*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана***  ***(национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ \_\_\_\_\_ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА \_\_\_\_\_\_\_ Компьютерные Системы и сети (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по лабораторной работе № 5**

**Вариант 22**

**Дисциплина: машинно-зависимые языки и основы компиляции**

**Название лабораторной работы:** **Программирование с использованием разноязыковых модулей**

Студент гр. ИУ6-44  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шумаков А.А.**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Данилюк С.С.**

(Подпись, дата) (И.О. Фамилия)

Москва, 2018

**Цель работы**: изучение конвенций о способах передачи управления и данных при вызове из программы, написанной на языке высокого уровня, подпрограмм, написанных на ассемблере.

**Задание**

Дан текст не более 255 символов. Слова отделяются друг от друга пробелами. Удалить гласные буквы из слов с заданными номерами.

**Выполнение**

Приведем на рисунке 1 схему алгоритма программы, написанной на С++, а на рисунке 2 приведем схему алгоритма программы, написанной на ассемблере.



Рисунок 1: Схема алгоритма программы на C++



Рисунок 2 Схема алгоритма программы на ассемблере

1. Приведем код программы на C++:

#include "stdafx.h"

#include "cstring"

#include <conio.h>

#include <cstdio>

#include <iostream>

using namespace std;

extern "C" void \_\_cdecl GLASDEL(char \*str, int \*num, int len);

extern "C" void \_\_cdecl RESULT(char \*s) //Печать итоговой строчки

{

cout << "Передача управления обратно в C++\n"<< endl;

cout << "Полученная строка: " << s << endl;

}

int main()

{

setlocale(LC\_CTYPE, "Rus");

int slov = 1 ,k = 0; // slov = количество слов в строке, k = число введеных на удаление гласных слов

char str[255]; // строка

int help = 0; // вспомогательная переменная

bool check = true; // флаг правильности ввода номера слова на удаление гласных

cout << "Введите строку символов на английском языке\n";

cin.get(str,255);

cout << "Ваша строка\n";

cout << str << '\n';

cout << "длина строки: " << strlen(str) << '\n';

// Считаем количество слов

for (int i = 0; i < strlen(str) ; i++)

{

if (str[i] == ' ') slov++;

}

// Массив для хранения слов, в которых нужно удалить все гласные

int \*num = new int[slov];

cout << "слов: " << slov << endl;

cout << "Введите номера слов в которых хотите удалить гласные буквы\n";

cout << "Для окончания ввода введите 0\n";

for (int i = 0, a = 1; a != 0; i++)

{

cin >> a;

check = true;

for (int j = 0; j < k; j++) // Проверка на повторное вхождение в массив номеров слов

if (num[j] == a) check = false;

if (a != 0 && a <= slov && a > -1 && check!=false)

{

num[k] = a;

k++;

}

}

for (int i = 0; i < k-1; i++) // Сортировка номеров слов

for (int j = k-1 ; j > i; j--)

if (num[i] > num[j])

{

help = num[i];

num[i] = num[j];

num[j] = help;

}

cout << "Передача управления в ассемблер" << endl << endl;

GLASDEL(str, num, strlen(str));

cout << endl;

delete[] num;

system("pause");

return 0;

}

Б) Приведем код программы на ассемблере:

.586

.MODEL flat

.DATA

i dword 0 ; общее смещение

position dword 0 ; смещение на начало нового слова

current dword 0

glasn DB "aeiouy" ; строка с гласными

.CODE

RESULT proto c str2:ptr byte;Прототип внешней процедуры вывода результата

public \_GLASDEL;Объявление внутреннего имени процедуры

\_GLASDEL proc

;начало пролога

PUSH EBP

MOV EBP,ESP

PUSH EBX

PUSH EDI

PUSH ESI

;конец пролога

MOV EDI,[EBP+8] ;загружаем адрес строки в edi

MOV ESI,[EBP+12] ;загружаем адрес массива с номерами нужных слов

MOV ECX,[EBP+16] ; загружаем длину строки

MOV EBX,0 ;обнуляем счетчик номера слова

MOV EDX,0 ;обнуляем счетчик текущей длины слова

mov position,-1 ;position всегда будет указывать на начало нового слова

CLD

cic1: push EAX ; корректируем position на новое слово

dec i ; изначально в position установиться 0

mov EAX,i

inc i

mov position,EAX

inc position

pop EAX

CMP ECX,0 ;цикл с предусловием конец строки достигнут?

JE con ;если да, переход на вывод результатов

MOV EAX,' ' ;загружаем в al пробел для поиска

MOV EDX,0 ;обнуляем счетчик текущей длины слова

INC EBX ;увеличиваем счетчик номера слова

cic2: dec ECX ;уменьшаем счетчик оставшейся длины слова

push ECX

mov ECX,i

mov AL,[EDI+ECX] ; в i записываетс текущее смещение для всей строки

inc i

pop ECX

cmp AL,' ' ;проверяем очередной символ на пробел

JE consl ;если пробел - переходим на сравнение слова с номером

INC EDX ;если нет - увеличиваем длину слова

CMP ECX,0 ;конец строки достигнут?

JE consl2 ;если да, проверяем последнее слово

JMP cic2 ;переход на следующую итерацию цикла поиска пробела

consl2: DEC EDX;уменьшение счетчика длины слова (частный случай)

consl:

CMP [ESI],EBX ;сравниваем номер слова с номером в массиве

JE prod ;если одинаковы - начинаем удалять гласные

JMP cic1

prod:;подготовка к удалению гласных

push ESI

PUSH EBX

push EDX

push ECX

mov ECX,0

dec i ; в i смещение для конца слова

mov EDX,position ; в EDX смещение для начала нового слова

mov ESI,EDI ; В ESI загружаем исходную строку

add ESI,EDX ; Смещаем ESI для на начало нового слова

dec EDX

kont: mov EBX,0 ; в EBX смещение для поиска гласной

inc EDX ; в строке glasn

mov AL,[glasn+EBX] ; в AL первую гласную из строки glasn

cmp EDX,i ; проверка на последнее слово и символ

je prepare ;Если истинна то выходим из удаления

cycle1:

cmp AL,[EDI+EDX] ; сравнение текущего символа с гласной

je glas ; Если гласная то переходим непосредственно к удалению

cmp EDX,i ; проверка на последнее слово и символ

je prepare

cmp EBX,5 ; если достигнут конец в строке glasn

je kont ; начинаем сравнения со слудующим символом в EDI

cycle2: inc EBX ; увеличиваем смещение в строке glasn

repcycle: mov AL,[glasn+EBX] ; В AL новая гласная для сравнения с текущим символом EDI

jmp cycle1 ; повторяем условие сравнения с гласной

glas:

mov AL,[EDI+EDX] ; в AL текущий символ

cmp AL,[glasn+EBX] ; повторно сравниваем с гласной

je glasafter ; если гласная то удаляем

dec i ; если конец слова то выходим

cmp i,EDX

je prepare

mov [EDI+EDX],AL ; если не гласная оставлям символ в EDI

cmp EDX,i ; проверка на конец слова

je prepare

inc i

jmp glas ; возвращаемся на метку glas

glasafter: ; удаление и проверка на подряд идущие гласные

cmp EDX,i

je prepare

mov AL,' '

mov [EDI+EDX],AL ; помещаем пробел вместо гласной

mov EBX,0 ; смещение на начало строки glasn

inc EDX ; увеличиваем смещение для EDI

jmp repcycle ; повтоорно ищем следующую гласную

prepare:

mov AL,[EDI+EDX]

cmp AL,[glasn+EBX] ; переход на последнюю замену

je lastzam

cmp EBX,5 ; если конец в строки glasn, то выходим

je final

inc EBX ; переходим к следующей гласной

jmp prepare ; сравниваем с новой гласной

lastzam:

mov AL,' ' ; удаляем гласную

mov [EDI+EDX],AL

final: pop ECX ; подготовка к выходу из удаления

pop EDX

pop EBX

pop ESI

add ESI,4 ; В ESI номер слова, в котором следует удалить гласные

inc i

jmp cic1 ; возвращаемся на поиск конца нового слова

odin: mov position,0 ; Если ввели один символ

jmp prod

con:;Передача управления в С++ и эпилог

cmp position,1 ; Доп проверка на гласную

je odin ; если слово всего одно

Invoke RESULT,[EBP+8]

POP ESI

POP EDI

POP EBX

POP EBP

ret

\_GLASDEL endp

End

На рисунке 3 приведены результаты выполнения программы

**Тесты программы**

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| abcd qwerty laba five is very hard  Введенные номера слов: 1 2 3 4 5 7 | bcd qw rt l b five s very hard | bcd qw rt l b five s very hard |
| delal labu five days kursach teper nado delet  Введенные номера слов: 1 5 8 | d l l labu five days k rs ch teper nado d l t | d l l labu five days k rs ch teper nado d l t |
| assembker I am never foget you  Введенные номера слов: 2 3 6 | assembler m never foget | assembler m never foget |

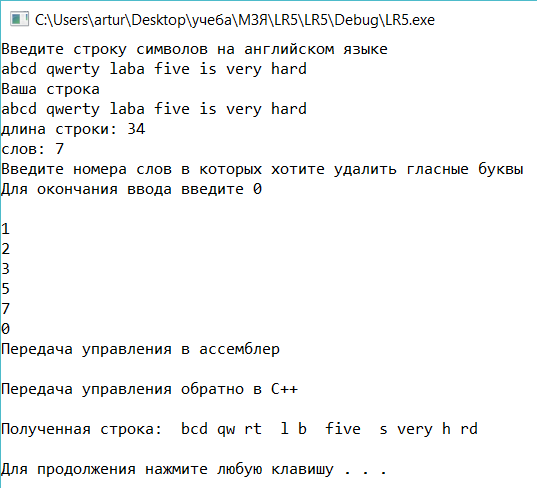
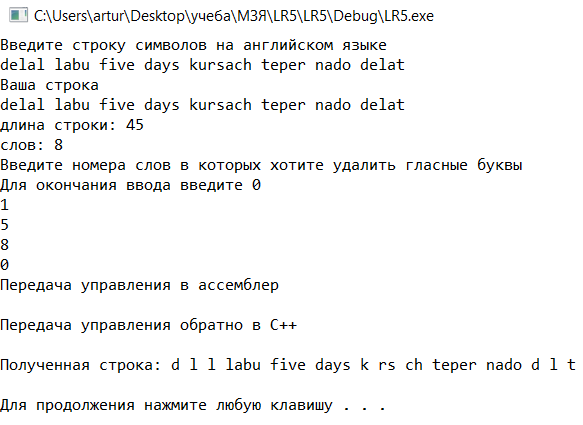
 

Рисунок 3: Результаты выполнения программы

На рисунке 4 приведен вид стека в момент передачи управления.



Рисунок 4: Стек в момент передачи управления

**Выводы**

* Была разработана программа, выполняющая данное задание, которая корректно обрабатывает строки длинной до 255 символов.
* Было изучено программирование процедур на языке ассемблера и связывание разноязыковых модулей.