**Отчет по лабораторным работам № 25-26** по курсу “ Практикум на ЭВМ ”

Студент группы М8О-102Б-21, Яценко Александр Владимирович, № по списку 20

Контакты: e-mail - alexander.iatsenko@gmail.com

Работа выполнена: « » \_\_\_\_\_\_\_\_\_\_\_\_201\_\_\_г.

Преподаватель: Никулин Сергей Петрович Каф.806\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Отчет сдан « » \_\_\_\_\_\_\_\_\_201 \_\_ г., итоговая оценка \_\_\_\_\_

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

**1. Тема:** Автоматизация сборки программ модульной структуры на языке Си с использованием утилиты make. Абстрактные типы данных. Рекурсия. Модульное программирование на языке Си

**2. Цель работы:** Изучить принципы работы утилиты make. Составить и отладить определённый модуль реализации по заданной схеме модуля для абстрактного типа данных.

**3. Задание:**

АТД: очередь

Процедура: поиск в очереди двух элементов, идущих подряд, первый из которых больше второго. Если такие элементы найдены, их перестановка.

Метод: сортировка методом пузырька

**4. Оборудование:**

*Оборудование ПЭВМ студента:*

Процессор: AMD Ryzen 5 4600H, с ОП 6 Мб (виртуальная машина), НМД 25600 Мб. Монитор: Huawei 16,1 IPS 1920×1080, 137 PP.

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

Операционная система семейства Linux, наименование Fedora, версия 5.13.12-200. Fc24.x86\_64 GNU/Linux

Прикладные системы и программы: CLion

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

Алгоритм сортировки заключается в повторении процедуры до тех пор, пока она не вернёт значение “false”, которое значит завершение сортировки. В ходе выполнения процедуры ищется элемент списка, который будет больше следующего. Если такой находится, то в буферную очередь записывается элемент из очереди и только потом проверяемый элемент.

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

**Код программы makefile:**

CC = gcc

LD = gcc

OBJ = main.o queue.o sort.o

.SUFFIXEX: .c .o

run: $(OBJ)

$(LD) -o run $(OBJ)

.c.o :

$(CC) -c $< -o $@

main.o: main.c queue.h

queue.c: queue.h

sort.c: queue.h

clean :

rm -f \*.o run

**Код программы main.c:**

#include "queue.h"

int main() {

queue q; q\_create(&q);

data\_type x;

printf("Select: 1-push 2-print 3-pop ");

printf("4-front 5-size 6-sort 7-exit\n");

int state = -1;

while (state != 7) {

printf(">");

scanf("%d", &state);

switch (state) {

case 1:

scanf("%lf %d", &(x.key), &(x.value));

q\_push(&q, &x);

break;

case 2:

q\_print(&q);

break;

case 3:

q\_pop(&q);

break;

case 4:

if (!q\_is\_empty(&q)) {

x = \*q\_front(&q);

printf("%.2lf %d\n", x.key, x.value);

}

break;

case 5:

printf("size = %d\n", q\_size(&q));

break;

case 6:

q\_sort(&q);

}

}

return 0;

}

**Код заголовочного файла queue.h:**

#include <stdbool.h>

#include <stdio.h>

#define MAX\_SIZE 10000

typedef struct {

double key;

int value;

} data\_type;

typedef struct queue {

int top, size;

data\_type data[MAX\_SIZE];

} queue;

void q\_create(queue \*);

bool q\_is\_empty(const queue \*);

void q\_push(queue \*, data\_type \*);

void q\_pop(queue \*);

void q\_print(const queue \*);

int q\_size(const queue \*);

data\_type\* q\_front(queue \*);

george@hp:~/Рабочий стол/labs/labs/2sem/25-26$ cat queue.c

#include "queue.h"

void q\_create(queue \*q) {

q->top = q->size = 0; }

bool q\_is\_empty(const queue \*q) { return q->size == 0; }

void q\_push(queue \*q, data\_type\* x) {

if (q->size != MAX\_SIZE) {

q->data[(q->top + q->size) % MAX\_SIZE].key = x->key;

q->data[(q->top + q->size) % MAX\_SIZE].value = x->value;

q->size++;

}

}

void q\_pop(queue \*q) {

if (!q\_is\_empty(q)) {

q->size--;

q->top = (q->top + 1) % MAX\_SIZE;

}

}

void q\_print(const queue \*q) {

for (int i = 0, top = q->top; i < q->size; i++, top = ++top % MAX\_SIZE)

printf("%d: %.2lf %d\n", i, q->data[top].key, q->data[top].value);

}

int q\_size(const queue \*q) { return q->size; }

data\_type\* q\_front(queue \*q) {

data\_type \*x = &(q->data[q->top]);

return x;

}

**Код программы sort.c:**

#include "queue.h"

bool procedure(queue \*q) {

data\_type x;

queue w; q\_create(&w);

bool flag = false;

while (!q\_is\_empty(q)) {

x = \*q\_front(q); q\_pop(q);

if (!flag && !q\_is\_empty(q) && x.key > q\_front(q)->key) {

q\_push(&w, q\_front(q));

q\_pop(q);

flag = true;

}

q\_push(&w, &x);

}

q\_create(q);

while (!q\_is\_empty(&w)) {

q\_push(q, q\_front(&w));

q\_pop(&w);

}

return flag;

}

void q\_sort(queue \*q) { if (q\_size(q) > 1) while (procedure(q)) {} }

*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,подписанныйпреподавателем)

Select: 1-push 2-print 3-pop 4-front 5-size 6-sort 7-exit

>2

>3

>4

>5

size = 0

>6

>1

3 4

>1

2 5

>1 4 3

>1 1 4

>2

0: 3.00 4

1: 2.00 5

2: 4.00 3

3: 1.00 4

>4

3.00 4

>5

size = 4

>6

>2

0: 1.00 4

1: 2.00 5

2: 3.00 4

3: 4.00 3

>3

>2

0: 2.00 5

1: 3.00 4

2: 4.00 3

>1 1 6

>2

0: 2.00 5

1: 3.00 4

2: 4.00 3

3: 1.00 6

>6

>2

0: 1.00 6

1: 2.00 5

2: 3.00 4

3: 4.00 3

>7

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

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

1. **Замечания автора** по существу работы: -
2. **Выводы:** работа выполнена

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