

ACT 1

```
org 100h

mov ah, 0
mov al, 3
int 10h

mov cx, 10
mov bl, 1

; Create a loop
print_Loop:

    ; Convert number to ascii, by passing 0 because '0' have a value so it start there
    mov al, bl
    add al, '0'
    mov ah, 0eh
    int 10h

    ; print new line
    mov al, 13
    mov ah, 0eh
    int 10h

    mov al, 10
    mov ah, 0eh
    int 10h

    inc bl

    ; Jump in this while decrementing the c or counter
    loop print_Loop

mov ah, 0
int 16h

mov ax 4c00h
int 21h
```

The image shows a Windows desktop with two windows. The background window is a Notepad++ editor with assembly code for a program that prints a grid of 'a's in different colors. The foreground window is an x86-64 emulator (emu8086) running the code. The emulator's CPU window shows the instruction 'mov dx, 0' being executed. The emulator's screen window shows a grid of 'a's in various colors, matching the code's logic.

The screenshot displays a DOS emulator window titled "emulator: calc.com_". The main window shows the execution of a program named "calc.com". The program's output is displayed in a black window titled "emulator screen (80x25 chars)".

The program's code is shown in the background, featuring macros for input, calculation, and output. The main code block is as follows:

```

name "calc"
; this sample gets two numbers from the user,
; then it calculates the sum of these numbers,
; and prints it out.

name "calc"
; these macros are copied from emu0086.inc ;;;
; this macro prints a string that is given as a parameter, example:
PRINTN 'hello world!'
; the same as PRINT, but new line is automatically added.
PRINTN MACRO sdat
LOCAL next_char, s_dcl, printed, skip_dcl
PUSH AX ; store registers...
PUSH SI ;
JMP skip_dcl ; skip declaration.
s_dcl DB sdat, 0Ah, 0Ah, 0
skip_dcl:
LEA SI, s_dcl
next_char:
MOV AL, CS:[SI]
CMP AL, 0
JZ printed
INC SI
MOV AH, 0Eh ; teletype function.
INT 10h
JMP next_char
printed:
POP SI ; re-store registers...
POP AX ;
ENDM

; this macro prints a char in AL and advances
; the current cursor position:
PUTC MACRO char
PUSH AX
MOV AL, char
MOV AH, 0Eh
INT 10h
POP AX
ENDM

org 100h
jmp start ; skip data.
msg1 db 0Ah, 0Ah, 'input numbers in this range: [-32768..32767]', 0Ah, 0Ah
db 0Ah, 0Ah, 'enter first number: $'

```

The emulator screen shows the following output:

```

input numbers in this range: [-32768..32767]
enter first number: 300
enter second number: 233
the sum is: 533

```

The emulator window also shows a menu bar with options: file, math, debug, view, external, virtual devices, virtual drive, help. Below the menu bar are buttons for Load, reload, step back, and a yellow button labeled "waiting for input" with a "stop" button next to it. A "step delay ms: 0" label is also visible.