|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования FPMI_ngtu_neti_rgb_polya«Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 3 | | |
| по дисциплине «Языки программирования и методы трансляции» | | |
| **ИНТЕРФЕЙС С ЯЗЫКОМ С++** | | |
|  | | |
|  | Группа ПМ-04 |  |
| Бригада 11 | Плешкова Дарья |
|  | Сальников дмитрий |
|  |  |
|  |  |
|  |  |
| Преподаватель | сивак мария алексеевна |
|  | петров роман владимирович |
| Новосибирск, 2022 | | |

**1. Задание**

Написать программу, удаляющую точки в конце исходной строки. Для вызова процедуры использовать соглашение о вызовах stdcall.

**2. Алгоритм**

1. Получаем адрес начала строки и количество символов строки
2. Записываем в счетчик цикла длину строки
3. В цикле сравниваем элементы строки с точкой (проход осуществляется с конца)

3.1. Если текущий элемент является точкой, приравниваем его к символу конца строки, переходим к следующему символу.

3.2. Если текущий элемент не является точкой, выходим из цикла

1. Выходим из подпрограммы, переходя к адресу возврата

**3. Программные средства**

* PUSH – команда «положить в стек»
* CALL – команда вызова внешней функции
* MOV – команда, пересылающая байт, слово или двойное слово между регистром и ячейкой памяти или между двумя регистрами
* CMP – команда условного сравнения
* JE – команда перехода по условию «приемник равен источнику»
* JMP – команда перехода без условия
* ADD – команда, помещающая сумму приемника и источника в приемник
* DEC – команда, вычитающая единицу из регистра или ячейки памяти
* POP – команда «взять из стека»
* RET – команда завершения процедуры
* LOOP – команда повторения цикла до конца счетчика

**4. Тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Исходные данные** | **Результат** | **Комментарий** |
| **1.** | Hello! | Hello! | Строка с символами и без точек |
| **2.** | Hello!... | Hello! | Строка с символами и точками |
| **3.** | … | (пустая строка) | Строка без символов и с точками |
| **4.** | (пустая строка) | (пустая строка) | Пустая строка |
| **5.** | He…llo!... | He…llo! | Строка с точками в середине |

**5. Текст программы**

**Source.cpp**

#include <iostream>

#include <string>

using namespace std;

extern "C" void \_\_stdcall DeleteDots(const char\* str, uint16\_t lenght);

int main() {

string str;

cout << "Enter the string: ";

getline(cin, str);

if (str.length()) DeleteDots(str.c\_str(), str.length());

cout << "Result:\t\t " << str;

}

**asm.asm**

*.386*

.MODEL FLAT

.CODE

\_DeleteDots@8 PROC ; СТЕК: [Старое значение EBP 4][Адрес возврата 4][Адрес первого элемента строки 4][длина строки 4]

PUSH EBP ; Сохраняем старое значение EBP

MOV EBP, ESP ; EBP = старое значение EBP

ADD EBP, 8 ; EBP = адрес первого элемента строки

; [EBP] + 4 = длина строки

MOV EDI, [EBP] + 4 ; EDI = индекс последнего символа (длина строки)

DEC EDI ; Вычитаем единицу (нумерация идет с 0)

MOV ECX, [EBP] + 4 ; Счетчик цикла = длина строки

ForEachChar: ; Метка начала цикла

MOV EAX, [EBP] ; EAX = адрес начала строки

MOV AL, [EAX + EDI] ; AL = текущий элемент(адрес начала строки + индекс элемента)

CMP AL, '.' ; Сравниваем текущий элемент с точкой

JNE Next ; Если не точка, перейти на метку Next

CALL ShiftStr ; Вызываем функцию, приравнивающую текущий элемент к символу конца строки

DEC EDI ; Передвигаемся влево по строке на один элемент

LOOP ForEachChar ; Просматриваем строку с конца до появления первого элемента, не являющегося точкой

Next:

POP EBP ; Возвращаем старое значение EBP

RET 8 ; Возвращаемся в модуль С++ с переходом к адресу возврата

\_DeleteDots@8 ENDP

ShiftStr PROC

PUSH ECX ; Помещаем в стек счетчик цикла

PUSH EAX ; Помещаем в стек адрес текущего символа

PUSH EDX ; Помещаем в стек следующий символ (в DL)

MOV ECX, [EBP] + 4 ; ECX = длина строки

MOV EAX, [EBP] ; EAX = адрес первого символа строки

ADD EAX, EDI ; EAX = адрес текущего символа

MOV DL, [EAX] + 1 ; DL = адрес следующего символа

MOV [EAX], DL ; Текущий символ = следующий символ (т.е. становится символом окончания строки)

POP EDX

POP EAX ; Возвращаем старые значения из стека

POP ECX

RET ; Выходим из подпрограммы

ShiftStr ENDP

END