ГУАП

КАФЕДРА №14

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

Коренева Е.А

Должность, уч. степень, звание подпись, дата инициалы, фамилия

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

по курсу: ПРОГРАММИРОВАНИЕ НА ЯЗЫКАХ АССЕМБЛЕРА

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. 1441

подпись, дата инициалы, фамилия

Санкт-Петербург

2016

Оглавление

1. Постановка задачи..........................................................................3

2. Формализация задачи......................................................................3

3. Алгоритм........................................................................................4

4. Листинг..........................................................................................5

5. Источники......................................................................................7

1. Постановка задачи

Реализовать игру “виселица”, используя для строковых операций ассемблер x86

2. Формализация задачи

Программа должна принимать ввод игрока, выводить превдографическое изображение виселицы, отображающее текущее состояние игры и строку, состоящую из пробелов и угаданных игроком букв.

3. Алгоритм [2]

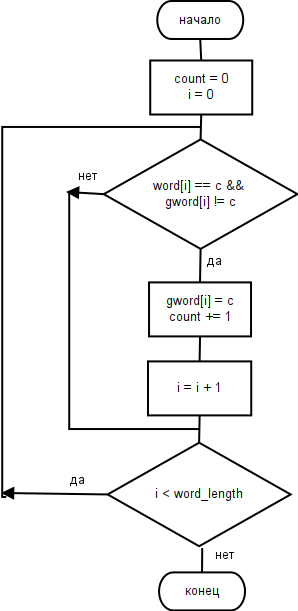


Рисунок 1. Схема алгоритма

4. Листинг [1]

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <ctype.h>

#include "text.h"

#define HISTORY\_LEN 16

**typedef** **enum** { GAME, WON, FAIL } state\_t;

**void** draw(**const** **char**\* str) {

printf("%s\n", str);

}

**char** ask\_user(**char**\* history) {

**static** **int** history\_index = 0;

**char** c;

printf("Enter letter: ");

**do** {

scanf("%c", &c);

} **while**(!isalpha(c));

printf("\n");

history[history\_index++ % (HISTORY\_LEN-1)] = c;

**return** c;

}

**int** update\_word(**const** **char**\* word, **char**\* game\_word, **char** c) {

**int** count = 0;

**asm** **\_\_volatile\_\_** (

" mov eax, 0;" // обнуляем счетчик букв

" mov rbx, %1;" // копируем адрес word

" mov rcx, %2;" // копируем адрес game\_word

"loop1:" // метка цикла

" cmp [rbx], %3;" // сравниваем букву по указателю word с введенной

" jne next; " // если не совпадает, переходим к следующей итерации цикла

" cmp [rcx], %3;" // сравниваем букву по указателб на game\_word с введенной

" je next; " // если совпадает (буква уже была введена), переходим к следующей итерации

" inc eax;" // увеличиваем смктчик угаданных букв

" movb [rcx], %3;" // копируем букву в game\_word по текущему указателю

" next:" // метка перехода к следующей итерации

" inc rbx;" // перемещаемся к следующей букве в word

" inc rcx;" // перемещаемся к следующей букве в game\_word

" cmpb [rbx], 0;" // сравниваем текущую букву с символом конца строки

" jne loop1;" // если не конец строки, переходим начало цикла

" mov %0, eax;" // копируем счетчик букв в переменную

: "=r" (count)

: "r" (word), "r" (game\_word), "r" (c)

: "rax", "rbx", "rcx"

);

**return** count;

}

**int** main(**int** argc, **char** \*argv[]) {

**const** **char**\* word = "assembler";

**char**\* game\_word = (**char**\*) malloc(word);

**char** history[HISTORY\_LEN];

**int** level;

// Заполняем историю и слово пробелами

memset(game\_word, '\_', strlen(word) \* **sizeof**(**char**));

memset(history, '\_', (HISTORY\_LEN - 1) \* **sizeof**(**char**));

state\_t state = GAME;

**char** c;

**int** changed\_letters;

**while**(state == GAME) {

system("clear");

fflush(stdin);

fflush(stdout);

draw(history);

draw(levels[level]);

draw(game\_word);

c = ask\_user(history);

changed\_letters = update\_word(word, game\_word, c);

**if**(changed\_letters == 0) {

level++;

}

**if**(strcmp(word, game\_word) == 0) {

state = WON;

}

**if**(level >= MAX\_LEVEL) {

state = FAIL;

}

}

**if**(state == WON) {

draw(you\_won);

} **else** {

draw(game\_over);

}

**return** 0;

}

5. Источники

1. x86 Assembly: <https://en.wikibooks.org/wiki/X86_Assembly>
2. ГОСТ 19.701-90