МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждения высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ

**ЛАБОРАТОРНАЯ РАБОТА № 4**

по дисциплине

**«Машинно-ориентированное программирование»**

на тему:

**«Обработка символьной информации»**

*Вариант № 3*

Выполнил:

Студент группы

КТбо2-8

|  |  |  |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Жалнин Д. И. |
|  | *подпись* |  |
|  |  |  |

Проверил:

ассистент кафедры

МОП ЭВМ

|  |  |  |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гуляев Н. А. |
|  | *подпись* |  |

Оценка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Таганрог 2020

# **1 ЦЕЛЬ РАБОТЫ**

* 1. **Дидактическая цель работы**

Ознакомление с методами составления программ на языке «Ассемблер», использующих ввод/вывод информации в консоли пользователя, а также с методами обработки этой информации.

**1.2 Практическая цель работы**

В рамках лабораторной работы необходимо разработать программу на языке ассемблера, алгоритм которой выполняет задачу согласно описанному индивидуальному заданию, скомпилировать и запустить код программы с помощью программного пакета «TASM».

# **2 ВАРИАНТ ЗАДАНИЯ**

**2.1 Общие требования**

Для всех вариантов требуется выполнить разработку программного модуля при помощи СРПО «Turbo Assembler», реализующего некоторую обработку массива символов, введенных из консоли, а так же вывод результата работы.

**2.2 Индивидуальное задание, вариант № 3**

Ввести с клавиатуры строку, состоящую из целых чисел, разделенных пробелами. Найти количество четных чисел.

# **3 ХОД РАБОТЫ**

**3.1 Описание высокоуровневой реализации**

В ход работы была составлена программа на языке программирования «Ассемблер», используя модель памяти «**small**», в котором допускается наличие одного сегмента кода и единственного сегмента данных. Размер стека программы – 256 байт.

Перед началом основного сегмента кода происходит инициализация структур данных, необходимых для работы программы:

**Counter db 0 ; Счетчик четных чисел**

**Number db 0 ; Переменная для временного хранения каждого из чисел в ; процессе работы**

**Inbuff db 80, ?, 82 dup(?) ; Буфер для ввода символов из консоли**

**; Максимальное количество символов – 80**

**OutBuff db 0Ah, 0Dh, ?, ?, 0Dh, 0Ah, '$'**

**; Буфер для вывода количества найденных чисел в консоль.**

**; Максимальным может быть 2х-символьное число, поэтому для его вывода ; выделено два байта**

Алгоритм работы программы:

**codeseg**

**startupcode**

**Begin:**

**lea DX, Inbuff**

**mov AH, 0Ah**

**int 21h**

**mov CL, Inbuff+1; count of symbols**

**lea BX, Inbuff+2; adress of beggining**

**jmp Loop1**

**; Происходит считывание строки с числами с помощью прерывания 21h. Адрес на начало**

**; массива передается в**

**; регистр BX. Количество символов в регистр CL. Следующий шаг в «Loop1»**

**Mult:**

**mov AL, 10**

**mul Number**

**mov Number, AL**

**jmp Loop10**

**; Происходит умножение числа на 10, для добавления новой цифры в число**

**Loop1:**

**mov AL, [BX]**

**add BL, 1**

**cmp AL, ' '**

**je NotDigit**

**sub AL, '0'**

**push AX**

**jmp Mult**

**; Происходит распознавание введенного символа – символ является пробелом или цифрой.**

**; Если число является цифрой, переходим в «Mult» и продолжаем считывание**

**; Если число – пробел, переходим в «NotDigit» и определяем четность числа**

**Loop10:**

**pop AX**

**add Number, AL**

**loop Loop1**

**NotDigit:**

**cmp Number, 0**

**je NotDigit1**

**test Number, 1**

**jnz Odd**

**mov AL, 1**

**inc Counter**

**; При вводе пробела, проверяется последний бит числа. Если он не равен единице, в**

**; счетчик прибавляется единица**

**Odd:**

**mov Number, 0**

**; Число сбрасывается в 0**

**NotDigit1:**

**cmp CL, 0**

**je Print**

**loop Loop1**

**; Если программа прошла по всем символам строки, идет переход к выводу**

**; Иначе цикл продолжит свою работу**

**Print:**

**xor AX, AX**

**mov AL, Counter**

**mov BL, 10**

**div BL**

**add AH, '0'**

**add AL, '0'**

**cmp AL, 0**

**mov OutBuff+2, AL**

**mov OutBuff+3, AH**

**lea DX, OutBuff**

**mov AH, 09h**

**int 21h**

**; В массив вывода переносим два байта с первой и второй цифрой счетчика**

**; Вызываем прерывание 21h для вывода в консоль буфера вывода**

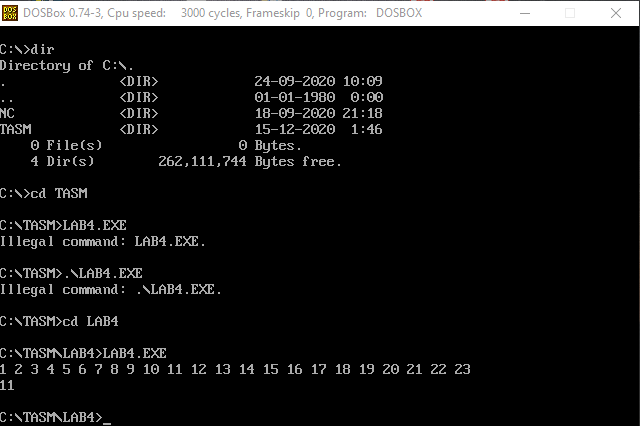
**QUIT:**

**exitcode 0**

**end**

**3.4 Описание полученных результатов**

Программный модуль был скомпилирован, запущен и отлажен в рамках среды «DOS BOX». При вызове «TASM» были заданы ключи «-L -ZI», которые позволили получить отладочные файлы. При вызове «TLINK» были использованы ключи «-V». С целью отладки был запущен отладчик «TD», в котором было проведено пошаговое исполнение программы. На *рисунках 1 и 2* можно наблюдать результат выполнения программы. Результат работы программы совпал с ожидаемым – программа отработала корректно.

**

*Рисунок 1 – результат работы программы*

# **4 ВЫВОДЫ**

**4.1 Полученные знания, навыки, умения**

В ходе выполнения лабораторной работы была разработан и отлажен программный модуль, который считает некоторое количество чисел из консоли, определяет количество четных и выводит результат работы в консоль.