc -c -o variant.o variant.c  cc main.o lib.o variant.o -o 26.out  rm \*.o \*.gch  dmitry@dmitry-VirtualBox:~/lubs/25-26$ ./26.out  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |1  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  2 4 5 2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |1  6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 5 2 6  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  Queue is empty  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  3  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 3 2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 2 3  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  Queue is empty  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 2 4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 2 4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  Queue is empty  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  3  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  3 2 1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 2 3  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  3  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  2  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |3  1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  4 3 2 1  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |5  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |4  1 2 3 4  1. Add  2. Pop  3. Push\_back  4. Print  5. Sort  |6 |

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

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

1. **Замечания автора по существу работы:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. **Выводы:** данные лабораторные работы помогают студенту разобраться в модульной разработке программ, понять, что большие сложные проекты следует делить на различные блоки, также студент получает необходимый опыт в реализации абстрактных типов данных, из обработки и взаимодействия с ними.

Недочеты при выполнении задания могут быть устранены следующим образом: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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