Ассемблер Intel 8086

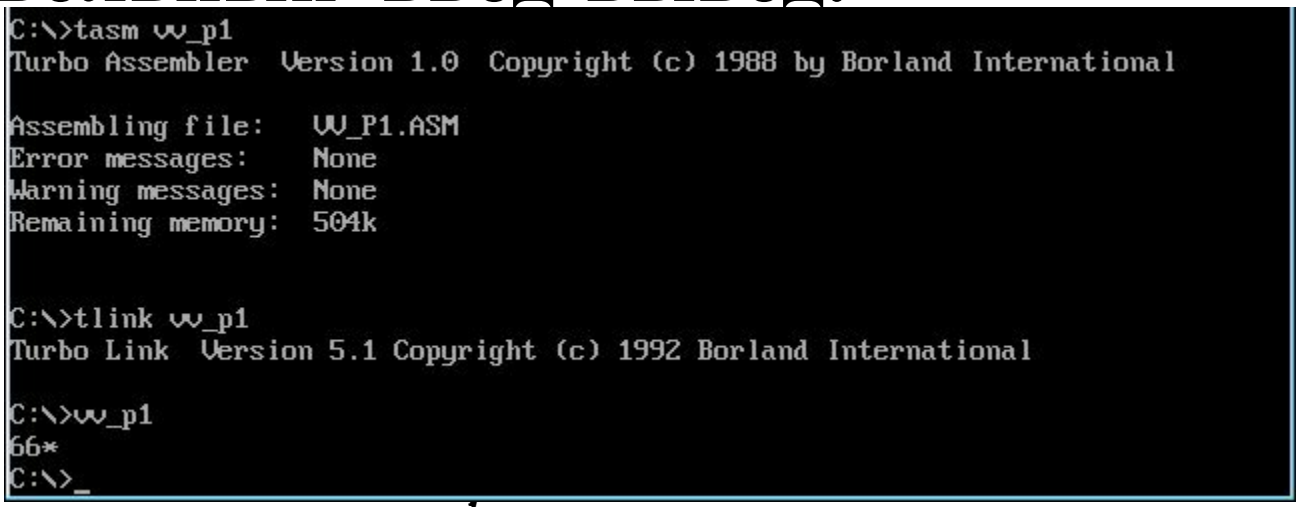
Символьный ввод-вывод.

```
dosseg
.model small
.stack 200h
.data
.code
Begin:
   mov ax,@Data
   mov ds, ax

   ;ввод символа
   Mov  ah,01 ; номер функции помещаем в ah
   Int  21h  ; передаем управление DOS

   ;вывод символа
   Mov  ah,02 ; номер функции помещаем в ah
   Mov  dl,al ; код выводимого символа помещаем в dl
   Int  21h  ; передаем управление DOS
   Mov  ah,02 ; номер функции помещаем в ah
   Mov  dl, "*" ; код выводимого символа помещаем в dl
   Int  21h  ; передаем управление DOS

   mov ax, 4C00h
   int 21h
   end Begin
```



Ассемблер Intel 8086

Вывод строки.

```
dosseg
.model small
.stack 200h
.data
   Msg  db  'This text should appear on the screen' ; размещаем (в
   ; сегменте данных) выводимую строку
   Len = $-Msg  ; Len присваиваем
   ;значение длины строки (разница
   ; между текущим смещением в
   ;сегменте и смещением до Msg)
.code
Begin:
   mov ax,@Data
   mov ds, ax

   Mov  cx,Len  ; инициализируем счетчик цикла
   Lea  si,Msg  ; устанавливаем si на начало области Msg (на
   ; первый байт выводимой строки)
M: Mov  ah,02   ; номер функции помещаем в ah
   Mov  dl,[si]   ; код выводимого символа помещаем в dl
   Int  21h    ; передаем управление DOS
   Inc  si    ; продвигаем si к следующему символу (байту)
   Loop M    ; организуем цикл с меткой M

   mov ax, 4C00h
   int 21h
   end Begin
```

```
C:\>tasm vv_p2
Turbo Assembler  Version 1.0  Copyright (c) 1988 by Borl

Assembling file:   VV_P2.ASM
Error messages:    None
Warning messages:  None
Remaining memory:  504k


C:\>tlink vv_p2
Turbo Link  Version 5.1 Copyright (c) 1992 Borland Inter

C:\>vv_p2
This text should appear on the screen
C:\>
```

Ассемблер Intel 8086

Вывод строки.

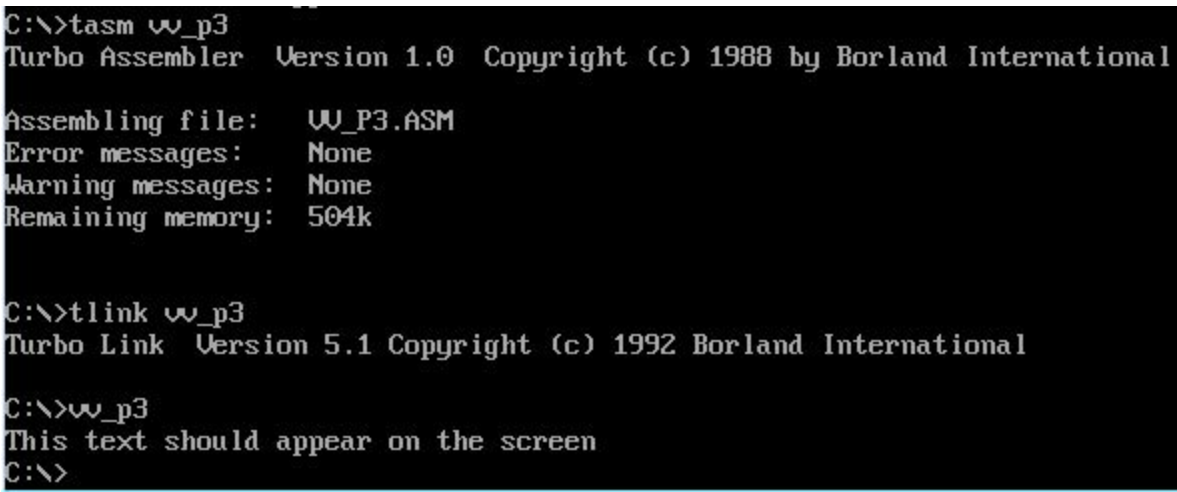
```
dosseg
.model small
.stack 200h
.data
   Msg  db  'This text should appear on the screen$'  ;  размещаем
   ;(в сегменте данных) выводимую строку, последний знак $ - ограничитель
   ;строки для DOS

.code
Begin:
   mov ax,@Data
   mov ds, ax

   Mov  ah,09   ; номер функции помещаем в ah
   Lea dx,Msg  ;смещение в сегменте до Msg помещаем в dx
   Int 21h    ; передаем управление DOS

   mov ax, 4C00h
   int 21h

   end Begin
```



Ассемблер Intel 8086

Ввод строки.

```
dosseg
.model small
.stack 200h
.data
    Len = 256     ; пусть для определенности строка не может превышать 256 байт
    Str    db  Len dup (?) ; резервируем область памяти под вводимую
    ; строку (в сегменте данных)
.code
Begin:
    mov ax,@Data
    mov ds, ax

    cr = 0dh     ; присваиваем cr значение кода клавиши «Enter»
    Mov  cx,Len   ; инициализируем счетчик цикла
    Lea  di,str    ; устанавливаем di на начало области Str
M:  Mov  ah,01    ; номер функции помещаем в ah
    Int  21h     ; передаем управление DOS
    Mov  [di],al    ; код  введенного символа помещаем в очередной байт области Str
    Cmp  al,cr     ; проверяем, не является ли полученный код кодом клавиши «Enter»
    Je  Ex     ; если да – ввод завершен, выходим из цикла
    Inc  di     ; продвигаем di к следующему байту Str
    Loop M     ; организуем цикл
Ex: ; при выходе из цикла cx содержит количество незаполненных байтов в Str
    Mov  [di],'$'
    Mov  ah,09    ; номер функции помещаем в ah
    Lea dx,Str  ;смещение в сегменте до Msg помещаем в dx
    Int 21h     ; передаем управление DOS

    mov ax, 4C00h
    int 21h
    end Begin
```

```
C:\>tlink vv_p4
Turbo Link  Version 5.1 Copyright (c) 1992 Borland International

C:\>vv_p4
1234asd
1234asd
```

Ассемблер Intel 8086

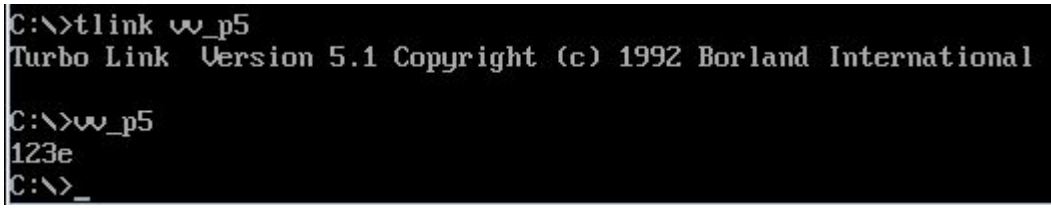
Ввод строки.

```
dosseg
.model small
.stack 200h
.data
   Str    db  255, ?, 255 dup (?),0dh,'$' ; резервируем область памяти под
   ; вводимую строку (в сегменте данных). Первый байт – максимальная длина
   ; строки, второй – фактическая длина (заполняется операционной системой
   ; в процессе ввода), далее – собственно строка символов
.code
Begin:
   mov ax,@Data
   mov ds, ax

   Mov  ah,0ah   ; номер функции помещаем в ah
   Lea  dx,Str   ;смещение в сегменте до Str помещаем в dx
   Int  21h    ; передаем управление DOS

   Lea  di,Str
   Mov  ch,0  ; инициализируем счетчик цикла
   Mov  cl,[di+1]  ; инициализируем счетчик цикла
   Lea  si,Str  ; устанавливаем si на начало области Str (на
   ; первый байт выводимой строки)
   add  si,2
M: Mov  ah,02   ; номер функции помещаем в ah
   Mov  dl,[si]   ; код выводимого символа помещаем в dl
   Int  21h    ; передаем управление DOS
   Inc  si    ; продвигаем si к следующему символу (байту)
   Loop M    ; организуем цикл с меткой M

   mov ax, 4C00h
   int 21h
   end Begin
```



Ассемблер Intel 8086

Ввод-вывод чисел.

Для ввода-вывода числовых данных операционная система не предоставляет никаких возможностей, поэтому преобразование вводимых символов в число и обратно полностью возлагается на программу.

Рассмотрим это преобразование на примере числа 6543 и десятичной системы счисления

Деление	Частное	Остаток	Код символа	Символ
6543 : 10	654	3	51	‘3’
654 : 10	65	4	52	‘4’
65 : 10	6	5	53	‘5’
6 : 10	0	6	54	‘6’

Ассемблер Intel 8086

Вывод беззнаковых чисел.

```
dosseg
.model small
.stack 200h
.data
.code
UnsignedOut proc
    xor cx,cx  ;обнуляем счетчик цифр
    mov bx,10  ;в bx помещаем делитель(основание системы счисления)
m:      inc cx     ;считаем количество получающихся цифр
    xor dx,dx  ;преобразуем делимое к 32 разрядам
    div bx     ;получаем очередную цифру
    push dx    ;сохраняем ее в стеке
    or  ax,ax  ;проверяем, есть ли еще цифры
    jnz m    ;если да – на метку m при выходе из цикла в стеке лежат цифры, в cx –
    их
    ;количество
m1:     pop dx     ;извлекаем цифру из стека
    add dx,'0'  ;преобразуем в код символа
    mov ah,2   ;функцией 02 выводим на экран
    int 21h
    loop m1    ;повторяем cx раз
    ret     ;возвращаемся из процедуры
UnsignedOut endp
Begin:
    mov ax,@Data
    mov ds, ax

    mov ax,7459
    call UnsignedOut

    mov ax, 4C00h
    int 21h
```

```
Assembling file:   VV_P6.ASM
Error messages:    None
Warning messages:  None
Remaining memory:  504k


C:\>tlink vv_p6
Turbo Link  Version 5.1 Copyright (c) 1992 Borland International

C:\>vv_p6
7459
C:\>_
```

Ассемблер Intel 8086

Вывод целых чисел.

```
dosseg
.model small
.stack 200h
.data
.code
IntegerOut  proc
    xor cx,cx  ;обнуляем счетчик цифр
    mov bx,10  ;в bx помещаем делитель
    cmp ax,0   ;проверяем знак числа
    jge m    ;если неотрицательное – на m
    neg ax     ;иначе – меняем знак числа
    push ax    ;сохраняем число перед вызовом функции,
    ;использующей ax
    mov ah,2   ;функцией 02 выводим знак '-'
    mov dl,'-'
    int 21h
    pop ax     ;восстанавливаем число в ax
m:      inc cx     ;считаем количество получающихся
    цифр
    xor dx,dx  ;преобразуем делимое к 32 разрядам
    div bx     ;получаем очередную цифру
    push dx    ;сохраняем ее в стеке
    or  ax,ax  ;проверяем есть ли еще цифры
    jnz m    ;если да – на метку m
    ;при выходе из цикла в стеке лежат цифры, в cx – их
    ;количество
    m1:     pop dx     ;извлекаем цифру из стека
    add dx,'0'  ;преобразуем в код символа
    mov ah,2   ;функцией 02 выводим на экран
    int 21h
    loop m1    ;повторяем cx раз
    ret     ;возвращаемся из процедуры
```

```
Begin:
    mov ax,@Data
    mov ds, ax

    mov ax,-10519
    call IntegerOut

    mov ax, 4C00h
    int 21h

    end Begin
```

```
C:\>tasm vv_p7
Turbo Assembler  Version 1.0  Copyri

Assembling file:   VV_P7.ASM
Error messages:    None
Warning messages:  None
Remaining memory:  504k


C:\>tlink vv_p7
Turbo Link  Version 5.1 Copyright (c

C:\>vv_p7
-10519
C:\>
```

Ассемблер Intel 8086

Ввод беззнаковых чисел.

Ввод чисел сопровождается преобразованием строки символов в число. Данное преобразование производится получением последовательности цифр из символов (вычитанием 48 – кода символа ’0’) и последующим сложением цифр с соответствующими весами. Данная операция описывается формулой (для числа из пяти цифр):

m=((((s1-48)*n+(s2-48))*n+(s3-48))*n+(s4-48))*n+(s5-48) , где
m – получаемое число,
n – основание системы счисления,
s1,s2,s3,s4,s5 – коды первого, второго, третьего, четвертого и пятого символов строки соответственно.

Например строка ‘31562’ для десятичной системы счисления:
s1=51, s2=49, s3=53, s454=, s5=50
s1-48=3
(s1-48)*10+(s2-48)=31
((s1-48)*10+(s2-48))*10+(s3-48)=315
(((s1-48)*10+(s2-48))*10+(s3-48))*10+(s4-48)=3156
((((s1-48)*10+(s2-48))*10+(s3-48))*10+(s4-48))*10+(s5-48)=31562

Ассемблер Intel 8086

Ввод беззнаковых чисел.

```
UnsignedIn  proc
start:    mov ah,0ah    ;функцией 0a вводим строку символов и размещаем ее в
   ;области string
   lea dx,string
   int 21h
   xor ax,ax    ;обнуляем ax, в котором будем формировать число
   lea si,string+2 ;устанавливаем si на первый символ введенной ;строки
   ;анализируем текущий символ
m2:      cmp byte ptr [si],cr ;если это cr – строка закончилась, выходим
   je  ex
   cmp byte ptr [si],'0';если код символа меньше кода '0' – ;это не цифра
   jb  err       ;прыгаем на метку err
   cmp byte ptr [si],'9';если код символа больше кода '9' – ;это не цифра
   ja  err       ;прыгаем на метку err
   mov bx,10      ;домножаем полученное число на основание ;системы счисления
   mul bx
   sub byte ptr [si],'0';вычитаем код символа '0' (получаем очередную цифру)
   add al,[si]     ;добавляем цифру к числу
   adc ah,0
   inc si        ;продвигаем si к следующему символу
   jmp m2        ;организуем цикл функцией 09 выводим сообщение об ошибке
err:     lea  dx,errmsg
   mov ah,9
   int 21h
   jmp start    ;повторяем ввод
ex:     ret
UnsignedIn  endp
```

Ассемблер Intel 8086

Ввод беззнаковых чисел.

```
dosseg
.model small
.stack 200h
.data
   cr = 0dh ;cr присваиваем значение кода символа возврата каретки (клавиши
   «Enter»)
   lf = 0ah ;lf присваиваем значение кода символа перевода строки
   string    db  255, 0, 255 dup (?)
   errmsg    db  'Недопустимый символ, можно'
   db  'использовать только цифры',cr,lf,'$'
.code
Begin:
   mov ax,@Data
   mov ds, ax

   call UnsignedIn
   call IntegerOut

   mov ax, 4C00h
   int 21h

   end Begin
```

```
C:\>tasm vv_p8
Turbo Assembler  Version 1.0  Copyright (c) 1988 by Borland International

Assembling file:   VV_P8.ASM
*Warning* VV_P8.ASM(76) Reserved word used as symbol: ERR
Error messages:    None
Warning messages:  1
Remaining memory:  503k


C:\>tlink vv_p8
Turbo Link  Version 5.1 Copyright (c) 1992 Borland International

C:\>vv_p8
456
C:\>
```

Ассемблер Intel 8086

Ввод целых чисел.

```
IntegerIn  proc
start: mov ah,0ah;функцией 0a вводим строку символов и размещаем ее в области string
   lea dx,string
   int 21h
   xor ax,ax    ;обнуляем ax, в котором будем формировать число
   lea si,string+2 ;устанавливаем si на первый символ введенной строки
   mov negflag,ax  ;обнуляем флаг отрицательности числа (предполагаем, что оно
   ;будет неотрицательным)
   cmp byte ptr [si],'-';первый символ – это минус?
   jne m2          ;если нет – на m2
   not negflag      ;отмечаем, что число отрицательное ;(negflag не равен 0)
   inc si          ;продвигаем si со знака числа к первой цифре
   jmp m3          ;прыгаем на разбор строки цифр
m2:     cmp byte ptr [si],'+';первый символ – это плюс?
   jne m3          ;если нет – на m
   inc si          ;продвигаем si со знака числа к первой цифре
   ;анализируем текущий символ
```

Ассемблер Intel 8086

Ввод целых чисел.

```
m3:      cmp byte ptr [si],cr ;если это cr – строка закончилась, выходим
   ;из цикла разбора ;символов
   je  ex1
   cmp byte ptr [si],'0';если код символа меньше кода '0' – ;это не цифра
   jb  err       ;прыгаем на метку err
   cmp byte ptr [si],'9';если код символа больше кода '9' – ;это не цифра
   ja  err       ;прыгаем на метку err
   mov bx,10      ;домножаем полученное число на основание ;системы счисления
   mul bx
   sub byte ptr [si],'0';вычитаем код символа '0' (получаем очередную цифру)
   add al,[si]     ;добавляем цифру к числу
   adc ah,0
   inc si        ;продвигаем si к следующему символу
   jmp m3        ;организуем цикл функцией 09 выводим сообщение об ошибке
err:     lea  dx,errmsg
   mov ah,9
   int 21h
   jmp start    ;повторяем ввод
ex1:     cmp negflag,0  ;число положительное?
   je  ex      ;если да – выходим
   neg ax      ;меняем знак числа
ex:     ret
IntegerIn  endp
```

Ассемблер Intel 8086

Ввод целых чисел.

```
dosseg
.model small
.stack 200h
.data
    cr = 0dh ;cr присваиваем значение кода символа
    ;возврата каретки (клавиши «Enter»)
    lf = 0ah ;lf присваиваем значение кода символа
    ;перевода строки
    ;в сегменте данных описываем область string для
    ;вводимой строки, сообщение об ошибке errmsg и флаг
    ;отрицательности числа negflag (0 – неотрицательное,
    ;0ffffh - отрицательное)
    string    db  255, 0, 255 dup (?)
    errmsg    db  'Недопустимый символ, можно'
    db  'использовать только цифры',cr,lf,'$'
    negflag   dw  ?
.code
Begin:
    mov ax,@Data
    mov ds, ax

    call IntegerIn
    call IntegerOut

    mov ax, 4C00h
    int 21h

    end Begin
```