**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Основы информатики»

Отчет по домашнему заданию

«Работа со строками»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-13Б |  | преподаватель каф. ИУ5 |
| Поляков Данила |  | Аксёнова М.В. |
| Подпись и дата |  | Подпись и дата: |

Москва, 2021 г.

**Постановка задачи**

Слова текста из малых латинских букв записаны не менее чем через один пробел; текст оканчивается точкой. БЕЗ ИСПОЛЬЗОВАНИЯ конструкции STRING:

а) написать программу ввода такого текста с клавиатуры;

б) напечатать все слова, отличающиеся от последнего слова, и имеющие максимальную длину.

**Разработка алгоритма**

**Описание алгоритма**

Алгоритм ввода подходящего по условия текста работает следующим образом:

нажатая клавиша считывается в буфер, после чего проверяется на корректность и лишь затем выводится. Всё это происходит до момента ввода точки. Клавиша backspace реализована сдвигом каретки на одну позицию влево, вставкой пробела и снова возврата на позицию назад.

Для решения задачи б) во введённом массиве удаляются все пробелы в конце строки перед точкой для большего удобства работы с массивом. Сохраняется последнее слово. Реализована функция записи слова в отдельный массив current\_word. Проверку на то, подходит ли данное слово по условию выполняет функция isWordSuit. Если считанное слово подходит, оно выводится на экран.

**Описание используемых переменных**

int c – считываемый код символа

int max\_lenght – максимальная длина слова в введённом тексте

int \*last\_word – массив кодов символов, образующих последнее слово

int \*current\_word – массив кодов символов, образующих текущее проверяемое слово

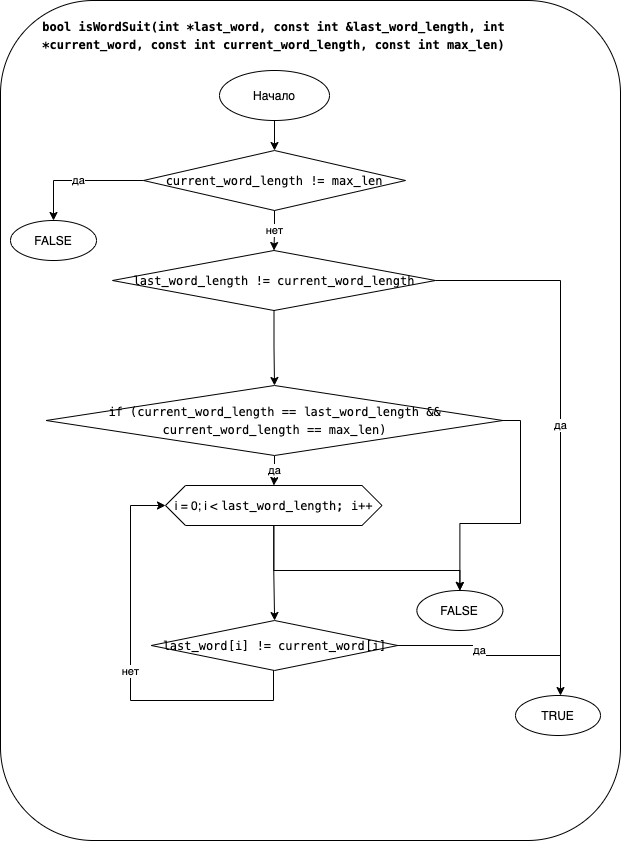
int n – номер элемента относительно начала строки

int \*arr - введённый текст

int kn, qn – переменная для расчёта позиции курсора;

int j = 0 - итератор для current\_word

int nCurrent – длина строки после удаления пробелов в конце

**Схема алгоритма**

**Текст программы**

**Листинг кода программы:**

#include <ncurses.h>

bool isWordSuit(int \*last\_word, const int last\_word\_length, int \*current\_word, const int current\_word\_length, const int max\_len)

{

if (current\_word\_length != max\_len)

{

return false;

}

if (last\_word\_length != current\_word\_length)

{

return true;

}

if (current\_word\_length == last\_word\_length && current\_word\_length == max\_len)

{

for (size\_t i = 0; i < last\_word\_length; i++)

{

if (last\_word[i] != current\_word[i])

{

return true;

}

}

}

return false;

}

int main()

{

initscr();

noecho();

int c;

int max\_lenght = 0;

int current\_length = 0;

int \*last\_word = new int[200];

int \*current\_word = new int[200];

int last\_word\_lenght;

int n = -1;

int \*arr = new int[400];

int kn = 0, qn = 0;

int j = 0;

while ((c = getch()) != '.')

{

if ((c > 96 && c < 123) || (c == 127) || (c == 32))

{

if (c > 96 && c < 123 || (c == 32))

{

echochar(c);

n++;

qn = n % 80;

kn = n / 80;

arr[n] = c;

}

if (c == 127 && n != -1)

{

arr[n] = -1;

move(kn, qn);

echochar(' ');

move(kn, qn);

n--;

qn = n % 80;

kn = n / 80;

}

}

}

echochar(c);

echochar('\n');

n++;

//clearing all whitespaces at the end;

int nCurrent = n;

while (arr[nCurrent - 1] == 32)

{

nCurrent--;

}

n = nCurrent;

n++;

arr[n - 1] = '.';

int z = 0;

for (z = n - 2; z>=0; z--)

{

if (arr[z]==' ')

{

break;

}

if (arr[z] != ' ')

{

last\_word\_lenght++;

}

else

{

break;

}

}

int qq = 0;

for (int i = z + 1; i < n - 1; i++)

{

last\_word[qq] = arr[i];

qq++;

}

for (size\_t i = 0; i < n-last\_word\_lenght; i++)

{

if (arr[i] == 32)

{

if (j > max\_lenght)

{

max\_lenght = j;

}

j = 0;

}

else

{

j++;

}

}

if (j > max\_lenght)

{

max\_lenght = j;

}

if (max\_lenght<last\_word\_lenght)

{

max\_lenght = last\_word\_lenght;

}

j = 0;

for (size\_t i = 0; i < n - last\_word\_lenght; i++)

{

if (arr[i] == 32)

{

if (isWordSuit(last\_word, last\_word\_lenght, current\_word, j, max\_lenght))

{

for (size\_t q = 0; q < j; q++)

{

echochar(current\_word[q]);

}

echochar(' ');

}

j = 0;

}

else

{

current\_word[j] = arr[i];

j++;

}

}

while (getch() != 'q')

{

}

delete[] arr;

delete[] last\_word;

delete[] current\_word;

endwin();

return 0;

}

**Анализ результатов**

