Отчет по лабораторной работе № 12 по курсу
Архитектура компьютера и информационных сетей

Студент группы М8О-103Б-22 Клименко Виталий Максимович, № по списку 11

Контакты www, e-mail, icq, skype vitalikklimenko96@gmail.com

Работа выполнена: 2022 г.

Преподаватель: доцент Никулин С.П.

Входной контроль знаний с оценкой _____

Отчет сдан « » _____ 202 __ г., итоговая оценка ___

Подпись преподавателя _____

1. **Тема:** Техника работы с целыми числами. Системы счисления

2. **Цель работы:** Составить программу на языке Си в целом типе данных, которая для любых допустимых и корректно введенных чисел в десятичном изображении, выполняет указанное задание

3. **Задание** (*вариант № 30*): Упорядочить цифры числа попарно по возрастанию (номер цифры считается справа налево)

4. **Оборудование** (*лабораторное*):
ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____ Мб,
НМД _____ Мб. Терминал _____ адрес _____. Принтер _____
Другие устройства _____

Оборудование ПЭВМ студента, если использовалось:
Процессор Intel 4x 3.5GHz с ОП 16 ГБ НМД HDD 200 ГБ . Монитор Встроенный 1920x1080
Другие устройства Touchpad Synaptics

5. **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 UNIX, наименование Pop!_OS версия 22.04 jammy
интерпретатор команд bash версия 5.1.16
Система программирования GNU Compiler Collection (GCC) версия 11.2.0
Редактор текстов Visual Studio Code версия 1.72.2
Утилиты операционной системы _____
Прикладные системы и программы gcc, gdb
Местонахождение и имена файлов программ и данных на домашнем компьютере _____

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

Для ввода числа использовать структуру данных string, описанную в отдельном файле. После ввода числа, развернуть его и упорядочить цифры числа попарно по возрастанию. Вывести полученное зачение

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

Ввести число, выполнить над ним указанные действия, вывести ответ

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)

```
vitos@vitos-pop:~/Studying/mai/labs/l12$ cat makefile
all:
gcc -std=c99 -g -Wall -Wextra -o main main.c string.c
./mainvitos@vitos-pop:~/Studying/mai/labs/l12$ cat string.c
#include "string.h"

//? Maybe deprecated, but those are for error handling
#define UNDEFINED 0
#define SUCCESS 1
#define ERROR -1

// How big initial memory is
const uint64_t INIT_CAPACITY = 2;
// How big relative to previous allocated memory new memory should be
const uint64_t EXTENDED_CAPACITY = 2;

// Memory-related macros
#define new(a, n) (a*)calloc(n, sizeof(a) * n)
#define reallocate(ptr, a, n) (a*)realloc(ptr, sizeof(a) * n)

#define max(a, b) (a > b ? a : b)
#define min(a, b) (a < b ? a : b)

// Initialize string with default capacity
int init_string(string* s) {
    int result = UNDEFINED;
    s->memory_size = INIT_CAPACITY;
    s->values = new(char, INIT_CAPACITY);
    s->last_element = 0;
    s->values[0] = '\0';
    return result;
}

// Adds memory
int string_resize(string* s, uint64_t capacity) {
    uint64_t new_capacity = s->memory_size * capacity;
    s->memory_size = new_capacity;
    s->values = reallocate(s->values, char, new_capacity);
    return SUCCESS;
}

// Copies s2 to s1 (taking minimal memory size): s1='aboba', s2='bobr' -> s1='bobr '
int copy_string(string* s1, string s2) {
    int result = UNDEFINED;
    uint64_t minimal = min(s1->last_element, s2.last_element);
    for (uint64_t i = 0; i < s1->last_element; ++i) {
        if (i < minimal) {
            s1->values[i] = s2.values[i];
        } else {
            s1->values[i] = ' ';
        }
    }
    s1->last_element = minimal;
    return result;
}

// Adds a char to string, adding more memory if needed
int add_char(string* s, char value) {
    int result = UNDEFINED;
    if (s->last_element < s->memory_size + 1) {
        result = SUCCESS;
    } else {
        result = string_resize(s, EXTENDED_CAPACITY);
    }
    s->values[s->last_element] = value;
    s->values[s->last_element + 1] = '\0';
    s->last_element++;
    return result;
}

// Reads string char by char returning 1 if EOF
int read_string(string* s) {
    char c = ' ';
    int end = 0;

    while (1) {
        c = getchar();
        if (c == ' ' || c == ',' || c == '\t' || c == '\n') {
            break;
```

```
        } else if ((int) c == EOF) {
            end = 1;
            break;
        }
        add_char(s, c);
    }

    return end;
}

int reverse_string(string* s) {
    int result = UNDEFINED;
    string s1;
    result = init_string(&s1);
    result = copy_string(&s1, *s);
    for (uint64_t i = 0; i < s->last_element; ++i) {
        add_char(&s1, s->values[s->last_element - 1 - i]);
    }
    *s = s1;
    return result;
}
vitos@vitos-pop:~/Studying/mai/labs/l12$ cat string.h
#ifndef STRING_H

#define STRING_H

#include <stdio.h>
#include <stdlib.h>
#include <inttypes.h>


typedef struct string {
    char *values;
    uint64_t last_element;
    uint64_t memory_size;
} string;


// Initialize string with default capacity
int init_string(string* s);

// Adds memory
int string_resize(string* s, uint64_t capacity);

// Copies s2 to s1 (taking minimal memory size) -> s1='aboba', s2='bobr' -> s1='bobr '
int copy_string(string* s1, string s2);

// Adds a char to string, adding more memory if needed
int add_char(string* s, char value);

// Reads string char by char returning 1 if EOF
int read_string(string* s);

int reverse_string(string* s);

#endifvitos@vitos-pop:~/Studying/mai/labs/l12$ cat main.c
/* Лабораторная работа №12 вариант №30    */
/* Студент гр. М8О-103Б-22 Клименко В. М. */
#include <stdio.h>
#include <string.h>
#include "string.h"

void solve(string* s) {
    char t1, t2;
    reverse_string(s);
    for (unsigned long i = 0; i < s->last_element - s->last_element % 2; i += 2) {
        t1 = s->values[i];
        t2 = s->values[i + 1];
        if (t2 == '-') continue;
        if (t1 - '0' > t2 - '0') {
            s->values[i] = t2;
            s->values[i + 1] = t1;
        }
    }
    reverse_string(s);
}

int main() {
    int END = 0;
    string inp;

    init_string(&inp);

    while (1) {
        init_string(&inp);
```

```
        END = read_string(&inp);
        if (END) break;
        solve(&inp);
        printf("Number after processing: %s\n", inp.values);
    }

    printf("End of programm...\n");
    return 0;
}
vitos@vitos-pop:~/Studying/mai/labs/l12$ make
gcc -std=c99 -g -Wall -Wextra -o main main.c string.c
./main
123
Number after processing: 132
132
Number after processing: 132
1234
Number after processing: 2143
-100
Number after processing: -100
-101
Number after processing: -110
90955843
Number after processing: 90958543
98372450239847532094587230948572394805775484848475
Number after processing: 983725420938475329054873290847532948075758484848475
9283745983274474747474747
Number after processing: 98273549832747474747474
0
Number after processing: 0
1
Number after processing: 1
-1
Number after processing: -1
-211111111111111111111111111111111111111
Number after processing: -2111111111111111111111111111111111111111
-111111111111111111111111111111111111112
Number after processing: -1111111111111111111111111111111111111121
End of programm...
vitos@vitos-pop:~/Studying/mai/labs/l12$
```

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 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 выполнять действия над цифрами, представленными через тип char

Недочёты при выполнении задания могут быть устранены следующим образом: _____

Подпись студента _____