Цель: закрепление навыков разработки и отладки многомодульной программ на языке ассемблера для IBM-совместимых компьютеров.

При решении сложных задач используется метод декомпозиции, предусматривающий разбиение решаемой задачи на совокупность подзадач, которые упорядочиваются иерархически. Процесс решение каждой подзадачи можно представить в виде совокупности действий, оформленных как программный модуль для преобразования указанных исходных данных в результирующие данные. Для управления последовательностью своевременных вызовов программных модулей, предназначенных для решения соответствующих подзадач, выделяется программный модуль MAIN верхнего уровня в иерархической структуре модулей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 16 | 1 | 1 | Больше 5 | Цифры на пробелы |

|  |  |
| --- | --- |
| TITLE Main  EXTRN INPUT:FAR,MASKS:FAR  EXTRN OUTNUMBER:FAR,OUTSTAR:FAR  EXTRN FILECHANGE:FAR  PUBLIC resmask,msk,object,result  .model small  .stack 200h  .data  msk dw ?  object dw ?  resmask dw ? ;хранение результата побитового преобразования  result db ? ;хранение количества единиц  message2 db 10,13,'Vvod object $'  message3 db 10,13,'Vvod IIepeMeHHou msk $'  mess2 db 10,13,'number of 1$'  mess3 db 10,13,'more than 5? $'  .code  begin:  mov ax,@data  mov ds,ax  mov ah,9  lea dx,message3  int 21h  lea di,msk  call INPUT  mov ah,9  lea dx,message3  int 21h  lea di,object  call INPUT  call MASKS  mov resmask,0  mov resmask,ax ;подсчет единиц  mov ah,09h  lea dx,mess2  int 21h  mov ah,1  int 21h  lea bx,resmask  mov cx,16  mov al,0  Metka:  rcl word ptr[bx],1  JC pe1  JMP Loo2 | pe1:  inc al    Loo2:  loop Metka  mov result,al ;отображение результата (больше пяти - звездочки, меньше - цифры)  mov ah,09h  lea dx,mess3  int 21h  cmp result,6  jb outnumb  call OUTSTAR  JMP Tree  outnumb:call OUTNUMBER  Tree:  call FILECHANGE  mov ah,1  int 21h  mov ax, 4C00h ;выход из программы  int 21h  End begin |

|  |  |
| --- | --- |
| TITLE INPUT  PUBLIC INPUT  EXTRN msk:dword  EXTRN object:dword  .Model Small  .STACK 100h  .data  message db 10,13,'Enter 1 or 0$'  .code  INPUT proc far  mov ax,@data  mov ds,ax  mov cx,16  vv\_0\_1:  mov ah,9  lea dx,message  int 21h  mov ah,1  int 21h  cmp al,31h  jz vv0  cmp al,30h  jz vv1  jmp vv\_0\_1  vv0:  stc  rcl word ptr[di],1  jmp loo1  vv1:  clc  rcl word ptr[di],1  loo1:  loop vv\_0\_1  ret  INPUT endp  End | TITLE MASKS  Public MASKS  EXTRN msk:word  EXTRN object:word  .Model Small  .STACK 100h  .data  .code  MASKS proc far  mov ax,@data  mov ds,ax  mov ax,object  mov bx,msk  or ax,bx  ret  MASKS endp  End  TITLE OUTNUMBER  Public OUTNUMBER  EXTRN result:byte  .Model Small  .STACK 100h  .data  mess db 10,13,'less than 5, 4ucJloBou $'  .code  OUTNUMBER proc far  mov ax,@data  mov ds,ax ;вывод на экран количества единиц    mov ah,09h  lea dx,mess  int 21h  cmp result,10  jb m1  sub result,10  mov dl,’1’  mov ah,2  int 21h  m1:  mov dl,result  or dl,30h  mov ah,2  int 21h  ret  OUTNUMBER endp  End |

|  |  |
| --- | --- |
| TITLE OUTSTAR  Public OUTSTAR  extrn result:byte  EXTRN resmask:dword  .Model Small  .STACK 100h  .data  messa db 10,13,'$'  Prob db 1 dup(' '), '$'  Star db '\*', 10d, 13d, '$'  mess db 10,13,'More than 5, 3Be#Do4ku $'  .code  OUTSTAR proc  mov ax,@data  mov ds,ax  mov ah,09h  lea dx,mess  int 21h  mov ah,09h  lea dx,messa  int 21h  lea bx,resmask  mov cx,16  mov al,0  std  M:    rcl word ptr[bx],1  JC p1  JMP L  p1:  inc al    L:  loop M  mov result,al  xor cx,cx  mov cl,result  mov bx,0  diag:  push cx  mov cx,bx  jz no\_cond  stri:  mov ah,2  mov dl,0  int 21h  loop stri  no\_cond:  mov ah,9  lea dx,star  int 21h  pop cx  inc bx  loop diag  ret  outstar endp  end | TITLE FILECHANGE  Public FILECHANGE  .model small  .stack 100h  .data  mess123 db 10,13, '$'  mess1 db 10,13, 'Open first file $'  mess11 db 10,13, 'Reading first file $'  mess2 db 10,13, 'Successful $'  mess3 db 10,13, 'Error $'  mess4 db 10,13, 'Error while reading $'  mess5 db 10,13, 'Now you can view a text $'  mess6 db 10,13, 'Edited text $'  mess7 db 10,13, 'Error in creating $'  mess8 db 10,13, 'Creating a second file $'  mess9 db 10,13, 'Error in writing $'  mess10 db 10,13, 'Saving a second file $'  file1 db 'Lab12.txt', 0 ; местоположение файла  file2 db 'Lab.txt', 0 ;создаваемый файл  buffer db 75 dup(?)  buffer2 db 75 dup(?)  FM dw ?  FM2 dw ?  .code  FILECHANGE proc far    mov ax,@data  mov ds,ax  mov ah,09h  lea dx,mess123  int 21h  mov ah, 09h  lea dx,mess1  int 21h  ; открываем первый файл  mov ah,3dh ; в ah задан номер функции открытия файла  mov al,0 ; в al задан режим открытия  lea dx,file1 ; в dx адрес строки содержащей идентификацию файла  int 21h  jc err1  jmp p1 ;если ошибка, то переход к выходу из программы  err1:  mov ah,09h  lea dx,mess3  int 21h  JMP Finish  p1: mov [FM],ax ;в FM сохранятеся логический номер файла получаемого от DOS  mov ah,09h  lea dx,mess2  int 21h ;читаем первый файл  mov ah, 09h  lea dx,mess11  int 21h |

|  |  |
| --- | --- |
| mov ah, 3Fh ; в ah задан номер функции чтения из файла.  mov bx, FM ; регистр bx должен содержать логический номер файла.  mov cx, 75 ; в cx количество читаемых байтов.  lea dx, buffer ; в ds:dx указывается адрес буфера для ввода данных из файла.  int 21h ; вызов функции чтения.  jc err2  JMP p2  err2:  mov ah,09h  lea dx,mess4  int 21h  JMP Finish  p2: mov ah,09h  lea dx,mess2  int 21h  ;закрытие первого файла  mov ah, 3Eh  mov bx, [FM]  int 21h  ;вывод прочитанного текста на экран  mov ah,09h  lea dx,mess5  int 21h  mov ah,09h  lea dx,mess123  int 21h  lea di,buffer ;Адрес буфера с прочитанными данными  mov bx, 75 ;количество символов в тексте  mov byte ptr [di+bx], '$' ;для выводs текста б  mov ah, 09h  lea dx,buffer  int 21h  mov cx,75  cld ;сброса флага df в ноль, т.е. проверка будет идти слева направо  lea si,buffer  lea di,buffer2  Cycle: lodsb ;загружаем в al первый байт из Buf  cmp al,'0' ;проверяем, входит ли он в диапазон цифр  jb jump\_01 ;если нет, то оставляем и пишем во второй буфер  cmp al,'9' ;второй буфер служит для хранения редактированного текста  ja jump\_01  mov bx,[si-1] ;если входит, меняем на пробел и пишем во второй буфер  mov [di],bx  mov [di],00h  inc di  jmp jump\_02  jump\_01:  mov bx,[si-1]  mov [di],bx  inc di  jump\_02:  loop Cycle  ;вывод редактированного текста | mov ah,09h  lea dx,mess6  int 21h  mov ah,1  int 21h  mov ah,09h  lea dx,mess123  int 21h  int 21h  lea di,buffer2 ;Адрес буфера с прочитанными данными  mov bx, 75 ;количеству символов в строке  mov byte ptr [di+bx], '$' ;для вывода текста без 'мусора'    mov ah, 09h  lea dx,buffer2  int 21h  ;создание 2 файла для сохранения  mov ah,09h  lea dx,mess8  int 21h  mov ah,3ch  lea dx,file2  mov cx,0  int 21h  jc err3  mov [FM2],ax  mov ah,09h  lea dx,mess2  int 21h  ;запись в файл  mov ah,09h  lea dx,mess10  int 21h  mov ah, 40h  mov bx,[FM2]  lea dx, buffer2  mov cx, 75  int 21h  jc err4  mov ah,09h  lea dx,mess2  int 21h  ;закрытие второго файла  mov ah, 3Eh  mov bx, [FM2]  int 21h  JMP Finish  err3:  mov ah,09h  lea dx,mess7  int 21h  JMP Finish  err4:  mov ah,09h  lea dx,mess9  int 21h  JMP Finish  Finish:  ret FILECHANGE endp  End |

Пример работы программы:



