**Отчет. Костромин А. А. ПО-324.**

**Задание 1: Двумерные массивы. (Вариант 10).**

В программе определить квадратную матрицу

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 151 | 105 | 109 | 190 | 106 |
| 24 | 69 | 82 | 10 | 131 |
| 118 | 128 | 130 | 107 | 198 |
| 122 | 143 | 144 | 79 | 19 |
| 112 | 65 | 110 | 153 | 144 |

Найти максимальные элементы каждой строки матрицы. Вывести на экран в шестнадцатеричном виде исходную матрицу и найденные значения.

masm

model small

.stack 256

.data

arr db 151, 105, 109, 190, 106

db 24, 69, 82, 10, 131

db 118, 128, 130, 107, 198

db 122, 143, 144, 79, 19

db 112, 65, 110, 153, 144

max db ?

arrMax db 5 dup(?)

tabl db "0123456789ABCDEF"

newstr db 10, 13, '$'

rez db "\*\* $"

nStr macro

mov ah, 9

lea dx, newstr

int 21h

endm

sixSS macro

push ax

shr al, 4

lea bx, tabl

xlat

mov rez, al

pop ax

and al, 0fh

xlat

mov rez + 1, al

mov ah, 9

lea dx, rez

int 21h

endm

search macro

cmp max, al

ja m3

mov max, al

m3: endm

.code

main: mov ax, @data

mov ds, ax

xor si, si

xor di, di

xor bx, bx

mov cx, 5

m1: push cx

mov al, arr[si][bx]

mov max, al

push bx

sixSS

pop bx

inc bx

mov cx, 4

m0: mov al, arr[si][bx]

search

push bx

sixSS

pop bx

inc bx

loop m0

nStr

mov al, max

mov arrMax[di], al

inc di

xor bx, bx

add si, 5

pop cx

loop m1

nStr

xor si, si

mov cx, 5

m2: mov al, arrMax[si]

sixSS

nStr

inc si

loop m2

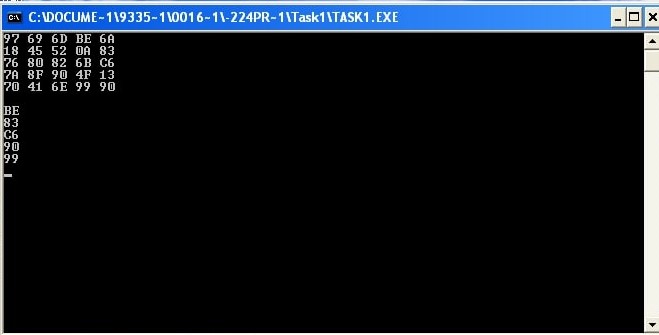
mov ah, 7

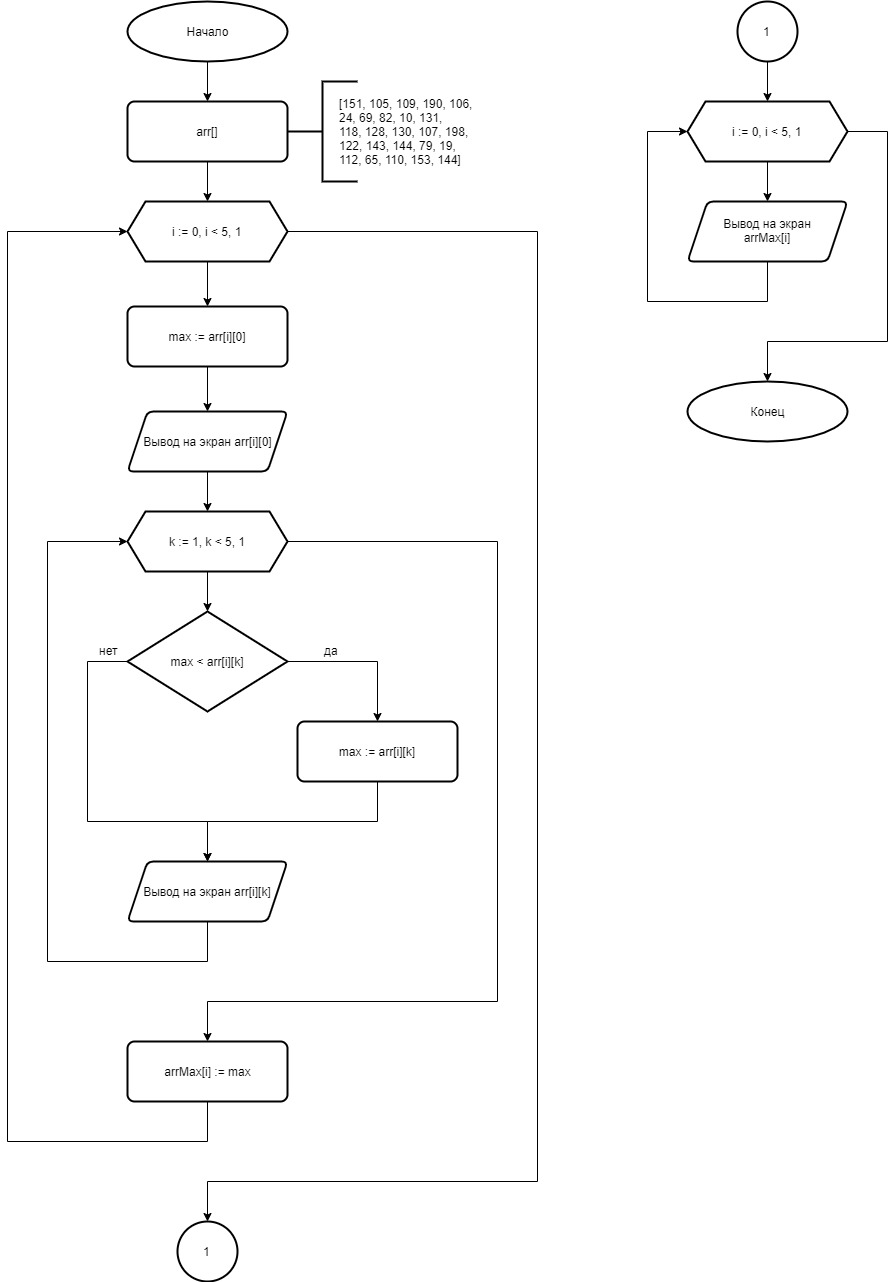
int 21h

mov ax, 4c00h

int 21h

end main





**Задание 2: Текстовый режим.**

При помощи цветных окон нарисовать полосы всех цветов на экране. Вывод полосы реализовать в цикле по нажатию клавиши на клавиатуре.

masm

model small

stack 256

.data

widthCL db 0

widthDL db 4

.code

main: mov ax, @data

mov ds, ax

xor bl, bl

mov cx, 16

m0: mov ah, 6

mov al, 0

mov bh, bl

shl bh, 4

push cx

mov ch, 0

mov cl, widthCL

mov dh, 24

mov dl, widthDL

int 10h

pop cx

add widthCL, 5

add widthDL, 5

inc bl

mov ah, 7

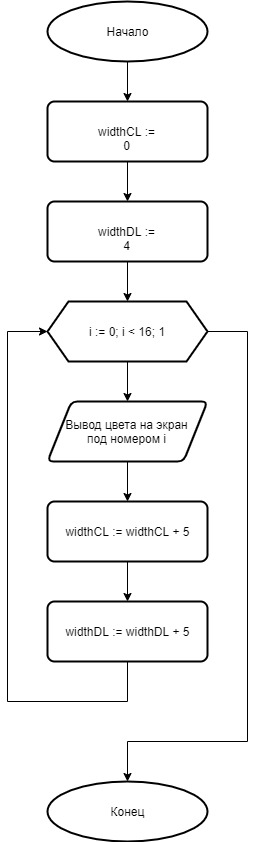
int 21h

loop m0

mov ax, 4c00h

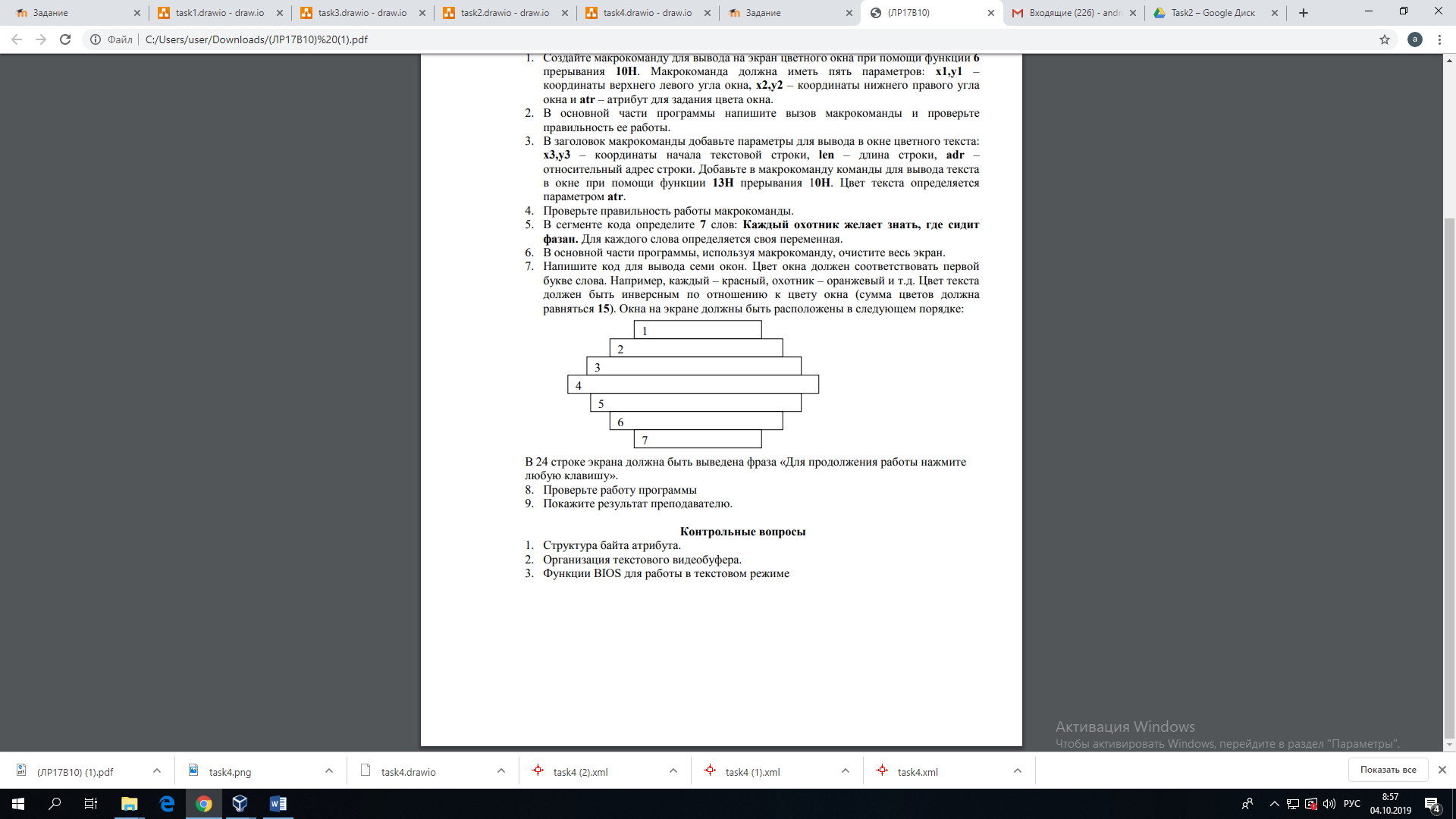
int 21h

end main



**Задание 3: Функции BIOS. (Вариант 10).**

Написать программу, которая выводит на экран фразу: **Каждый охотник желает знать, где сидят фазаны,**позволяющую запомнить порядок следования цветов радуги (первая буква слова кодирует цвет: каждый- красный, охотник – оранжевый, желает – желтый, знать – зеленый, где – голубой, сидят – синий, фазаны – фиолетовый). Каждое слово фразы должно быть выведено при помощи функции 13h в окне наиболее подходящего цвета. Цвет символов должен быть инверсным по отношению к цвету окна. Каждое окно должно выводится после нажатия клавиши на клавиатуре. В 24 строке экрана должна быть выведена фраза «Для продолжения работы нажмите любую клавишу». Фраза должна быть выведена напрямую в видеопамять.



masm

model small

stack 256

.data

kajdiy db "Kajdiy"

lenK = $ - kajdiy

ohotnik db "ohotnik"

lenO = $ - ohotnik

jelaet db "jelaet"

lenJ = $ - jelaet

znat db "znat'"

lenZ = $ - znat

gde db "gde"

lenG = $ - gde

sidit db "sidit"

lenS = $ - sidit

fazan db "fazan"

lenF = $ - fazan

press db "Press any key to continue"

lenPress = $ - press

window macro x1, y1, x2, y2, atr, x3, y3, string, len

mov ah, 6

mov al, 0

mov bh, atr

shl bh, 4

mov ch, y1

mov cl, x1

mov dh, y2

mov dl, x2

int 10h

mov ah, 13h

mov al, 0

mov bl, bh

mov bh, 0

add bl, 15

sub bl, atr

mov cx, len

mov dh, y3

mov dl, x3

push ds

pop es

lea bp, string

int 10h

mov ah, 7

int 21h

endm

clean macro

mov ah, 6

mov al, 0

mov bh, 0

mov cx, 0

mov dh, 24

mov dl, 79

int 10h

endm

.code

main: mov ax, @data

mov ds, ax

clean

window 35, 1, 44, 3, 4, 37, 2, kajdiy, lenK

window 33, 4, 46, 6, 6, 36, 5, ohotnik, lenO

window 31, 7, 48, 9, 14, 37, 8, jelaet, lenJ

window 29, 10, 50, 12, 2, 38, 11, znat, lenZ

window 31, 13, 48, 15, 9, 39, 14, gde, lenG

window 33, 16, 46, 18, 1, 38, 17, sidit, lenS

window 35, 19, 44, 21, 5, 38, 20, fazan, lenF

mov ax, 0b800h

mov es, ax

mov bx, (80\*24)\*2

lea si, press

mov cx, lenPress

m0: mov al, [si]

mov es:[bx], al

inc bx

mov byte ptr es:[bx], 0fh

inc bx

inc si

loop m0

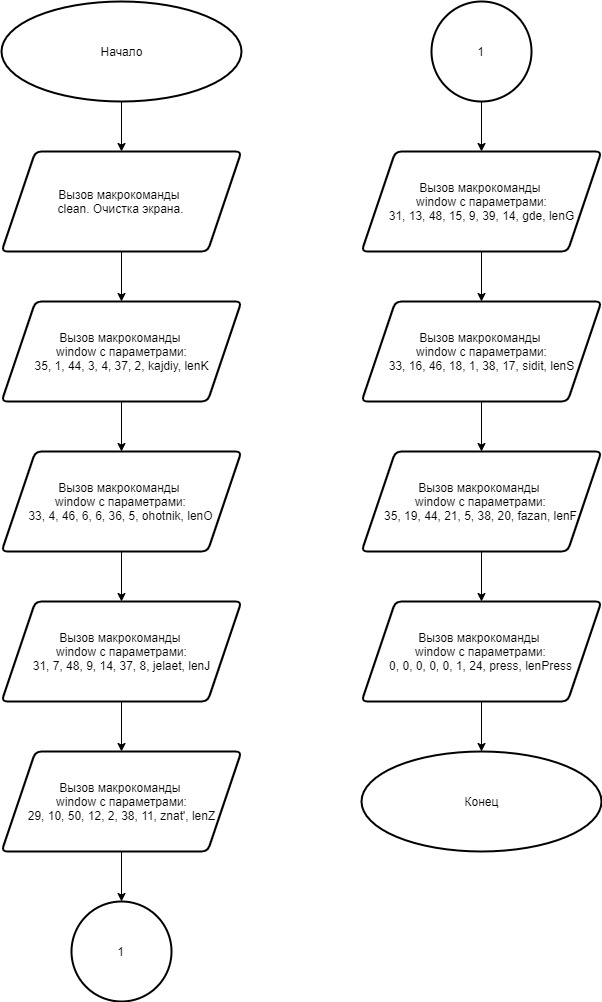
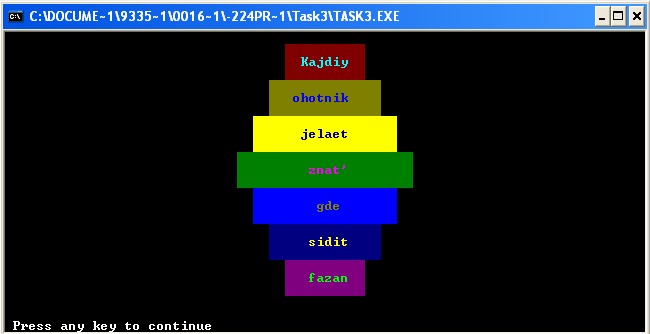
mov ah, 7

int 21h

mov ax, 4c00h

int 21h

end main



**Задание 4: Графический режим. (Вариант 1).**

Ракета

masm

model small

.stack 256

.data

ugol db 1

cycleUgol db ?

addDI db ?

addOrSub db ?

.code

line proc

m0: push cx

mov ah, 0ch

mov bh, 0

mov cx, si

mov dx, di

int 10h

cmp addOrSub, 1

jne toSub

xor bx, bx

mov bl, addDI

add di, bx

jmp skipSub

toSub: xor bx, bx

mov bl, addDI

sub di, bx

skipSub: xor bx, bx

mov bl, ugol

cmp cycleUgol, bl

jne m1

inc si

mov ugol, 0

m1: pop cx

inc ugol

loop m0

mov ugol, 1

ret

line endp

main: mov ax, @data

mov ds, ax

mov ah, 0h

mov al, 12h

int 10h

;нос ракеты

mov si, 330

mov di, 10

mov cx, 50

mov al, 7

mov cycleUgol, 1

mov addDI, 1

mov addOrSub, 1

call line

mov si, 280

mov di, 60

mov cx, 50

mov addOrSub, 0

call line

mov si, 280

mov di, 60

mov cx, 100

mov addDI, 0

mov addOrSub, 1

call line

;иллюминатор

mov si, 323

mov di, 100

mov cx, 16

call line

mov si, 313

mov di, 110

mov cx, 10

mov addDI, 1

mov addOrSub, 0

call line

mov si, 339

mov di, 101

mov cx, 10

mov addOrSub, 1

call line

mov si, 313

mov di, 110

mov cx, 10

mov cycleUgol, 0

call line

mov si, 348

mov di, 111

mov cx, 10

call line

mov si, 313

mov di, 120

mov cx, 10

mov cycleUgol, 1

call line

mov si, 323

mov di, 130

mov cx, 15

mov addDI, 0

call line

mov si, 338

mov di, 130

mov cx, 10

mov addDI, 1

mov addOrSub, 0

call line

;стенки ракеты

mov si, 280

mov di, 60

mov cx, 200

mov cycleUgol, 0

mov addOrSub, 1

call line

mov si, 379

mov di, 60

mov cx, 200

call line

mov si, 280

mov di, 260

mov cx, 100

mov cycleUgol, 1

mov addDI, 0

call line

;низ ракеты

mov si, 256

mov di, 310

mov cx, 50

mov cycleUgol, 2

mov addDI, 1

mov addOrSub, 0

call line

mov si, 379

mov di, 260

mov cx, 50

mov addOrSub, 1

call line

mov si, 256

mov di, 310

mov cx, 60

mov cycleUgol, 0

call line

mov si, 256

mov di, 370

mov cx, 20

mov cycleUgol, 1

mov addDI, 0

call line

mov si, 275

mov di, 370

mov cx, 50

mov cycleUgol, 4

mov addDI, 1

mov addOrSub, 0

call line

mov si, 288

mov di, 320

mov cx, 20

mov cycleUgol, 1

mov addDI, 0

mov addOrSub, 1

call line

mov si, 307

mov di, 320

mov cx, 50

mov cycleUgol, 4

mov addDI, 1

call line

mov si, 320

mov di, 370

mov cx, 20

mov cycleUgol, 1

mov addDI, 0

call line

mov si, 339

mov di, 370

mov cx, 50

mov cycleUgol, 4

mov addDI, 1

mov addOrSub, 0

call line

mov si, 352

mov di, 320

mov cx, 20

mov cycleUgol, 1

mov addDI, 0

mov addOrSub, 1

call line

mov si, 371

mov di, 320

mov cx, 50

mov cycleUgol, 4

mov addDI, 1

call line

mov si, 384

mov di, 370

mov cx, 20

mov cycleUgol, 1

mov addDI, 0

call line

mov si, 403

mov di, 310

mov cx, 60

mov cycleUgol, 0

mov addDI, 1

call line

mov ah, 8

int 21h

mov ah, 0

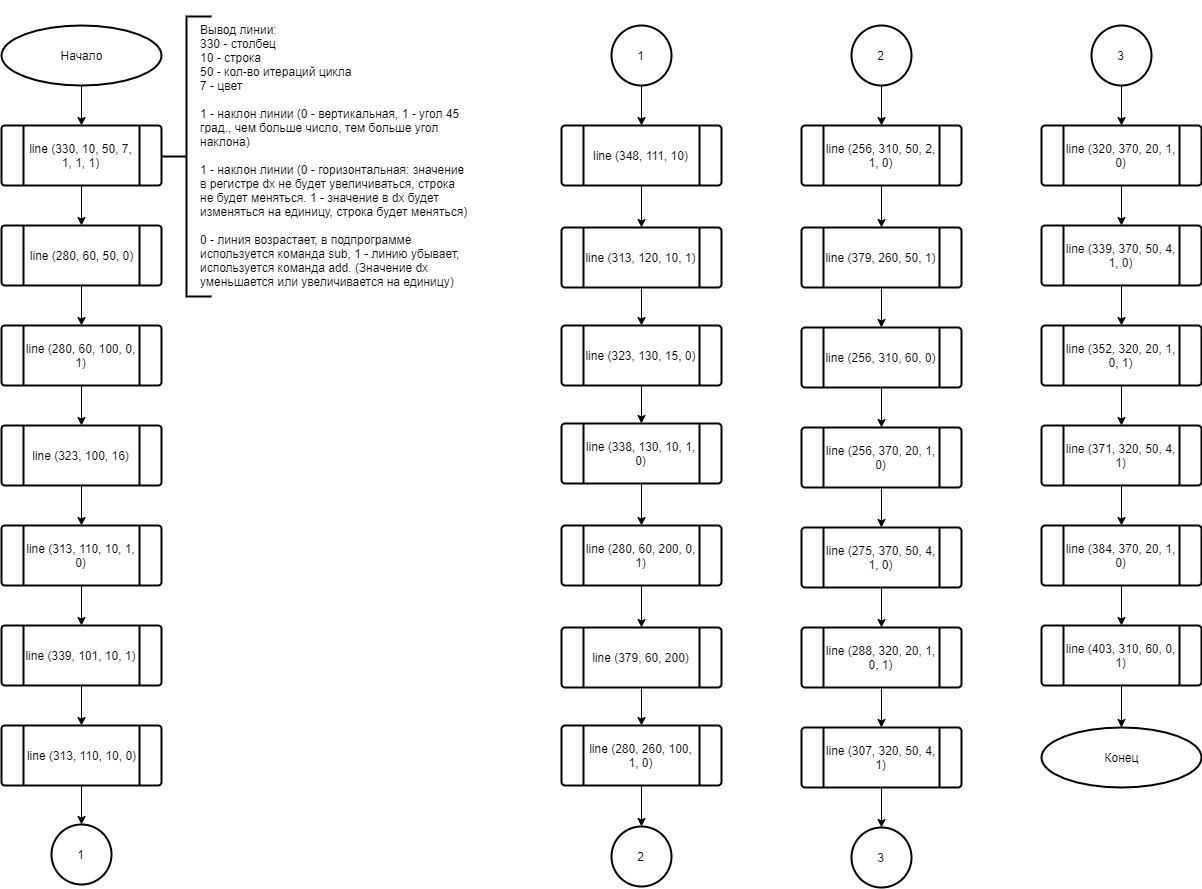
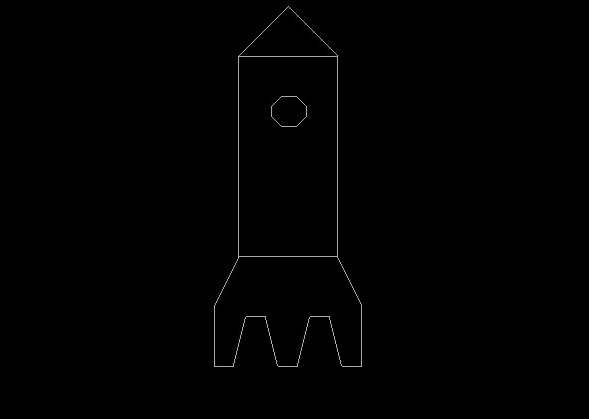
mov al, 3

int 10h

mov ax, 4c00h

int 21h

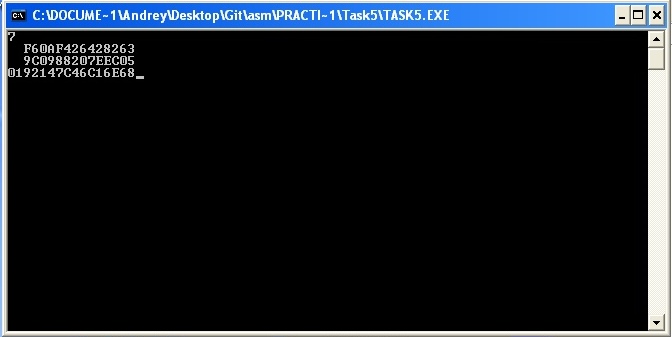
end main

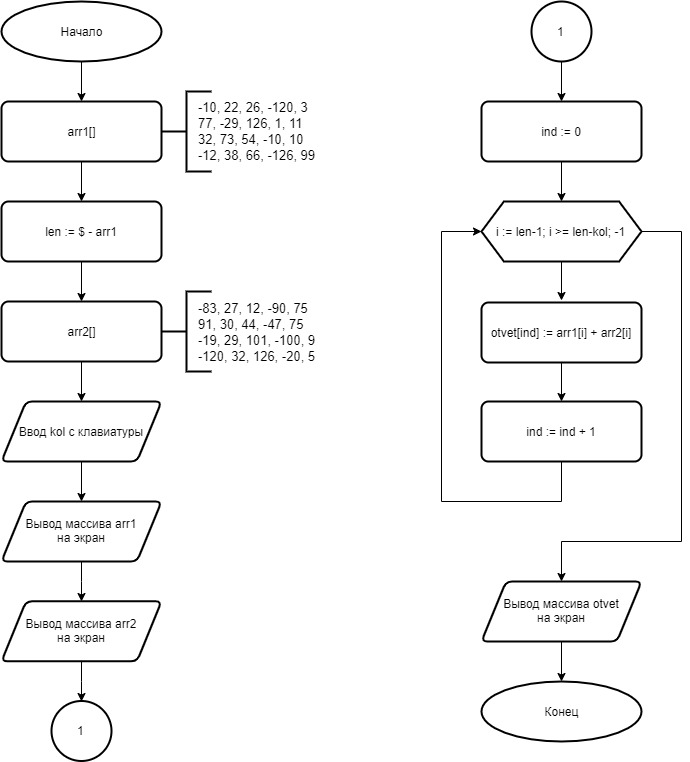


**Задание 5: Сложение с учетом знака чисел размером N байтов**

Изучите текст приведенной ниже подпрограммы. Напишите программу, которая выполняет сложение с учетом знака чисел размером N байтов, используя эту подпрограмму. В сегменте данных определите два массива размерности 20 байт с любыми значениями. С клавиатуры пользователь должен вводить число не больше 20. Это число соответствует количеству байтов, которые нужно использовать для сложения. На экран выведите в шестнадцатеричном виде исходные числа и результат. Отчет о проделанной работе должен содержать текст программы, блок-схему и окно выполнения.

|  |
| --- |
| masm |
|  | model small |
|  | .stack 256 |
|  | .data |
|  | arr1 db -10, 22, 26, -120, 3 |
|  | db 77, -29, 126, 1, 11 |
|  | db 32, 73, 54, -10, 10 |
|  | db -12, 38, 66, -126, 99 |
|  | len = $ - arr1 |
|  | arr2 db -83, 27, 12, -90, 75 |
|  | db 91, 30, 44, -47, 75 |
|  | db -19, 29, 101, -100, 9 |
|  | db -120, 32, 126, -20, 5 |
|  | otvet db 20 dup(?) |
|  | buf db 5 |
|  | db ? |
|  | db 5 dup(?) |
|  | kol db ? |
|  | tabl db "0123456789ABCDEF" |
|  | newstr db 10, 13, '$' |
|  | rez db "\*\*$" |
|  |  |
|  | output macro arr |
|  | local m0 |
|  | xor ax, ax |
|  | mov al, len |
|  | sub al, kol |
|  | mov si, ax |
|  | mov cl, kol |
|  | m0: mov ah, 9 |
|  | mov al, arr[si] |
|  | sixSS |
|  | inc si |
|  | loop m0 |
|  | endm |
|  |  |
|  | nStr macro |
|  | mov ah, 9 |
|  | lea dx, newstr |
|  | int 21h |
|  | endm |
|  |  |
|  | sixSS macro |
|  | push ax |
|  | shr al, 4 |
|  | lea bx, tabl |
|  | xlat |
|  | mov rez, al |
|  | pop ax |
|  | and al, 0fh |
|  | xlat |
|  | mov rez + 1, al |
|  | mov ah, 9 |
|  | lea dx, rez |
|  | int 21h |
|  | endm |
|  |  |
|  | space macro |
|  | local cSpace |
|  | mov cx, 2 |
|  | cSpace: mov ah, 2 |
|  | mov dl, ' ' |
|  | int 21h |
|  | loop cSpace |
|  | endm |
|  |  |
|  | .code |
|  | main: mov ax, @data |
|  | mov ds, ax |
|  |  |
|  | mov ah, 10 |
|  | lea dx, buf |
|  | int 21h |
|  | mov bl, buf + 1 |
|  | mov bh, 0 |
|  | mov buf[bx + 2], '$' |
|  |  |
|  | mov al, buf[1] |
|  | cmp al, 2 |
|  | jne oneNum |
|  | mov ah, buf[2] |
|  | mov al, buf[3] |
|  | and ax, 0f0fh |
|  | aad |
|  | jmp twoNum |
|  | oneNum: mov al, buf[2] |
|  | and ax, 0fh |
|  | aad |
|  |  |
|  | twoNum: mov kol, al |
|  | space |
|  | output arr1 |
|  | nStr |
|  | space |
|  | output arr2 |
|  | nStr |
|  |  |
|  | mov di, 1 |
|  | mov si, len-1 |
|  | mov cl, kol |
|  | sum: mov al, arr2[si] |
|  | adc arr1[si], al |
|  | mov al, arr1[si] |
|  | mov otvet[di], al |
|  | dec si |
|  | inc di |
|  | loop sum |
|  |  |
|  | jnc noPer |
|  | adc otvet[0], 0 |
|  | mov al, otvet[0] |
|  | sixSS |
|  | jmp noSpace |
|  | noPer: space |
|  | noSpace:mov si, word ptr kol |
|  | and si, 00ffh |
|  | mov cl, kol |
|  | outOtv: mov al, otvet[si] |
|  | sixSS |
|  | dec si |
|  | loop outOtv |
|  |  |
|  | mov ah, 7 |
|  | int 21h |
|  | mov ax, 4c00h |
|  | int 21h |
|  | end main |

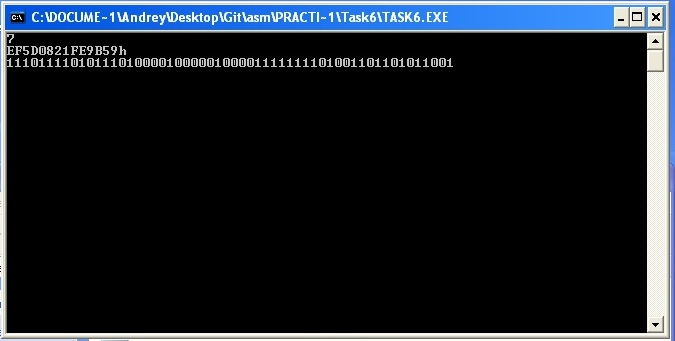


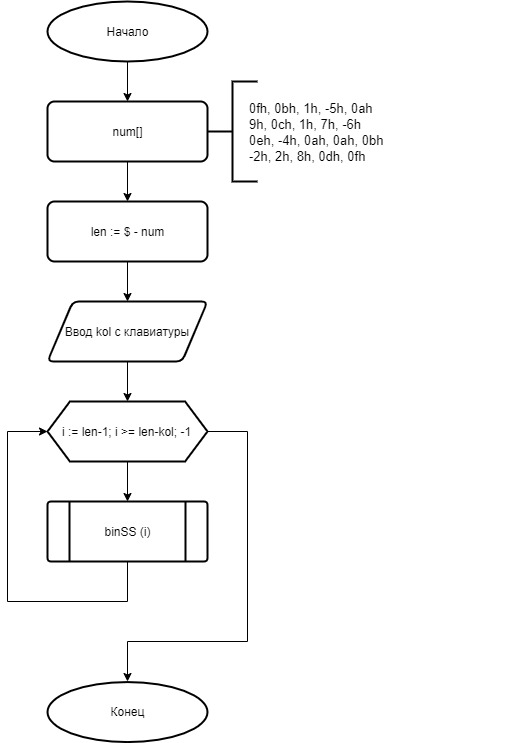


**Задание 6: Вычисление двоичного дополнения числа размером N байтов**

Напишите подпрограмму, позволяющую вычислить двоичное дополнение числа. В сегменте данных определите массив из 20 байт. С клавиатуры вводится число не больше 20 (размер числа) и вычисляется при помощи [подпрограммы](http://do.urtk.su/mod/lesson/view.php?id=4238) двоичное дополнение нужной размерности. На экран нужно вывести исходное число и его дополнение в двоичном виде

|  |
| --- |
| masm |
|  | model small |
|  | .stack 256 |
|  | .data |
|  | kol db ? |
|  | num db 4fh, 0fbh, 1fh, -5h, 0a2h |
|  | db 99h, 0fch, 01h, 0eeh, -6h |
|  | db 03eh, -4h, 75h, 59h, 9bh |
|  | db -2h, 21h, 08h, 5dh, 0efh |
|  | len = $ - num |
|  | buf db 5 |
|  | db ? |
|  | db 5 dup(?) |
|  | tabl db "0123456789ABCDEF" |
|  | newstr db 10, 13, '$' |
|  | rez db "\*\*$" |
|  |  |
|  | output macro |
|  | mov si, 19 |
|  | mov cl, kol |
|  | m0: mov ah, 9 |
|  | mov al, num[si] |
|  | sixSS |
|  | dec si |
|  | loop m0 |
|  | endm |
|  |  |
|  | sixSS macro |
|  | push ax |
|  | shr al, 4 |
|  | lea bx, tabl |
|  | xlat |
|  | mov rez, al |
|  | pop ax |
|  | and al, 0fh |
|  | xlat |
|  | mov rez + 1, al |
|  | mov ah, 9 |
|  | lea dx, rez |
|  | int 21h |
|  | endm |
|  |  |
|  | nStr macro |
|  | mov ah, 9 |
|  | lea dx, newstr |
|  | int 21h |
|  | endm |
|  |  |
|  | .code |
|  |  |
|  | binSS proc |
|  | mov cx, 8 |
|  | mov ah, 2 |
|  | m1: rol num[si], 1 |
|  | mov dl, '0' |
|  | adc dl, 0 |
|  | int 21h |
|  | loop m1 |
|  | ret |
|  | binSS endp |
|  |  |
|  | main: mov ax, @data |
|  | mov ds, ax |
|  |  |
|  | mov ah, 10 |
|  | lea dx, buf |
|  | int 21h |
|  | mov bl, buf + 1 |
|  | mov bh, 0 |
|  | mov buf[bx + 2], '$' |
|  |  |
|  | mov al, buf[1] |
|  | cmp al, 2 |
|  | jne oneNum |
|  | mov ah, buf[2] |
|  | mov al, buf[3] |
|  | and ax, 0f0fh |
|  | aad |
|  | jmp twoNum |
|  | oneNum: mov al, buf[2] |
|  | and ax, 0fh |
|  | aad |
|  |  |
|  | twoNum: mov kol, al |
|  | nStr |
|  | output |
|  | mov ah, 2 |
|  | mov dl, 'h' |
|  | int 21h |
|  | nStr |
|  |  |
|  | mov si, len-1 |
|  | mov cl, kol |
|  | m5: push cx |
|  | call binSS |
|  | pop cx |
|  | dec si |
|  | loop m5 |
|  |  |
|  | mov ah, 7 |
|  | int 21h |
|  | mov ax, 4c00h |
|  | int 21h |
|  | end main |





**Задание 7: Умножение чисел размером N и М байтов без учета знака**

Напишите программу, в которой реализуйте алгоритм умножения столбиком для чисел размером N и М байтов без учета знака. N и M определите через оператор = в программе. На экран выведите исходные значения в шестнадцатеричном виде и полученный результат.

|  |
| --- |
| masm |
|  | model small |
|  | .stack 256 |
|  | .data |
|  | num1 db 12h, 0f3h, 99h, 0cah, 51h, 40h, 75h, 0f1h, 0ddh, 84h |
|  | len = $ - num1 |
|  | num2 db 39h, 1ah, 99h, 0cdh, 7ah, 0dah, 0adh, 59h, 73h, 0eah |
|  | N = 5 |
|  | M = 3 |
|  | tabl db "0123456789ABCDEF" |
|  | newstr db 10, 13, '$' |
|  | rez db "\*\*$" |
|  | otvet db 20 dup(0) |
|  | shift db 0 |
|  | output macro num, cyc |
|  | local one |
|  | mov si, len-cyc |
|  | mov cl, cyc |
|  | one:mov al, num[si] |
|  | sixSS |
|  | inc si |
|  | loop one |
|  | endm |
|  |  |
|  | sixSS macro |
|  | push ax |
|  | shr al, 4 |
|  | lea bx, tabl |
|  | xlat |
|  | mov rez, al |
|  | pop ax |
|  | and al, 0fh |
|  | xlat |
|  | mov rez + 1, al |
|  | mov ah, 9 |
|  | lea dx, rez |
|  | int 21h |
|  | endm |
|  |  |
|  | nStr macro |
|  | mov ah, 9 |
|  | lea dx, newstr |
|  | int 21h |
|  | endm |
|  |  |
|  | space macro cycle |
|  | local cSpace |
|  | mov cx, cycle |
|  | cSpace: mov ah, 2 |
|  | mov dl, ' ' |
|  | int 21h |
|  | loop cSpace |
|  | endm |
|  |  |
|  | .code |
|  | main: mov ax, @data |
|  | mov ds, ax |
|  |  |
|  | space M\*2 |
|  | output num1, N |
|  | nStr |
|  | space N\*2 |
|  | output num2, M |
|  | nStr |
|  |  |
|  | mov di, len-1 |
|  | mov cl, M |
|  | nextMul:push cx |
|  | mov si, len-1 |
|  | mov cl, N |
|  | mov bx, N+M-1 |
|  |  |
|  | multi: push bx |
|  | mov al, num1[si] |
|  | mov bl, num2[di] |
|  | mul bl |
|  | pop bx |
|  | sub bx, word ptr shift |
|  | adc otvet[bx], al |
|  | adc otvet[bx-1], ah |
|  | adc otvet[bx-2], 0 |
|  | add bx, word ptr shift |
|  | dec si |
|  | dec bx |
|  | loop multi |
|  |  |
|  | dec di |
|  | pop cx |
|  | inc shift |
|  | loop nextMul |
|  |  |
|  | xor si, si |
|  | mov cx, N+M |
|  | print: mov al, otvet[si] |
|  | sixSS |
|  | inc si |
|  | loop print |
|  |  |
|  | mov ah, 7 |
|  | int 21h |
|  | mov ax, 4c00h |
|  | int 21h |
|  | end main |

