МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

Дисциплина «МОП»

**Лабораторная работа №2**

Разработка программ обработки текстов и целых чисел.

**Вариант 5**

Выполнил

студент группы ИВТВМбд-31

Захарычев Н. А.

Проверил

ст. преподаватель кафедры ВТ

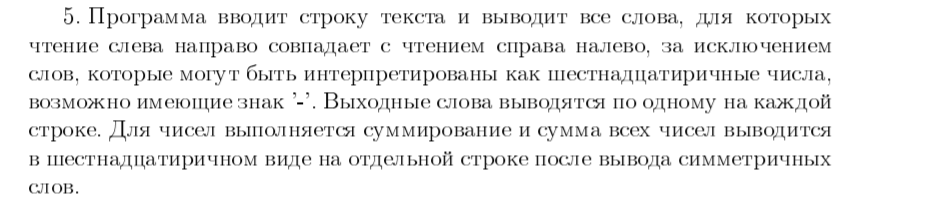
Лылова А. В.

**г.Ульяновск 2017**

**Цель работы:**

Изучение методов представления текстовой информации и целых чисел различной разрядности в программах на языке ассемблера и памяти машины. Освоение программно-технических приемов организации доступа к стокам, их частям и совокупностям. Приобретение умения программировать на ассемблере обработку текстовой информации и ввод-вывод целых чисел в различных системах счисления.

**Задание по варианту:**



**Ход работы:**

**Анализ исходных модулей лабораторной работы**

Назначения файлов:

*mop21.asm –* основной код программы

*mopl2l.asm –* перечень процедур для обработки данных

*mopl2.inc –* перечень констант для работы программы

*mopl2.mac –* переченьмакросов

Назначение процедур файла *mopl2l.asm*

PUTSS – вывод на экран строки с задержкой по времени между символами

PUTC – вывод символа на терминал

GETCH – ввод символа с терминала

GETS – ввод строки в буфер

SLEN – определение длины строки

UBINARY – преобразование числа в беззнаковое, десятичное

DILAY – выполнение задержки при выполнении программы.

Назначение макросов файла *mop21.mac*

PUTL – вывод строки на экран

PUTLS – вывод строки с задержкой

PUTLSC – вывод строки с задержкой и выравниванием по центру

FIXTIME – макрос фиксации начала измерения времени

DURAT – макрос окончания измерения времени и получения числа тактов

EXIT – завершение работы программы

**Представление объектов задачи в памяти**

**MAS\_STR DB 83 DUP(0*)***– Входная строка

**MAS\_LIT DB 'q', 'w', 'e', 'r', 't', 'y', 'u', 'i', 'o', 'p', 'a', 's', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'z', 'x', 'c', 'v', 'b', 'n', 'm'**– Массив символов(букв)

**MAS\_LIT\_OCT DB 161, 167, 145, 162, 164, 171, 165, 151, 157, 160, 141, 163, 144, 146, 147, 150, 152, 153, 154, 172, 170,143, 166, 142, 156,155**– Массив кодов символов(букв) в восьмиричном виде

**MAS\_LIT\_COUNT DB 26 DUP(0)** – Массив количества каждого символа во входной строке

**MAS\_STR\_LEN DW ?** – Количество символов входной строки

**CURR\_OCT DD 0** – Очередное восьмиричное число

**SUM\_OCT DD 0** – Итоговая сумма восьмиричных чисел

**FACT\_OCT DW 8** – Множитель для манипуляции с восьмиричными числами

**DIVIDER DB 10** – Множитель для манипуляции с десятичными числами

**COUNT\_OCT\_O\_CURR DB 0** – Количество восьмиричных чисел в строке

**Посимвольный ввод строки:**

GETS PROC NEAR

PUSH SI

MOV SI, DX

MOV AH, FUGETS

INT DOSFU

; прописать байт 0 в конец строки

XOR AH, AH

MOV AL, [SI+1]

ADD SI, AX

MOV BYTE PTR [SI+2], 0

POP SI

RET

GETS ENDP

**Проверки строки на слова и цифры:**

CHECK\_STR:

CMP STR\_[BX], ' '

JE CHECK\_WORD

CMP STR\_[BX], 0

JE CHECK\_WORD

MOV DL, STR\_[BX]

MOV WORD\_[DI], DL

INC DI

INC BX

JMP CHECK\_STR

;-----------------

;Проверка всех слов

;-----------------

CHECK\_WORD:

INC BX

MOV SI, 0

;разбор слова

DEC DI

CMP WORD\_[DI], 'h'

JNE WORDS

CMP WORD\_[0], 'h'

JE WORDS

JMP CHECK\_NUMBER

**Текущая посимвольная проверка слов и цифр:**

WORDS:

;-----------------

;Текущая проверка слова

;-----------------

NEXT\_SYMB:

CMP SI, DI

JGE PRINT\_POL

MOV DL, WORD\_[SI]

CMP DL, WORD\_[DI]

JNE NEXT\_WORD

INC SI

DEC DI

JMP NEXT\_SYMB

;-----------------

;Текущая проверка числа

;-----------------

CHECK\_NUMBER:

MOV DX, DI ;длинна числа

MOV LEN, DI

NEXT\_NUMBER:

CLD

XOR AX, AX

MOV CX, ARR\_H\_LEN

MOV DI, 0

MOV AL, WORD\_[SI]

;Посимвольно проверим

;-----------------

@L:

CMP ARR\_H[DI], AL

JE NEXT

INC DI

LOOP @L

JMP WORDS

;-----------------

NEXT:

INC SI

DEC DX

JNZ NEXT\_NUMBER

MOV LEN, SI

MOV WORD\_[SI], 0

;-----------------

;Перевод в число

MOV SI, 0

MOV CL, 4H

XOR DX, DX

MOV AX, WORD PTR WORD\_[SI]

MOV DI, LEN

TO\_NUMBER:

;развернуть слово

SUB AL, 30H

CMP AL, 9H

JLE PRINT\_

SUB AL, 7H

CMP AL, 0FH

JLE PRINT\_

SUB AX, 20H

**Вывод палиндромов и проверки условий на выход, преобразование цифр:**

;-----------------

;Вывод

;-----------------

PRINT\_:

SHL DX, CL

ADD DL, AL

SHR AX, CL

SHR AX, CL

DEC DI

JNZ TO\_NUMBER

MOV DI, I

MOV ARR[DI], DL

ADD I, 1

JMP NEXT\_WORD

;-----------------

;Вывод всех палиндромов

;-----------------

PRINT\_POL:

ADD COUNT\_POL, 1

PUTLS WORD\_

NEXT\_WORD:

MOV LEN, 0

MOV DI, 0

CALL CLR\_WORD\_

CMP STR\_[BX], 0

JE @@E

JMP CHECK\_STR

@@E:

CMP COUNT\_POL, 0

JNE @N

PUTL NO\_POL

@N:

XOR CX, CX

XOR SI, SI

XOR AX, AX

MOV CX, I

CMP CX, 0

JE NO\_PRINT

@C:

MOV BL, ARR[SI]

ADD AX, BX

INC SI

LOOP @C

MOV SUM, AL

;-------------------------

MOV BL, AL

CMP AL, 0FH

JLE @JR

SHR AL, 4H

CMP AL, 9H

JLE PR1

ADD AL, 7H

PR1:

ADD AL, 30H

CALL PUTC

MOV AL, BL

AND AL, 0FH

@JR:

CMP AL, 9H

JLE PR2

ADD AL, 7H

PR2:

ADD AL, 30H

CALL PUTC

MOV AL, HEX

CALL PUTC

NO\_PRINT: PUTL EMPTYS

EXIT

EXTRN PUTSS: NEAR

EXTRN PUTC: NEAR

EXTRN GETCH: NEAR

EXTRN GETS: NEAR

EXTRN SLEN: NEAR

EXTRN UTOA10: NEAR

CLR\_WORD\_ PROC NEAR

MOV CX, 80

PUSH SI

MOV SI, 0

CLR:

MOV WORD\_[SI], 0

INC SI

LOOP CLR

POP SI

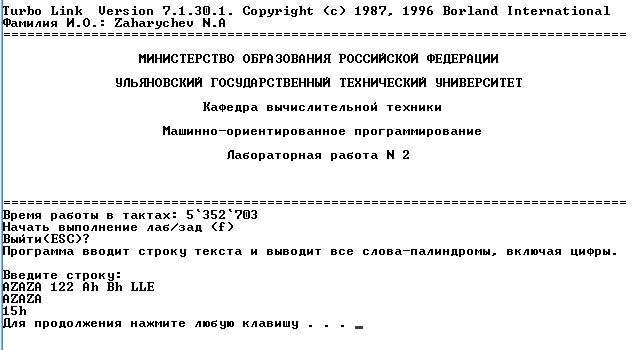
RET

CLR\_WORD\_ ENDP

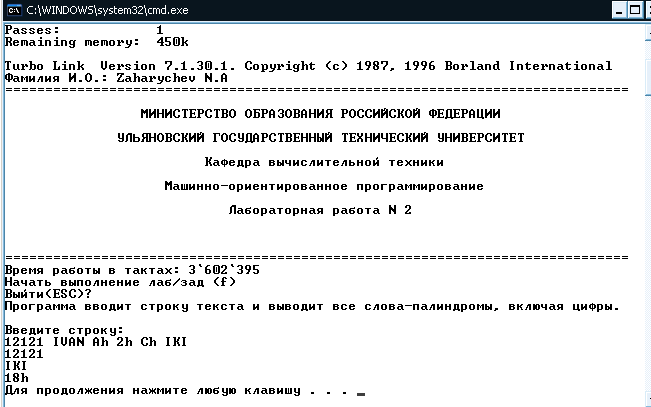
END BEGIN

**Контрольные примеры**

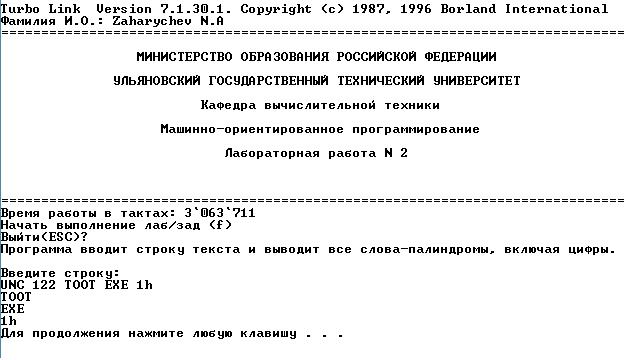
**Пример №1**



**Пример №2**



**Пример №3**



**Исходный код программы:**

mopl2.asm

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*; MOPL1.ASM - учебный пример для выполнения*

*; лабораторной работы N1 по машинно-ориентированному программированию*

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*.MODEL SMALL*

*.STACK 200h*

*.386*

*; Используются декларации констант и макросов*

*INCLUDE MOPL2.INC*

*INCLUDE MOPL2.MAC*

*; Декларации данных*

*.DATA*

*SLINE DB 78 DUP (CHSEP), 0*

*REQ DB "Фамилия И.О.: ",0FFh*

*MINIS DB "МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ",0*

*ULSTU DB "УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ",0*

*DEPT DB "Кафедра вычислительной техники",0*

*MOP DB "Машинно-ориентированное программирование",0*

*LABR DB "Лабораторная работа N 2",0*

*REQ1 DB "Начать выполнение лаб/зад (f)", 0*

*;------------- Новые переменные ------------------------------------------------------------------*

*REQ2 DB "Выйти(ESC)?", 0FFh*

*;-------------------------------------------------------------------------------------------------*

*TASK\_TEXT DB "Программа вводит строку текста и выводит все слова-палиндромы, включая цифры.", 0*

*INPUT\_STR DB "Введите строку: ", 0*

*OUTPUT\_STR DB "Результат: ", 0*

*NO\_POL DB "Полиндромов нет", 0*

*HEX DB "h",0ffh*

*SUM DB 0 ;сумма чисел*

*ARR DB 100 DUP (0)*

*ARR\_H DB 'ABCDEFabcdef0123456789', 0 ; массив HEX*

*ARR\_H\_LEN DW $-ARR\_H ; длина массива*

*WORD\_ DB 80 DUP(0) ;массив слов*

*COUNT\_POL DB 0 ;кол-во полиндромов*

*MASL DB 0 ;кол-во шестн. чисел*

*LEN DW 0 ;ДЛИНА ЧИСЛА*

*I DW 0*

*EMPTYS DB 0*

*BUFLEN = 70*

*BUFLEN1 = 50*

*BUF DB BUFLEN*

*LENS DB ?*

*STR\_ DB BUFLEN DUP (0)*

*PAUSE DW 0, 0 ; младшее и старшее слова задержки при выводе строки*

*TI DB LENNUM+LENNUM/2 DUP(?), 0 ; строка вывода числа тактов*

*; запас для разделительных "`"*

*SNAME DB BUFLEN1 DUP (0)*

*TACTS DB "Время работы в тактах: ",0FFh*

*;========================= Программа =========================*

*.CODE*

*; Макрос заполнения строки LINE от позиции POS содержимым CNT объектов,*

*; адресуемых адресом ADR при ширине поля вывода WFLD*

*BEGIN LABEL NEAR*

*; инициализация сегментного регистра*

*MOV AX, @DATA*

*MOV DS, AX*

*; инициализация задержки*

*MOV PAUSE, PAUSE\_L*

*MOV PAUSE+2,PAUSE\_H*

*PUTLS REQ ; запрос имени*

*; ввод имени*

*LEA DX, BUF*

*CALL GETS*

*@@L: ; циклический процесс повторения вывода заставки*

*; вывод заставки*

*; ИЗМЕРЕНИЕ ВРЕМЕНИ НАЧАТЬ ЗДЕСЬ*

*FIXTIME*

*PUTL EMPTYS*

*PUTL SLINE ; разделительная черта*

*PUTL EMPTYS*

*PUTLSC MINIS ; первая*

*PUTL EMPTYS*

*PUTLSC ULSTU ; и*

*PUTL EMPTYS*

*PUTLSC DEPT ; последующие*

*PUTL EMPTYS*

*PUTLSC MOP ; строки*

*PUTL EMPTYS*

*PUTLSC LABR ; заставки*

*PUTL EMPTYS*

*; приветствие*

*PUTLSC SNAME ; ФИО студента*

*PUTL EMPTYS*

*; разделительная черта*

*PUTL SLINE*

*; ИЗМЕРЕНИЕ ВРЕМЕНИ ЗАКОНЧИТЬ ЗДЕСЬ*

*DURAT ; подсчет затраченного времени*

*; Преобразование числа тиков в строку и вывод*

*LEA DI, TI*

*CALL UTOA10*

*PUTL TACTS*

*PUTL TI ; вывод числа тактов*

*; обработка команды*

*PUTL REQ1*

*;------Вывод своих строк с действиями -------------------*

*PUTL REQ2*

*;--------------------------------------------------------*

*CALL GETCH*

*PUTL EMPTYS*

*PUTLS TASK\_TEXT*

*PUTL EMPTYS*

*PUTLS INPUT\_STR*

*LEA DX, BUF*

*CALL GETS*

*PUTL EMPTYS*

*XOR AX, AX*

*MOV BX, 0*

*MOV DI, 0 ;длина слова*

*;-----------------*

*;Проверка строки*

*;-----------------*

*CHECK\_STR:*

*CMP STR\_[BX], ' '*

*JE CHECK\_WORD*

*CMP STR\_[BX], 0*

*JE CHECK\_WORD*

*MOV DL, STR\_[BX]*

*MOV WORD\_[DI], DL*

*INC DI*

*INC BX*

*JMP CHECK\_STR*

*;-----------------*

*;Проверка всех слов*

*;-----------------*

*CHECK\_WORD:*

*INC BX*

*MOV SI, 0*

*;разбор слова*

*DEC DI*

*CMP WORD\_[DI], 'h'*

*JNE WORDS*

*CMP WORD\_[0], 'h'*

*JE WORDS*

*JMP CHECK\_NUMBER*

*WORDS:*

*;-----------------*

*;Текущая проверка слова*

*;-----------------*

*NEXT\_SYMB:*

*CMP SI, DI*

*JGE PRINT\_POL*

*MOV DL, WORD\_[SI]*

*CMP DL, WORD\_[DI]*

*JNE NEXT\_WORD*

*INC SI*

*DEC DI*

*JMP NEXT\_SYMB*

*;-----------------*

*;Текущая проверка числа*

*;-----------------*

*CHECK\_NUMBER:*

*MOV DX, DI ;длинна числа*

*MOV LEN, DI*

*NEXT\_NUMBER:*

*CLD*

*XOR AX, AX*

*MOV CX, ARR\_H\_LEN*

*MOV DI, 0*

*MOV AL, WORD\_[SI]*

*;Посимвольно проверим*

*;-----------------*

*@L:*

*CMP ARR\_H[DI], AL*

*JE NEXT*

*INC DI*

*LOOP @L*

*JMP WORDS*

*;-----------------*

*NEXT:*

*INC SI*

*DEC DX*

*JNZ NEXT\_NUMBER*

*MOV LEN, SI*

*MOV WORD\_[SI], 0*

*;-----------------*

*;Перевод в число*

*MOV SI, 0*

*MOV CL, 4H*

*XOR DX, DX*

*MOV AX, WORD PTR WORD\_[SI]*

*MOV DI, LEN*

*TO\_NUMBER:*

*;развернуть слово*

*SUB AL, 30H*

*CMP AL, 9H*

*JLE PRINT\_*

*SUB AL, 7H*

*CMP AL, 0FH*

*JLE PRINT\_*

*SUB AX, 20H*

*;-----------------*

*;Вывод*

*;-----------------*

*PRINT\_:*

*SHL DX, CL*

*ADD DL, AL*

*SHR AX, CL*

*SHR AX, CL*

*DEC DI*

*JNZ TO\_NUMBER*

*MOV DI, I*

*MOV ARR[DI], DL*

*ADD I, 1*

*JMP NEXT\_WORD*

*;-----------------*

*;Вывод всех палиндромов*

*;-----------------*

*PRINT\_POL:*

*ADD COUNT\_POL, 1*

*PUTLS WORD\_*

*NEXT\_WORD:*

*MOV LEN, 0*

*MOV DI, 0*

*CALL CLR\_WORD\_*

*CMP STR\_[BX], 0*

*JE @@E*

*JMP CHECK\_STR*

*@@E:*

*CMP COUNT\_POL, 0*

*JNE @N*

*PUTL NO\_POL*

*@N:*

*XOR CX, CX*

*XOR SI, SI*

*XOR AX, AX*

*MOV CX, I*

*CMP CX, 0*

*JE NO\_PRINT*

*@C:*

*MOV BL, ARR[SI]*

*ADD AX, BX*

*INC SI*

*LOOP @C*

*MOV SUM, AL*

*;-------------------------*

*MOV BL, AL*

*CMP AL, 0FH*

*JLE @JR*

*SHR AL, 4H*

*CMP AL, 9H*

*JLE PR1*

*ADD AL, 7H*

*PR1:*

*ADD AL, 30H*

*CALL PUTC*

*MOV AL, BL*

*AND AL, 0FH*

*@JR:*

*CMP AL, 9H*

*JLE PR2*

*ADD AL, 7H*

*PR2:*

*ADD AL, 30H*

*CALL PUTC*

*MOV AL, HEX*

*CALL PUTC*

*NO\_PRINT:*

*PUTL EMPTYS*

*EXIT*

*EXTRN PUTSS: NEAR*

*EXTRN PUTC: NEAR*

*EXTRN GETCH: NEAR*

*EXTRN GETS: NEAR*

*EXTRN SLEN: NEAR*

*EXTRN UTOA10: NEAR*

*CLR\_WORD\_ PROC NEAR*

*MOV CX, 80*

*PUSH SI*

*MOV SI, 0*

*CLR:*

*MOV WORD\_[SI], 0*

*INC SI*

*LOOP CLR*

*POP SI*

*RET*

*CLR\_WORD\_ ENDP*

*END BEGIN*

mopl2l.asm

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*; Процедуры для учебного примера лабораторной работы N 1 \**

*; по МОП. \**

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*.MODEL SMALL*

*.CODE*

*.386*

*INCLUDE MOPL2.INC*

*LOCALS*

*;=====================================================*

*; Подпрограмма вывода на экран строки, адресуемой SI,*

*; с задержкой времени между символами в <CX,DX> mcs.*

*; Завершителями строки являеются байты 0 или 0FFh.*

*; ЕСЛИ строка заканчивается байтом 0,*

*; ТО добавляется переход в начало новой строки*

*;*

*;=====================================================*

*PUTSS PROC NEAR*

*@@L: MOV AL, [SI]*

*CMP AL, 0FFH*

*JE @@R*

*CMP AL, 0*

*JZ @@E*

*CALL PUTC*

*INC SI*

*; CALL DILAY*

*JMP SHORT @@L*

*; Переход на следующую строку*

*@@E: MOV AL, CHCR*

*CALL PUTC*

*MOV AL, CHLF*

*CALL PUTC*

*@@R: RET*

*PUTSS ENDP*

*;==============================================*

*; Подпрограмма вывода AL на терминал*

*;==============================================*

*PUTC PROC NEAR*

*PUSH DX*

*MOV DL, AL*

*MOV AH, FUPUTC*

*INT DOSFU*

*POP DX*

*RET*

*PUTC ENDP*

*;==============================================*

*; Подпрограмма ввода символа в AL с терминала*

*;==============================================*

*GETCH PROC NEAR*

*MOV AH, FUGETCH*

*INT DOSFU*

*RET*

*GETCH ENDP*

*;=================================================*

*; Подпрограмма ввода строки в буфер, адресуемый DX*

*; и имеющий структуру:*

*; { char size; // размер буфера*

*; char len; // реально введено*

*; char str[size]; // символы строки }*

*;=================================================*

*GETS PROC NEAR*

*PUSH SI*

*MOV SI, DX*

*MOV AH, FUGETS*

*INT DOSFU*

*; прописать байт 0 в конец строки*

*XOR AH, AH*

*MOV AL, [SI+1]*

*ADD SI, AX*

*MOV BYTE PTR [SI+2], 0*

*POP SI*

*RET*

*GETS ENDP*

*;==============================================*

*; Подпрограмма подсчета числа символов в строке,*

*; адресуемой SI. Завершители строки: 0 и 0FFh*

*; Результат возвращается в AX*

*;==============================================*

*SLEN PROC NEAR*

*XOR AX, AX*

*LSLEN: CMP BYTE PTR [SI], 0*

*JE RSLEN*

*CMP BYTE PTR [SI], 0FFh*

*JE RSLEN*

*INC AX*

*INC SI*

*JMP SHORT LSLEN*

*RSLEN: RET*

*SLEN ENDP*

*;====================================================*

*; Подпрограмма преобразования <EDX,EAX> в беззнаковое*

*; десятичное, размещаемое по адресу DI*

*;==============================================*

*.DATA*

*UBINARY DQ 0 ; Исходное двоичное 64-разрядное*

*UPACK DT 0 ; Упакованные 18 десятичных цифр*

*.CODE*

*UTOA10 PROC NEAR*

*PUSH CX*

*PUSH DI*

*MOV DWORD PTR [UBINARY], EAX*

*MOV DWORD PTR [UBINARY+4], EDX*

*FINIT ; инициализация сопроцессора*

*FILD UBINARY ; забрасывание в него бинарного*

*FBSTP UPACK ; извлечение упаковонного десятичного*

*MOV CX, LENPACK ; получено 9 пар цифр*

*PUSH DS ; писать*

*POP ES ; будем*

*CLD ; через stosw*

*LEA SI, UPACK ; с конца*

*ADD SI, LENPACK ; буфера upack*

*; Цикл преобразования пар полубайтов в ASCII-коды цифр*

*@@L: XOR AX, AX*

*DEC SI*

*MOV AL, [SI]*

*SHL AX, 4*

*SHR AL, 4*

*ADD AX, 3030h*

*XCHG AL, AH*

*STOSW*

*LOOP @@L*

*; Фиксация конца строки*

*XOR AL, AL*

*STOSB*

*; Улучшим читабельность слишком длинного числа*

*CLD*

*MOV AX, LENNUM-4*

*@@L1: MOV CX, AX*

*POP DI ; встаем на начало строки*

*PUSH DI*

*MOV SI, DI*

*INC SI*

*REP MOVSB*

*MOV BYTE PTR [DI], CHCOMMA ; вставить разделитель троек цифр*

*SUB AX, 4 ; 3 цифры + разделитель обработаны*

*JS @@E ; прекратить, если осталось не больше 3-х цифр*

*JMP SHORT @@L1*

*@@E: POP SI*

*PUSH SI*

*XOR CX, CX*

*; Съедаем первые нули*

*; сначала подсчитываем*

*@@L2: CMP BYTE PTR [SI], '0'*

*JE @@N*

*CMP BYTE PTR [SI], CHCOMMA*

*JNE @@N1*

*@@N: INC CX*

*INC SI*

*JMP SHORT @@L2*

*@@N1: ; а теперь съедаем*

*POP DI*

*SUB CX, LENNUM+1*

*NEG CX*

*REP MOVSB*

*POP CX*

*RET*

*UTOA10 ENDP*

*;==============================================*

*; Подпрограмма задержки выполнения программы*

*; на <CX,DX> микросекунд*

*;==============================================*

*DILAY PROC NEAR*

*MOV AH, 86h*

*INT 15h*

*RET*

*DILAY ENDP*

*HEX2DEC PROC NEAR*

*MOV CX,0*

*MOV BX,10*

*LOOP1: MOV DX,0*

*DIV BX*

*ADD DL,30H*

*PUSH DX*

*INC CX*

*CMP AX,9*

*JG LOOP1*

*ADD AL,30H*

*MOV [SI],AL*

*LOOP2: POP AX*

*INC SI*

*MOV [SI],AL*

*LOOP LOOP2*

*RET*

*HEX2DEC ENDP*

*PUBLIC PUTSS, PUTC, GETCH, GETS, DILAY, SLEN, UTOA10, HEX2DEC*

*END*

mopl2.inc

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*; Константы для учебного примера лабораторной работы N 1 \**

*; по МОП. \**

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*CHESC = 1Bh*

*CHCR = 13*

*CHSEP = '='*

*CHCOMMA = '`'*

*CHLF = 10*

*DOSFU = 21h*

*FUPUTC = 2*

*FUGETCH = 8*

*FUGETS = 0Ah*

*FUEXIT = 4Ch*

*PAUSE\_L = 0000 ; Младшее слово стартового значения паузы*

*PAUSE\_H = 0 ; Старшее слово*

*LENNUM = 18 ; Число символов в строке десятичного числа*

*LENPACK = 9 ; Упаковонное в два раза короче*

mopl2.mac

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*; Макросы для учебного примера лабораторной работы N 1 \**

*; по МОП. \**

*;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*; Макрос вывода строки LINE*

*PUTL MACRO LINE*

*LEA SI, LINE*

*XOR CX, CX*

*XOR DX, DX*

*XOR AX, AX*

*CALL PUTSS*

*ENDM*

*; Макрос вывода строки LINE с задержкой PAUSE*

*PUTLS MACRO LINE*

*XOR AX, AX*

*LEA SI, LINE*

*; MOV DX, PAUSE*

*; MOV CX, PAUSE+2*

*CALL PUTSS*

*ENDM*

*; Как и PUTLS, но с центрированием посередине*

*PUTLSC MACRO LINE*

*LOCAL LO, NOMARG*

*LEA SI, LINE*

*CALL SLEN*

*MOV CX, 80*

*SUB CX, AX*

*SAR CX, 1*

*JZ NOMARG*

*LO: MOV AL, ' '*

*CALL PUTC*

*LOOP LO*

*NOMARG: PUTLS LINE*

*ENDM*

*; Макрос фиксации начала измерения времени*

*; Заносит в стек младшее слово числа тактов системных часов*

*FIXTIME MACRO*

*DB 0Fh, 31h; Команда RDTSC чтения счетчика тактов*

*PUSH EDX*

*PUSH EAX*

*ENDM*

*; Макрос получения в <EDX,EAX> числа тактов, прошедших с последнего*

*; момента фиксации времени. Извлекает продукт FIXTIME из стека*

*DURAT MACRO*

*DB 0Fh, 31h; Команда RDTSC чтения счетчика тактов*

*POP EBX*

*SUB EAX, EBX*

*POP EBX*

*SBB EDX, EBX*

*ENDM*

*; Макрос завершения программы*

*EXIT MACRO*

*MOV AH, FUEXIT*

*MOV AL, 0*

*INT DOSFU*

*ENDM*