.MODEL small

.CODE

ORG 100H

Begin: ;метка начала программы

jmp Start

filenameMsg db 'Имя файла: '

FileName DB 'string.txt',0,10,13,'$'

charMsg db 'Строка букв',10,13,'$'

binMsg db 'Двоичное число',10,13,'$'

octMsg db 'Восьмеричное число',10,13,'$'

decMsg db 'Десятичное число',10,13,'$'

hexMsg db 'Шестнадцатеричное число',10,13,'$'

mixedMsg db 'Строка состоит из смешанных символов',10,13,'$'

typesHeader db 10,13,'Возможные типы строки:',10,13,'$'

ErrStr DB 'Ошибка работы с файлом. Возможно, файл отсутствует.',10,13,'$'

Header DB 'Расчетно-графическая работа',10,13,'<<Системное программное обеспечение>>',10,13,'Студента группы ИВТ-314',10,13,'Лобова В.А.',10,13,'Вариант 13',10,10,13,'$'

PressContinue DB 'Press any key to continue...',10,10,13,'$'

progSizeMsg db 'Размер программы: '

progSize db ' ',' байт',10,13,'$'

isChar db 1

isBin db 1

isOct db 1

isDec db 1

isHex db 1

isMixed db 1

fileMsg db 'Строка в файле:'

Buffer DB 80 DUP(0)

Start:

;процедура вычисление размера файла

lea dx, EndFile ;загружаем в dx адрес метки на конце файла

mov ax, dx ;копируем в ax

lea dx, Begin ;загружаем в dx адрес метки на начале файла

sub ax,dx ;вычитаем из конца начало, получая размер в байтах в ax

;процедура перевода числа из регистра ax(размера программы) в символьную строку

std ; Устанавливаем ОБРАТНЫЙ порядок записи

lea di,progSize+3 ; ES:DI = последний символ строки progSize

;Начинаем перевод числа AX в строку

mov cx,10 ; Задаемся делителем CX = 10

Repeat:

mov dx,0 ; Обнуляем DX (для деления)

div cx ; Делим DX:AX на CX (10),

;Получаем в AX частное, в DX остаток

xchg ax,dx ; Меняем их местами (нас интересует остаток)

add al,'0' ; Получаем в AL символ десятичной цифры

stosb ; И записываем ее в строку progSize

xchg ax,dx ; Восстанавливаем AX (частное)

cmp ax,0 ; Сравниваем AX с 0

jne Repeat ; Если не ноль, то повторяем

;на этом вывод программы закончен. переходим к самой проге

lea dx,progSizeMsg ;загружаем адрес строки с размером программы

call Show ;функция вывода

lea dx,Header ;загружаем адрес строки заголовка

call Show ;функция вывода

lea dx, filenameMsg ;загружаем адрес строки с именем файла

call Show

lea dx,PressContinue ;адрес строки с сообщением о продолжении

call Show

MOV AH,10H ;ожидание нажатия кнопки

INT 16H

lea dx,FileName ;загружаем в dx адрес с именем файла

call OpenFile ;получаем дескриптор файла через функцию

jnc l ;проверить флаг на ошибку открытия файла(файл отсутствует) ;эта команда и ниже эквивалентны jc Error. Дальности условного перехода jc не хватает, поэтому используем jmp, так как у него дальность больше

jmp Error

l: mov bx,ax ;сохраняем дескриптор файла в bx

lea dx,Buffer ;записываем адрес буфера в dx

mov cx,1 ;сколько байт(символов) читать. Будем читать по одному символу

Read:

mov ah,3fh ;функция считывания из файла

int 21h ;сервис dos

inc dx ;сдвинем адрес записи на ячейку

cmp ax, 0 ;в ax будет 0, если достигнут конец файла

jnc l1 ;Аналогично эта команда и ниже эквивалентны одной команде jc Error. Дальности условного перехода jc не хватает, поэтому используем jmp, так как у него дальность больше

jmp Error ;проверить флаг на ошибку

l1: jne Read ;если не конец файла, читаем дальше

mov si,dx

mov [si],'$'

dec dx

;dec dx

Call CloseFile ;функция закрытие файла

;начинаем обработку

mov ax,dx ;сохраняем конечный адрес полученной строки в ax

lea dx,Buffer ;берем начальный адрес строки

;call Show

sub ax,dx ;Вычитаем из адреса конца строки адрес начала. Тем самым находим размер строки

mov cx,ax ;сохраним размер строки в cx. Будем использовать его как счетчик.

lea dx, fileMsg ;выведем сообщение с содержимымм файла

call Show

mov bx, 0 ;счетчик сдвига по буферу

Process:

mov al, [Buffer + bx] ;копируем из буфера очередной считанный символ

cmp al,'0' ;сравниваем с ascii-кодом 0

jnb l2 ;Аналогично эта команда и ниже эквивалентны одной команде jb Error. Дальности условного перехода jb не хватает, поэтому используем jmp, так как у него дальность больше

jmp Other ;в случае если значение лежит в интервале от начала до символа 0.

l2: cmp al,'1'

jbe Bin ;в случае если значение лежит в интервале от 0 до 1 включительно

cmp al,'7'

jbe Oct ;в случае если значение лежит в интервале от 1 до 7 включительно

cmp al,'9'

jbe Decim ;в случае если значение лежит в интервале от 8 до 9 включительно

cmp al,'A'

jnb l3 ;Аналогично эта команда и ниже эквивалентны одной команде jb Error. Дальности условного перехода jb не хватает, поэтому используем jmp, так как у него дальность больше

jmp Other ;в случае если значение лежит в интервале от до 7 включительно

l3: cmp al,'F'

jbe Hex

cmp al,'Z'

jbe Char

cmp al,'a'

jnb l4 ;Аналогично эта команда и ниже эквивалентны одной команде jb Error. Дальности условного перехода jb не хватает, поэтому используем jmp, так как у него дальность больше

jmp Other

l4: cmp al,'f'

jbe Hex

cmp al,'z'

jbe Char

jmp Other

Char: ;если в строке есть буквы(начиная с g), значит это уже не число

mov [isBin], 0

mov [isOct], 0

mov [isDec], 0

mov [isHex], 0

jmp Lop

Bin: ;двоичное число. если в строке есть цифра, то это уже не строка букв

mov [isChar], 0

jmp Lop

Oct: ;восьмеричное число. если в строке есть цифра больше 1, то это уже не строка букв и не двоичное число.

mov [isBin], 0

mov [isChar], 0

jmp Lop

Decim: ;десятичное число. если в строке есть цифра больше 7, то это уже не строка букв, не двоичное и не восьмеричное число.

mov [isBin], 0

mov [isOct], 0

mov [isChar], 0

jmp Lop

Hex: ;шестнадцатеричное число. если в строке есть цифра больше 9(a-f), то это уже не строка букв, не двоичное, не восьмеричное и не десятичное число.

mov [isBin], 0

mov [isOct], 0

mov [isDec], 0

jmp Lop

Other: ;в случае, если в строке встретилась не буква и не цифра.

mov [isBin], 0

mov [isOct], 0

mov [isDec], 0

mov [isHex], 0

mov [isChar], 0

jmp l5

Lop:

inc bx ;увеличиваем счетчик сдвига по буферу

dec cx ;уменьшаем общий счетчик

cmp cx, 0

je l5

jmp Process

l5: lea dx, typesHeader ;выводим сообщение

call Show

cmp [isChar], 1 ;проверяем, это строка букв?

jne a1 ;если нет, то проверяем сразу следующий флаг

lea dx, charMsg ;если да, выводим собщение об этом

call Show

mov [isMixed], 0 ;значит, это уже строка не смешанных символов

a1: cmp [isBin], 1 ;проверяем, это строка с двичным числом?

jne a2 ;если нет, то проверяем сразу следующий флаг

lea dx, binMsg ;если да, выводим собщение об этом

call Show

mov [isMixed], 0 ;значит, это уже строка не смешанных символов

a2: cmp [isOct], 1 ;проверяем, это строка с восьмеричным числом?

jne a3 ;если нет, то проверяем сразу следующий флаг

lea dx, octMsg ;если да, выводим собщение об этом

call Show

mov [isMixed], 0 ;значит, это уже строка не смешанных символов

a3: cmp [isDec], 1 ;проверяем, это строка с десятичным числом?

jne a4 ;если нет, то проверяем сразу следующий флаг

lea dx, decMsg ;если да, выводим собщение об этом

call Show

mov [isMixed], 0 ;значит, это уже строка не смешанных символов

a4: cmp [isHex], 1 ;проверяем, это строка с шестнадцатеричным числом?

jne a5 ;если нет, то проверяем сразу следующий флаг

lea dx, hexMsg ;если да, выводим собщение об этом

call Show

mov [isMixed], 0

a5: cmp [isMixed],1 ;проверяем, это строка букв?

jne Exit ;если нет, заканчиваем

lea dx, mixedMsg ;значит, это уже строка не смешанных символов

call Show

jmp Exit

a6:

Error:

lea dx, ErrStr ;загружаем в dx адрес строки с сообщением об ошибке

call Show

Exit:

MOV AH,10H ;ожидание нажатия кнопки

INT 16H

mov ah,04Ch ;функция DOS выхода из пpогpаммы

mov al,0h ;код возвpата

int 21h

OpenFile PROC ;Функция открытия файла

mov ah,3dh ;функция открытия файла

mov al,0 ;0 - открыть для чтения 1 - запись 2 - чтение,запись

int 21h ;сервис dos

ret ;возврат

OpenFile ENDP

CloseFile PROC ;функция закрытия файла

mov ah,3eh ;функция закрытия файла

int 21h ;сервис dos

ret

CloseFile ENDP

Show PROC ;функция вывода

mov ah,09 ;функция вывода строки на экран

int 21h ;dos

ret

Show ENDP

EndFile: ;метка на конце программы для вычисления ее размера

End Begin