МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

(КНИТУ-КАИ)

Институт компьютерных технологий и защиты информации\_\_\_

(наименование института (факультета), филиала)

\_ \_Кафедра Прикладной Математики и Информатики \_

(наименование кафедры)

\_ \_ «Программная инженерия» \_\_ \_

(наименование направления подготовки (специальности))

Лабораторная работа № 2

По дисциплине: Низкоуровневое программирование

Тема: «Побитовая обработка на языке Ассемблера»

Выполнил: студент группы 4311   
Наумов Денис Олегович

Проверил: доцент кафедры ПМИ   
Трегубов Владимир Михайлович

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

*(подпись)*

Казань, 2025 год

**Задание**

Вводится строка, состоящая из 20 символов. Вывести на экран шестнадцатеричное представление символов строки с 5 по 18 через 1.

**Листинг**

STACKSG     SEGMENT PARA STACK      *; Сегмент стека*

    DB 64 DUP(?)                    *; 64 байта для стека*

STACKSG     ENDS

DATASG      SEGMENT PARA 'DATA'     *; Сегмент данных*

    MySTR     DB 21, 0, 21 DUP(?)  *; Буфер ввода: макс.длина, факт.длина, данные*

    MSG1      DB 'Enter the string: ', '$'  *; Приглашение ввода*

    MSG2      DB 0Dh, 0Ah, 'Result: ', 0Dh, 0Ah, '$'  *; Заголовок результата*

    CHAR\_MSG  DB ' - $'             *; Разделитель*

    NEWLINE   DB 0Dh, 0Ah, '$'      *; Перевод строки*

DATASG      ENDS

CODESG      SEGMENT PARA 'CODE'     *; Сегмент кода*

    ASSUME  CS:CODESG, DS:DATASG, SS:STACKSG  *; Назначение сегментных регистров*

ENTRY   PROC FAR                    *; Главная процедура*

        PUSH DS                     *; Сохраняем DS*

        SUB AX, AX                  *; AX = 0*

        PUSH AX

        MOV AX, DATASG              *; Загружаем адрес данных*

        MOV DS, AX                  *; Устанавливаем DS*

        MOV AH, 09h                 *; Функция вывода строки*

        LEA DX, MSG1                *; Адрес приглашения*

        INT 21h                     *; Выводим "Enter the string: "*

        MOV AH, 0Ah                 *; Функция ввода строки*

        LEA DX, MySTR               *; Адрес буфера*

        INT 21h                     *; Вводим строку*

        MOV BL, MySTR+1             *; Фактическая длина строки*

        CMP BL, 20                  *; Проверяем длину >= 20*

        JB  Exit                    *; Если меньше - выход*

        MOV AH, 09h                 *; Функция вывода строки*

        LEA DX, MSG2                *; Адрес заголовка*

        INT 21h                     *; Выводим "Result: "*

        LEA SI, MySTR+2             *; Начало строки*

        ADD SI, 4                   *; Переходим к 5-му символу*

        MOV CX, 7                   *; Обработаем 7 символов*

ProcessLoop:

        MOV AL, [SI]                *; Текущий символ*

        MOV DL, AL                  *; Вывод символа*

        MOV AH, 02h                 *; Функция вывода символа*

        INT 21h                     *; Выводим символ*

        PUSH AX                     *; Сохраняем символ*

        MOV AH, 09h                 *; Функция вывода строки*

        LEA DX, CHAR\_MSG            *; Разделитель " - "*

        INT 21h                     *; Выводим разделитель*

        POP AX                      *; Восстанавливаем символ*

        CALL PrintHex               *; Выводим HEX-представление*

        PUSH AX                     *; Сохраняем символ*

        MOV AH, 09h                 *; Функция вывода строки*

        LEA DX, NEWLINE             *; Перевод строки*

        INT 21h                     *; Переходим на новую строку*

        POP AX                      *; Восстанавливаем символ*

        ADD SI, 2                   *; Следующий символ (через один)*

        LOOP ProcessLoop            *; Повторяем цикл*

Exit:

        MOV AH, 4Ch                 *; Функция завершения*

        INT 21h                     *; Выход в DOS*

        RET                         *; Возврат*

ENTRY   ENDP

PrintHex PROC                       *; Вывод байта в HEX*

        PUSH CX                     *; Сохраняем CX*

        PUSH AX                     *; Сохраняем AX*

        MOV CL, 4                   *; Сдвиг на 4 бита*

        SHR AL, CL                  *; Старшая тетрада в младшие биты*

        CALL PrintDigit             *; Выводим старшую цифру*

        POP AX                      *; Восстанавливаем символ*

        PUSH AX                     *; Сохраняем снова*

        AND AL, 0Fh                 *; Младшая тетрада*

        CALL PrintDigit             *; Выводим младшую цифру*

        POP AX                      *; Восстанавливаем AX*

        POP CX                      *; Восстанавливаем CX*

        RET                         *; Возврат*

PrintHex ENDP

PrintDigit PROC                     *; Вывод одной HEX-цифры*

        CMP AL, 9                   *; Сравниваем с 9*

        JLE IsDigit                 *; Если <=9 - цифра*

        ADD AL, 'A' - 10            *; Преобразуем 10-15 в 'A'-'F'*

        JMP PrintChar               *; Переход к выводу*

IsDigit:

        ADD AL, '0'                 *; Преобразуем 0-9 в '0'-'9'*

PrintChar:

        MOV DL, AL                  *; Символ для вывода*

        MOV AH, 02h                 *; Функция вывода символа*

        INT 21h                     *; Выводим цифру*

        RET                         *; Возврат*

PrintDigit ENDP

CODESG ENDS

END ENTRY

**Пример работы программы**

****